



Durham E-Theses

Discontinuous Galerkin discretised level set methods with applications to topology optimisation

ADAMS, THOMAS,DEAN

How to cite:

ADAMS, THOMAS,DEAN (2020) *Discontinuous Galerkin discretised level set methods with applications to topology optimisation*, Durham theses, Durham University. Available at Durham E-Theses Online: <http://etheses.dur.ac.uk/13686/>

Use policy

The full-text may be used and/or reproduced, and given to third parties in any format or medium, without prior permission or charge, for personal research or study, educational, or not-for-profit purposes provided that:

- a full bibliographic reference is made to the original source
- a [link](#) is made to the metadata record in Durham E-Theses
- the full-text is not changed in any way

The full-text must not be sold in any format or medium without the formal permission of the copyright holders.

Please consult the [full Durham E-Theses policy](#) for further details.

Discontinuous Galerkin discretised level set methods with applications to topology optimisation

Thomas Adams

Thesis submitted towards the
degree of Doctor of Philosophy



Department of Engineering
Durham University
United Kingdom

March 2020

Discontinuous Galerkin discretised level set methods with applications to topology optimisation

Thomas Adams

Abstract

This thesis presents research concerning level set methods discretised using discontinuous Galerkin (DG) methods. Whilst the context of this work is level set based topology optimisation, the main outcomes of the research concern advancements which are agnostic of application. The first of these outcomes are the development of two novel DG discretised PDE based level set reinitialisation techniques, the so called Elliptic and Parabolic reinitialisation methods, which are shown through experiment to be robust and satisfy theoretical optimal rates of convergence. A novel Runge-Kutta DG discretisation of a simplified level set evolution equation is presented which is shown through experiment to be high-order accurate for smooth problems (optimal error estimates do not yet exist in the literature based on the knowledge of the author). Narrow band level set methods are investigated, and a novel method for extending the level set function outside of the narrow band, based on the proposed Elliptic Reinitialisation method, is presented. Finally, a novel *hp*-adaptive scheme is developed for the DG discretised level set method driven by the degree with which the level set function can locally satisfy the Eikonal equation defining the level set reinitialisation problem. These component parts are thus combined to form a proposed DG discretised level set methodology, the efficacy of which is evaluated through the solution of numerous example problems. The thesis is concluded with a brief exploration of the proposed method for a minimum compliance design problem.

Contents

Abstract	2
Contents	3
List of Figures	7
List of Tables	13
List of Algorithms	14
Acronyms	15
Nomenclature	17
Declaration	23
Acknowledgements	24
1 Introduction	25
1.1 Context and overview	25
1.2 Thesis outline	30
2 Discontinuous Galerkin Methods	33
2.1 Overview	33
2.2 Mathematical preliminaries	36
2.3 Literature review	37
2.3.1 DG methods for diffusive PDEs	38
2.3.1.1 DG discretisations for diffusive PDEs	39
2.3.2 DG methods for advective PDEs	44
2.3.2.1 DG discretisations for advective PDEs	46
2.4 Summary	48
3 The Level Set Method	49
3.1 Overview	49
3.2 Mathematical preliminaries	50
3.2.1 Level set initialisation and reinitialisation	51
3.2.2 Narrow banded level set methods	52
3.3 Literature review: discontinuous Galerkin level set methods	52
3.4 Summary	54

4	Level Set Reinitialisation	55
4.1	Literature review: level set reinitialisation	56
4.2	Eikonal minimisation based reinitialisation: strong form	61
4.3	Eikonal Minimising Elliptic Reinitialisation method	62
4.3.1	Picard linearisation and spatial discretisation	63
4.3.2	Integration on an immersed implicit interface	63
4.3.3	Boundary conditions on implicit interfaces	67
4.3.4	Modified objective functionals for the minimisation based reinitialisation problem	71
4.4	Elliptic Reinitialisation method	75
4.4.1	Elliptic Reinitialisation method: numerical examples	76
4.4.1.1	Error measures	76
4.4.1.2	h-convergence study: circular interface	77
4.4.1.3	h-convergence study: circular interface with narrow band	79
4.4.1.4	h-convergence study: smooth star interface	81
4.4.1.5	h-convergence study: multiple interfaces	83
4.5	Parabolic Reinitialisation method	85
4.5.1	Parabolic reinitialisation method: spatial semi-discretisation	85
4.5.2	Parabolic reinitialisation method: full discretisations	86
4.5.2.1	Fully explicit	87
4.5.2.2	Semi-implicit	87
4.5.2.3	Fully implicit	88
4.5.3	Parabolic Reinitialisation: numerical examples	89
4.5.3.1	Investigation into critical time step	89
4.5.3.2	h-convergence study: circular interface	92
4.5.3.3	h-convergence study: smooth star interface	94
4.5.3.4	h-convergence study: multiple interfaces	95
4.6	Summary	97
5	Level Set Evolution	98
5.1	The level set evolution equation	98
5.1.1	Level set evolution equation: spatial semi-discretisation	99
5.1.2	Level set evolution equation: full discretisation	99
5.2	Narrow band level set method	101
5.2.1	Narrow band extrapolation	102
5.3	Anderson acceleration	104
5.4	A narrow banded discontinuous Galerkin level set method	108
5.4.1	Algorithm	108
5.4.2	Narrow banded discontinuous Galerkin level set method: numerical examples	109
5.4.2.1	h-convergence study: growing-shrinking-growing circular interface	112
5.4.2.2	h-convergence study: translating circular interface	116
5.4.2.3	h-convergence study: shearing circular interface	120
5.4.2.4	h-convergence study: two merging circular interfaces	123
5.5	Summary	126
6	Level Set Method: Adaptive Mesh Refinement	129
6.1	Literature review: adaptive mesh refinement for level set methods	129
6.2	Refinement criterion	131
6.3	Refinement strategy	132
6.3.1	Initial flagging strategy	134

6.3.2	hp-steering criterion	134
6.3.3	Enforcement of refinement limits	136
6.3.4	Flag smoothing	137
6.3.5	Mesh refinement	138
6.3.6	Project the solution onto the new mesh	138
6.3.7	Update the narrow band	138
6.3.8	Update the level set function	139
6.4	Level set reinitialisation with adaptive mesh refinement: numerical examples . .	139
6.4.1	hp-convergence study: circular interface	140
6.4.2	hp-convergence study: square interface	143
6.4.3	hp-convergence study: smooth star interface	144
6.4.4	hp-convergence study: elliptic interface	149
6.4.5	hp-convergence study: multiple interfaces	154
6.5	hp-adaptive level set method	158
6.5.1	Evolution algorithm	158
6.6	Level set evolution with adaptive mesh refinement: numerical examples	158
6.6.1	hp-convergence study: growing-shrinking-growing circle	158
6.6.1.1	Initialisation	158
6.6.1.2	Evolution	161
6.6.2	Growing-shrinking-growing problem with pure h-refinement	164
6.6.2.1	Initialisation	164
6.6.2.2	Evolution	165
6.6.3	hp-convergence study: translating circle	168
6.6.3.1	Initialisation	168
6.6.3.2	Evolution	168
6.6.4	hp-convergence study: shearing circle	172
6.6.4.1	Initialisation	172
6.6.4.2	Evolution	173
6.6.5	hp-convergence study: merging circles	177
6.6.5.1	Initialisation	177
6.6.5.2	Evolution	179
6.7	Summary	182
7	Topology Optimisation	184
7.1	Literature review: numerical methods for topology optimisation	185
7.2	Topology optimisation	188
7.2.1	Level set based topology optimisation	188
7.2.2	Shape sensitivity analysis	190
7.2.3	Computing the advection velocity field for a generic level set based topology optimisation	190
7.3	Minimum compliance of a linear elastic structure subject to a volume constraint	191
7.3.1	Linear elasticity equations	191
7.3.2	Minimum Compliance	193
7.3.3	Volume constraint	194
7.4	Discontinuous Galerkin based level set topology optimisation: numerical example	195
7.4.1	Algorithm	195
7.4.2	Numerical example: minimum compliance design of a cantilever beam subject to a volume constraint	197
7.5	Summary	203

8	Conclusions	206
8.1	Key developments	206
8.2	Suggestions for future work	207
A	Divergence Free Basis Functions for Müller’s Method	210
B	Butcher Tableaus	211
	References	214

List of Figures

3.1	A circular interface on a square domain, defined using the level set method. . . .	50
4.1	A quadrilateral element, τ , intersected by a level set interface, $\Gamma(\phi)$, which divides the element into two subdomains A and $\tau \setminus A$	64
4.2	Effect of the value of the penalty parameter, γ_D , on the solution of a reinitialisation problem at the level set interface. The solid line shows the level set function, the dashed line shows the analytical solution, and the horizontal line shows the $\phi = 0$ plane.	68
4.3	Examples showing problem dependency of the penalty parameter, γ_D . The solid line shows the level set function, the dashed line shows the analytical solution, and the horizontal line shows the $\phi = 0$ plane.	68
4.4	Effect of using too large of an interpolation space for the Lagrange multipliers to enforce a Dirichlet boundary condition. The solid line shows the level set function, the dashed line shows the analytical solution, and the horizontal line shows the $\phi = 0$ plane.	70
4.5	Three different objective functionals and their corresponding diffusion rates. . . .	72
4.6	Converged solutions to a simple problem demonstrating how each of the different objective functionals behave in the presence of small gradients. The solid line shows the level set function, the dashed line shows the analytical solution, and the horizontal line shows the $\phi = 0$ plane.	74
4.7	Error data and convergence rates for the circular interface problem in the domain $\Omega = (-2, 2)^2$, solved using the Elliptic Reinitialisation method.	78
4.8	Domain configuration for the circular interface reinitialisation problem where the singular part has been removed from the domain, i.e. $\Omega = (-2, 2)^2 \setminus (-0.4, 0.4)^2$	79
4.9	Error data and convergence rates for the circular interface reinitialisation problem on the domain $\Omega = (-2, 2)^2 \setminus (-0.4, 0.4)^2$, solved using the Elliptic Reinitialisation method.	80
4.10	Domain configuration for the smooth star interface reinitialisation problem, initialised as stated in Equation (4.61), where $\Omega = (-2, 2)^2$	82
4.11	Error data normalised by narrow band area, A_{NB} , and associated convergence rates for the smooth star interface reinitialisation problem, initialised as stated in Equation (4.61), in the domain $\Omega = (-2, 2)^2$, with narrow band, solved using the Elliptic Reinitialisation method.	82
4.12	Domain configuration for the multiple interface problem where, $\Omega = (-2, 2)^2$	84
4.13	Error data normalised by narrow band area, A_{NB} , and associated convergence rates for the multiple interfaces interface reinitialisation problem, initialised as stated in Equation (4.61), in the domain $\Omega = (-2, 2)^2$, with narrow band, solved using the Elliptic Reinitialisation method.	84

4.14	Variation of error in the L^2 norm of the converged solution to the circular interface reinitialisation problem with time step, for the three discretisations of the Parabolic Reinitialisation method on a fixed mesh with $h = 0.2$ and $p = 1$	90
4.15	Variation of critical time step and number of iterations for the three discretisations of the Parabolic Reinitialisation method, for the circular interface reinitialisation problem.	91
4.16	Error data and convergence rates for the circular interface reinitialisation problem in the domain $\Omega = (-2, 2)^2 \setminus (-0.4, 0.4)^2$, solved using the Parabolic Reinitialisation method.	93
4.17	Error data normalised by narrow band area, A_{NB} , and associated convergence rates for the smooth star interface reinitialisation problem in the domain $\Omega = (-2, 2)^2$, using a narrow band approach, solved using the Parabolic Reinitialisation method.	95
4.18	Error data normalised by the narrow band area, A_{NB} , and associated convergence rates for the multiple interfaces reinitialisation problem in the domain $\Omega = (-2, 2)^2$, using a narrow band approach, solved using the Parabolic Reinitialisation method.	96
5.1	Advection velocity field for the growing and shrinking circle on a square domain.	110
5.2	Advection velocity field for the translating circle problem.	110
5.3	Advection velocity field for the circle under homogeneous strain.	111
5.4	Advection velocity field for the growing and shrinking of the two merging circles problem.	111
5.5	Error data normalised by narrow band area, A_{NB} , and associated convergence rates for the growing-shrinking-growing (GSG) circular interface evolution problem on narrow banded Cartesian meshes at time, $t_e = 0.6$	113
5.6	Error and area ratio over time for the GSG circular interface evolution problem at time, $t_e = 0.6$, on a narrow banded Cartesian mesh where $h = 0.05$ and $p = 3$. The area ratio compares the area of the shape enclosed by the interface, A , with the area contained inside the initial interface, A_0	114
5.7	Computed and analytical interface position over time for the GSG circular interface evolution problem, on the narrow banded Cartesian mesh where $h = 0.05$ and $p = 3$	115
5.8	Error data normalised by narrow band area, A_{NB} , and associated convergence rates for the translating circular interface problem on narrow banded Cartesian meshes at time, $t_e = 0.5$	117
5.9	Error and area ratio over time for the translating circular interface problem on the narrow banded Cartesian mesh where $h = 0.01$ and $p = 3$. The area ratio compares the area of the shape enclosed by the interface, A , with the area contained inside the initial interface, A_0	118
5.10	Computed and analytical interface position over time for the translating circle problem, on the narrow banded Cartesian mesh where $h = 0.01$ and $p = 3$	119
5.11	Error data normalised by narrow band area, A_{NB} , and associated convergence rates for shearing circular interface problem on narrow banded Cartesian meshes at time, $t_e = 0.5$	121
5.12	Distribution of the signed distance error over the narrow banded Cartesian mesh where $p = 3$ and $h = 0.025$, at time $t_e = 0.5$, for the shearing circle problem. The thick black line denotes implied position of the interface.	122

5.13	Error and area ratio over time for the shearing circular interface problem, on the narrow banded Cartesian mesh where $h = 0.025$ and $p = 3$. The area ratio compares the area of the shape enclosed by the interface, A , with the area contained inside the initial interface, A_0	123
5.14	Computed and analytical interface position over time for the shearing circle problem, on the narrow banded Cartesian mesh where $h = 0.025$ and $p = 3$	124
5.15	Error data normalised by narrow band area, A_{NB} , and associated convergence rates for the signed distance error measure, for the merging circles problem on narrow banded Cartesian meshes at time, $t_e = 0.5$	125
5.16	Signed distance error against pseudotime for the merging circles problem on the narrow banded Cartesian mesh where $p = 3$ and $h = 0.0375$	126
5.17	Distribution of the signed distance error over the narrow band Cartesian mesh where $h = 0.0375$ and $p = 3$, for two consecutive time steps of the merging circles problem, across which the solution becomes singular. The thick black line denotes the implied position of the interface.	127
5.18	Computed interface position over time for the merging circle problem, on the narrow banded Cartesian mesh where $h = 0.0375$ and $p = 3$	128
6.1	Error against the square root of the number of degrees of freedom (ndof), for the solution of the circular interface reinitialisation problem, on an hp -adaptively refined mesh. The red line denotes the error tolerance defining one of the stopping criteria, which can change as the area of narrow band changes.	141
6.2	Final computed mesh configuration for the circular interface reinitialisation problem on an hp -adaptively refined mesh where the colour of the element denotes the polynomial order of that element. The thick black line denotes the computed interface position.	142
6.3	Relative position of the initial interface and the domain boundaries for the square interface reinitialisation problem on the domain $\Omega = (-2, 2)^2$	144
6.4	Error against the square root of the number of degrees of freedom (ndof), for the solution of the square interface reinitialisation problem, on an hp -adaptively refined mesh. The red line denotes the error tolerance defining one of the stopping criteria, which can change as the area of narrow band changes.	145
6.5	Final computed mesh configuration for the square interface reinitialisation problem on an hp -adaptively refined mesh where the colour of the element denotes the polynomial order of that element. The thick black line denotes the computed interface position.	146
6.6	Relative position of the initial interface and the domain boundaries for the smooth star interface reinitialisation problem, defined by Equation (6.4), on the domain $\Omega = (-2, 2)^2$	147
6.7	Error against the square root of the number of degrees of freedom (ndof), for the solution of the smooth star interface reinitialisation problem, initialised as in Equation (6.4), on an hp -adaptively refined mesh. The red line denotes the error tolerance defining one of the stopping criteria, which can change as the area of narrow band changes.	148
6.8	Final computed mesh configuration for the smooth star interface reinitialisation problem, initialised as in Equation (6.4), on an hp -adaptively refined mesh where the colour of the element denotes the polynomial order of that element. The thick black line denotes the computed interface position.	149
6.9	Relative position of the initial interface and the domain boundaries for the elliptical interface reinitialisation problem on the domain $\Omega = (-2, 2)^2$	150

6.10	Error against the square root of the number of degrees of freedom (ndof), for the solution of the elliptical interface reinitialisation problem, on an hp -adaptively refined mesh. The red line denotes the error tolerance defining one of the stopping criteria, which can change as the area of narrow band changes.	151
6.11	Error and curvature distributions for the elliptic interface reinitialisation problem on a narrow banded Cartesian mesh after 100 iterations.	152
6.12	Error and curvature distributions for the circular interface reinitialisation problem on a narrow banded Cartesian mesh after 100 iterations.	153
6.13	Final computed mesh configuration for the elliptical interface reinitialisation problem on an hp -adaptively refined mesh where the colour of the element denotes the polynomial order of that element. The thick black line denotes the computed interface position.	154
6.14	Error against the square root of the number of degrees of freedom (ndof), for the solution of the multiple interfaces reinitialisation problem, on an hp -adaptively refined mesh. The red line denotes the error tolerance defining one of the stopping criteria, which can change as the area of narrow band changes.	156
6.15	Distribution of L^2 error over the mesh for the multiple interfaces reinitialisation problem on a mesh where the analytical solution does not align with the unique viscosity solution.	156
6.16	Final computed mesh configuration for the multiple interfaces reinitialisation problem on an hp -adaptively refined mesh where the colour of the element denotes the polynomial order of that element. The thick black line denotes the computed interface position.	157
6.17	Error against the square root of the number of degrees of freedom (ndof), for the initialisation of the growing-shrinking-growing circular interface problem, on an hp -adaptively refined mesh.	160
6.18	Final computed mesh configuration for the initialisation of the GSG circular interface problem on an hp -adaptively refined mesh where the colour of the element denotes the polynomial order of that element. The thick black line denotes the computed interface position.	161
6.19	Error over pseudotime for the GSG circular interface problem on an hp -adaptively refined mesh. The red line denotes the error tolerance defining one of the stopping criteria, which can change as the area of narrow band changes.	162
6.20	Configuration of mesh and interface position over time for the GSG circular interface evolution problem, on an hp -adaptively refined mesh. The thick black line denotes the computed interface position.	163
6.21	Comparison of errors against the square root of the number of degrees of freedom (ndof), for the initialisation of the growing-shrinking-growing circular interface problem, on an h -adaptively and hp -adaptively refined mesh.	165
6.22	Configuration of mesh and interface position over time for the GSG circular interface evolution problem, on an h -adaptively and hp -adaptively refined mesh. The thick black line, in both cases, denotes the computed interface position.	166
6.23	Comparison of errors over pseudotime for the GSG circular interface problem on an h -adaptively and hp -adaptively refined mesh. The red line denotes the error tolerance defining one of the stopping criteria, which can change as the area of narrow band changes.	166
6.24	Configuration of mesh and interface position over time for the GSG circular interface evolution problem, on an h -adaptively refined mesh. The thick black line denotes the computed interface position. Due to the rotational symmetry of the problem only displayed is the region $(0, 1.5)^2$, to improve visibility for such small h	167

6.25	Error against the square root of the number of degrees of freedom (ndof), for the initialisation of the translating circular interface problem, on an <i>hp</i> -adaptively refined mesh.	169
6.26	Final computed mesh configuration for the initialisation of the translating circular interface problem on an <i>hp</i> -adaptively refined mesh where the colour of the element denotes the polynomial order of that element. The thick black line denotes the computed interface position.	169
6.27	Error over pseudotime for the translating circular interface problem on an <i>hp</i> -adaptively refined mesh. The red line denotes the error tolerance defining one of the stopping criteria, which can change as the area of narrow band changes. . . .	170
6.28	Configuration of mesh and interface position over time for the translating circular interface evolution problem, on an <i>hp</i> -adaptively refined mesh. The thick black line denotes the computed interface position.	171
6.29	Error against the square root of the number of degrees of freedom (ndof), for the initialisation of the shearing circular interface problem, on an <i>hp</i> -adaptively refined mesh.	172
6.30	Final computed mesh configuration for the initialisation of the shearing circular interface problem on an <i>hp</i> -adaptively refined mesh where the colour of the element denotes the polynomial order of that element. The thick black line denotes the computed interface position.	173
6.31	Error and area ratio over pseudotime for the shearing circular interface problem on an <i>hp</i> -adaptively refined mesh. The red line denotes the error tolerance defining one of the stopping criteria, as this varies with the area of narrow band.	174
6.32	Configuration of mesh and interface position over time for the shearing circular interface evolution problem, on an <i>hp</i> -adaptively refined mesh, in the interval $t_e = (0, 0.01)$. The thick black line denotes the position of the discrete interface. .	176
6.33	Error over time, for the shearing circular interface problem, on an <i>hp</i> -adaptively refined mesh in the time period $t_e = (0, 0.025)$. The red line denotes the error tolerance defining one of the stopping criteria, which can change as the area of narrow band changes.	177
6.34	Configuration of mesh and interface position over time for the shearing circular interface evolution problem, on an <i>hp</i> -adaptively refined mesh. The thick black line denotes the position of the discrete interface.	178
6.35	Error against the square root of the number of degrees of freedom (ndof), for the initialisation of the merging circular interfaces problem, on an <i>hp</i> -adaptively refined mesh.	179
6.36	Final computed mesh configuration for the initialisation of the merging circular interfaces problem on an <i>hp</i> -adaptively refined mesh where the colour of the element denotes the polynomial order of that element. The thick black line denotes the computed interface position.	180
6.37	Error over pseudotime for the merging circular interfaces problem on an <i>hp</i> -adaptively refined mesh. The red line denotes the error tolerance defining one of the stopping criteria, which can change as the area of narrow band changes. .	180
6.38	Configuration of mesh and interface position over time for the merging circular interfaces evolution problem, on an <i>hp</i> -adaptively refined mesh. The thick black line denotes the computed position of the interface.	181
7.1	Flow chart demonstrating the computational procedure for level set based topology optimisation based on a shape sensitivity analysis.	189

7.2	Domain configuration for the cantilever beam topology optimisation problem, showing boundary and loading conditions.	198
7.3	Initial topology for the minimum compliance design of a cantilever beam.	198
7.4	Compliance, volume ratio and Lagrange multiplier estimates during the optimisation for the minimum compliance design of a cantilever beam subject to a volume constraint.	200
7.5	Varying topology over pseudotime for the optimising cantilever beam.	201
7.6	Converged solution to the minimum compliance design of a cantilever beam subject a volume constraint.	202
7.7	Solution to the minimum compliance design of a cantilever beam subject to a volume constraint after 310 iterations.	202
7.8	Narrow banded mesh at convergence for the minimum compliance design of a cantilever beam subject to a volume constraint. The thick black line denotes the position of the level set interface.	204
7.9	Narrow banded mesh after 310 iterations whilst computing the minimum compliance design of a cantilever beam subject to a volume constraint. The thick black line denotes the position of the level set interface.	205

List of Tables

4.1	Critical time steps with associated errors and computational expense for the three discretisations of the Parabolic Reinitialisation method, as well as the Elliptic Reinitialisation method, when solving the circular interface reinitialisation problem. For the Symmetric Interior Penalty Discontinuous Galerkin (SIPG)-Implicit Euler (IE) formulation of the Parabolic method, Δt_r was set to 1000.	90
5.1	General form of the Butcher tableau	100
B.1	Butcher Tableau for Heun's Method (RK2)	211
B.2	Butcher Tableau for Heun's 3 rd Order Method (RK3)	211
B.3	Butcher Tableau for the classical 4 th Order Method (RK4)	212
B.4	Butcher Tableau for a 5 th order RK method (RK5) [200]	212
B.5	Butcher Tableau for a 7 stage 6 th Order Runge-Kutta Method (RK6) [200] . . .	212
B.6	Butcher Tableau for a 9 stage 7 th Order Runge-Kutta Method (RK7) [200] . . .	212
B.7	Butcher Tableau for an 11 stage 8 th Order Runge-Kutta Method (RK8) [200] . .	213

List of Algorithms

1	Anderson acceleration algorithm for level set reinitialisation or extrapolation. . . .	106
2	Narrow band level set evolution algorithm for problems on Cartesian meshes of elements of uniform polynomial order.	108
3	Mesh refinement algorithm	133
4	<i>hp</i> -adaptive narrow banded level set evolution algorithm	159
5	<i>hp</i> -adaptive narrow banded level set evolution algorithm for minimum compliance problems.	196

Acronyms

AESO	Additive Evolutionary Structural Optimisation
BEM	Boundary Element Method
BESO	Bidirectional Evolutionary Structural Optimisation
BO	Baumann-Oden
CFL	Courant-Friedrichs-Lewy
CG	Continuous Galerkin
DG	Discontinuous Galerkin
DRLSE	Distance Regularised Level Set Evolution
EE	Explicit Euler
ENO	Essentially Non-Oscillatory
ESO	Evolutionary Structural Optimisation
FD	Finite Difference
FE	Finite Element
FV	Finite Volume
GSG	growing-shrinking-growing
IBP	Integration By Parts
IE	Implicit Euler
IP	Interior Penalty
IIPG	Incomplete Interior Penalty Discontinuous Galerkin
LDG	Local Discontinuous Galerkin
LF	Lax-Friedrichs
LHS	Left Hand Side
MUSCL	Monotone Upstream-Centered Scheme for Conservation Laws
NIPG	Nonsymmetric Interior Penalty Discontinuous Galerkin
ndof	number of degrees of freedom

ODE	Ordinary Differential Equation
PDE	Partial Differential Equation
RHS	Right Hand Side
RK	Runge-Kutta
SI	Semi-Implicit
SIMP	Solid Isotropic Microstructure with Penalisation
SIPG	Symmetric Interior Penalty Discontinuous Galerkin
SSP	Strong Stability Preserving
STDG	Space-Time Discontinuous Galerkin
TVB	Total Variation Bounded
TVD	Total Variation Diminishing
UNO	Uniformly Non-Oscillatory
WENO	Weighted Essentially Non-Oscillatory
XFEM	eXtended Finite Element Method

Nomenclature

Greek

$\{\alpha_i\}_{i=0}^{N_s}$	Runge-Kutta weights i.e. Butcher Tableau Coefficients, see Table 5.1.
$\{\beta'\}_{i=0}^{N_I}$	Divergence free basis functions used to generate a new quadrature rule for integrating over immersed implicit boundaries, see Appendix A.
$\Gamma(\phi)$	Level set interface, that is the zero-isocontour of the level set function, ϕ , see Equation (3.1).
γ_A	Penalty parameter associated with enforcing a constraint on the allowed amount of volume in a structure undergoing a minimum compliance optimisation, see Equation (7.27).
γ_D	Penalty parameter associated with enforcing a Dirichlet boundary condition, see Equation (4.28).
γ_S	Penalty parameter associated with enforcing a constraint to ensure signed distance-ness of a level set function after evolution, see Equation (4.5).
Δt_e	Time step associated with level set evolution.
Δt_r	Time step associated with level set reinitialisation.
ε	2D engineering strain vector, that is $\varepsilon = \{\varepsilon_{xx}, \varepsilon_{yy}, 2\varepsilon_{xy}\}$, see Equation (7.15).
ζ	Basis functions associated with the Lagrange multiplier space, \mathfrak{L} .
$\theta(u)$	Heaviside function, $\theta(u) = 1$ if $u > 0$ and $\theta(u) = 0$ otherwise.
Θ_τ	Analyticity (smoothness) estimate of solution variable on the element, τ , see Section 6.3.2.
ϑ	Generic variable used to describe an angle.
κ	Level set curvature.
Λ	Lagrange multiplier matrix, see Equation (4.51).
λ_A	Lagrange multiplier associated with enforcing a constraint on the allowed amount of volume in a structure undergoing a minimum compliance optimisation, see Equation (7.27).
λ_D	Lagrange multiplier associated with enforcing a Dirichlet boundary condition, see Equation (4.29).
λ_p	Lagrange multiplier associated with enforcing the linear elasticity solution on a structure undergoing a minimum compliance optimisation, see Equation (7.21).
μ	Discontinuity penalisation parameter, see Equation (2.18).
ν	Poisson's ratio.
$\{\xi_i\}_{i=0}^{N_\xi}$	Position of the integration points used in Müller's method, see Equation (4.27).
$\phi(\mathbf{x}, t_e)$	Level set function, solution variable to the level set evolution equation.
ϕ_D	Prescribed value of the level set function on a Dirichlet boundary.

ϕ_h^n	The discrete solution of the level set evolution equation at the n^{th} time step (a distinction is made here in that the superscript n will always be used to refer to evolution as opposed to m which will refer to a solution at the m^{th} time step to a reinitialisation equation).
$\tilde{\phi}(\mathbf{x}, t_r)$	Level set function, solution variable to a level set reinitialisation equation.
$\tilde{\phi}_h^m$	The discrete solution of a level set reinitialisation equation at the m^{th} time step (a distinction is made here in that the superscript m will always be used to refer to reinitialisation as opposed to n which will refer to a solution at the n^{th} time step to the evolution equation).
$\tilde{\phi}_h^{m,k}$	The discrete solution of a level set reinitialisation equation at the k^{th} iteration of a quasi-Newton scheme, during the m^{th} time step.
$\overline{\nabla\phi}_\tau$	The average value of the gradient of the level set function at the integration points in the elements which neighbour the given element, τ . This value can be used to generate the value of the level set function on an element which is outside the narrow band, see Section 5.2.1.
$\boldsymbol{\sigma}$	2D stress vector with constituents, $\boldsymbol{\sigma} = \{\sigma_{xx}, \sigma_{yy}, \sigma_{xy}\}$, see Equations (7.10) and (7.14).
$\hat{\tau}$	Reference element.
$\partial\tau$	Boundary of an element, τ .
τ	Element.
$\{\chi_i\}_{i=0}^{N_\chi}$	Set of constraints for an associated generic optimisation problem.
Ω	Computational domain.
ω	Direction of directional derivative, see Equation (7.6).
Ω^{ϕ^+}	The part of the computational domain, Ω , over which the value of the level set function is positive (and therefore is full of material), this can be written equivalently as $\Omega^{\phi^+} = \Omega \setminus D$.
$\partial\Omega$	Boundary of computational domain, Ω .
$\partial\Omega_-$	Inflow portion of boundary of computational domain, $\partial\Omega$.
$\partial\Omega_{\text{sym}}$	Portion of boundary of computational domain, $\partial\Omega$, over which a symmetry boundary condition is to be enforced.
$\partial\Omega_D$	Portion of boundary of computational domain, $\partial\Omega$, over which a Dirichlet boundary condition is to be enforced.
$\partial\Omega_N$	Portion of boundary of computational domain, $\partial\Omega$, over which a Neumann boundary condition is to be enforced.

Roman

A	Area (or volume for corresponding 3D problem) of material in a structure undergoing minimum compliance optimisation.
A_{req}	Required area (or volume for corresponding 3D problem) of material in a structure undergoing minimum compliance optimisation, such that the solution is admissible.
A_{NB}	Area (or volume for corresponding 3D problem) of the problem domain which is inside the level set narrow band.
\mathbf{b}	Advection velocity vector which drives the level set evolution equation. This can be decomposed into magnitude and direction as follows, $\mathbf{b} = b\mathbf{n}_\phi$.
b	Magnitude of the advection velocity vector, \mathbf{b} .
B_{adv}	Bilinear form associated with a generic advection problem discretised using the discontinuous Galerkin method, see Equation (2.38).

B_{dif}	Bilinear form associated with a generic diffusion problem discretised using the symmetric interior penalty discontinuous Galerkin method, see Equation (2.26).
B_{ER}	Bilinear form associated with the Elliptic Reinitialisation equations discretised using the symmetric interior penalty discontinuous Galerkin method, see Equation (4.21).
B_{LE}	Bilinear form associated with the linear elasticity equations discretised using the symmetric interior penalty discontinuous Galerkin method, see Equation (7.12).
B_{PR}	Bilinear form associated with the Parabolic Reinitialisation equations discretised using the symmetric interior penalty discontinuous Galerkin method, see Equation (4.67).
\mathcal{C}	Compliance functional, used to define the minimum compliance optimisation problem, see Equation (7.18).
$\{c_i\}_{i=0}^{N_s}$	Runge-Kutta nodes i.e. Butcher Tableau Coefficients, see Table 5.1.
C^0	The space of continuous functions.
C^1	The space of differentiable functions whose derivative is continuous.
C^∞	The space of infinitely differentiable functions.
D	The part of the computational domain, Ω , over which the value of the level set function is negative.
$d_n(\cdot)$	Diffusion functional associated with the corresponding potential functional P_n , where the subscript $n = 1, 2, 3$ correspond to different proposed diffusion functionals defined in equations (4.15), (4.41) and (4.43) respectively.
DF	Jacobian matrix associated with the quasi-Newton method.
\mathcal{E}_{ext}	Set of all exterior edges, $\mathcal{E}_{\text{ext}} = \bigcup e_{\text{ext}}$.
\mathcal{E}_D	Set of Dirichlet edges.
\mathcal{E}_N	Set of Neumann edges.
e_{ext}	Exterior edge, that is an element edge which has non-empty intersection with the domain boundary, $\partial\Omega$.
e_{int}	Interior edge, that is an element edge which has non-empty intersection with the edge of an adjacent element in a given partition.
E_{Int}	Interface error measure, which is the L^2 error in the solution along the level set interface, see Equation (4.57).
E_{DG}	Error in the Discontinuous Galerkin (DG) norm, see Equation (4.55).
E_{L^2}	Error in the L^2 norm, see Equation (4.53).
E_{L^∞}	Error in the L^∞ norm, see Equation (4.54).
E_{SD}	Signed distance error measure, which measures the deviation in the between the norm of gradient of the level set function and unity, see Equation (4.56).
f_τ	Affine mapping from reference element to mesh element.
F_{ER}	Force vector associated with the Elliptic Reinitialisation method, see Equation (4.52).
F_{E}	Force vector associated with the level set evolution equation, see Equation (5.10).
F_{NR}	Residual vector associated with the Parabolic Reinitialisation equation solved using a quasi-Newton method, see Equation (4.82).
\mathbf{g}_N	Tractions prescribed on the Neumann edges of a linear elasticity problem, see Equation (7.10).
G_{NR}	Residual vector associated with the Parabolic Reinitialisation equation solved using a quasi-Newton method, see Equation (4.83).
h	The edge length associated with an element.
H^1	The Sobolev space containing the subset of functions, ψ , in L^2 , by which both the functions, ψ , and their corresponding weak derivatives, ψ' , are square integrable.

H^2	The Sobolev space containing the subset of functions, ψ , in L^2 , by which the functions, ψ , and their corresponding first two weak derivatives, ψ' and ψ'' , are square integrable.
h_τ	The edge length associated with the given element, τ .
J_{adv}	Linear form associated with the with a generic advection problem discretised using the discontinuous Galerkin method, see Equation (2.39).
J_{dif}	Linear form associated with the with a generic diffusion problem discretised using the symmetric interior penalty discontinuous Galerkin method, see Equation (2.27).
J_{LE}	Linear form associated with the with the linear elasticity equations discretised using the symmetric interior penalty discontinuous Galerkin method, see Equation (7.13).
$J_{\text{ER},n}$	Linear form associated with the Elliptic Reinitialisation discretised using the symmetric interior penalty discontinuous Galerkin method, see Equation (4.22). The subscript $n = 1, 2, 3$ correspond to different proposed diffusion functionals defined in equations (4.15), (4.41) and (4.43) respectively.
K_{ER}	Stiffness matrix associated with Elliptic Reinitialisation equations, see Equation (4.50).
K_{PR}	Stiffness matrix associated with Parabolic Reinitialisation equations, see Equation (4.71).
K_{P}	Penalty matrix associated with the level set evolution equations, see Equation (5.9).
\mathcal{L}	Lagrangian of an optimisation problem, see Equation (7.3).
\mathfrak{L}	Lagrange multiplier space, see Equation (4.34).
$\{\ell_i\}_{i=0}^\infty$	Legendre polynomials.
L^1	The space of absolutely integrable functions.
L^2	The space of square integrable functions.
l^2	The space of square summable sequences.
L^∞	The space of all essentially bounded functions.
l^∞	The space of bounded sequences.
M	Mass matrix, see Equation (4.70).
$\hat{\mathbf{n}}$	Unit outward normal to the edge of an element.
\mathbf{n}_Γ	Unit outward normal to the level set interface, equivalent to $\mathbf{n}_\phi = \frac{\nabla\phi(\mathbf{x})}{ \nabla\phi(\mathbf{x}) }$ for $\mathbf{x} \in \Gamma(\phi)$.
\mathbf{n}_ϕ	Unit outward normal to the level set function, equivalent to $\mathbf{n}_\phi = \frac{\nabla\phi}{ \nabla\phi }$.
N_h	Number of solution variables (degrees of freedom) per element.
N_I	Number of basis functions per element (specific to moment fitting equations and Müller's method, see section 4.3.2).
N_s	Number of Runge-Kutta stages, see Section 5.1.2.
N_ξ	Number of integration points per element (specific to Müller's method, see Section 4.3.2).
N_{LM}	Number of Lagrange multipliers per element.
\mathbf{p}	Vector containing order of interpolation space associated with each element in the partition.
p	The order of an interpolation space.
p_τ	The order of the interpolation space associated with the given element, τ .
p_{RK}	The order of the Runge-Kutta discretisation.
P_n	Potential Functional driving the minimisation based reinitialisation methods, where the subscript $n = 1, 2, 3$ correspond to different proposed potential functionals defined in equations (4.11), (4.40) and (4.42) respectively.

$[q_{ij}]$	Runge-Kutta matrix i.e. Butcher Tableau Coefficients, see Table 5.1.
\mathcal{Q}_{p_τ}	The space of polynomials of degree no more than p_τ in each coordinate direction.
\mathbb{R}^d	d -dimensional space of real numbers.
r	Radius.
R_n	Residual Functional driving the minimisation based reinitialisation methods, where the subscript $n = 1, 2, 3$ correspond to a residual based on different potential functionals, p_n , defined in equations (4.11), (4.40) and (4.42) respectively.
\mathcal{S}	The set of elements being currently operated on, see Algorithm 3.
$S(\mathcal{T})$	The set of internal edges on the partition, \mathcal{T} , also referred to as the skeleton of the mesh.
\mathcal{T}	Partition of a domain (mesh).
\mathcal{T}_Γ	Subset of the mesh, containing only the elements which are cut by the level set interface, see Equation (4.32).
\mathcal{T}_{NB}	Subset of the mesh, containing only the elements which form a narrow band around the level set interface, see Equation (5.12).
\mathcal{T}_{oNB}	Subset of the mesh, containing elements which are outside of the narrow band subset, but which have a neighbouring element which is inside the narrow band subset, see Equation (5.13). This subset therefore consists of one layer of elements surrounding the narrow band.
\mathcal{T}_T	Subset of the mesh, containing elements which at the previous updating of the narrow band subset were not inside \mathcal{T}_{NB} , but now are after the current update, see Section 5.2.1.
T	Final time.
t	Temporal variable, pseudotime.
t_e	Pseudotime relating to the level set evolution problem.
t_r	Pseudotime relating to the level set reinitialisation problem.
\mathbf{u}	Displacement, solution variable to the linear elasticity equations, (7.10), the bold print denotes that this value is vector valued consisting in this case of a horizontal and vertical component, i.e. $\{u_x, u_y\}$.
\mathbf{u}_h	Discrete displacement, solution variable to the discretised linear elasticity equations, (7.11), the bold print denotes that this value is vector valued, discreteness is denoted by the subscript h .
\mathcal{U}	Set of admissible solutions to a generic shape optimisation problem, see Equation (7.1).
u	Generic continuous solution variable to a partial differential equation.
u_+	Value of solution variable u on an outflow boundary.
u_-	Value of solution variable u on an inflow boundary.
u_D	Prescribed value of solution variable u on the Dirichlet boundary.
u_h	Discrete representation of the generic continuous variable u , discreteness is denoted by the subscript h .
V	Generic variable denoting an infinite dimensional functional space.
v	Generic variable denoting a basis function associated with the generic functional space, V .
V_h	Generic variable denoting a finite dimensional approximation of the functional space, V .
v_h	Generic variable denoting a basis function associated with the finite dimensional functional space, V_h .

V_p	The hp -version of the discontinuous Galerkin finite element space, see Equation (2.2), This is a scalar-valued space.
V_p^2	The hp -version of the discontinuous Galerkin finite element space, the exponent denotes that the space is vector-valued with dimensionality 2, the space is defined in Equation (7.17).
w	Weights associated with the quadrature rule computed using Müller’s method, see Equation (4.27).
\mathbf{x}	Spatial variable, $\mathbf{x} = \{x, y\}$.
Y	Young’s modulus
Y_{material}	Young’s modulus of the part of the domain filled with material when using an ersatz material approach to model empty regions of the domain, see Section 7.2.
Y_{void}	Young’s modulus of the part of the domain filled with ‘emptiness’ when using an ersatz material approach to model empty regions of the domain, see Section 7.2.
z	Decay rate of the penalty parameter used in the augmented Lagrangian approach to enforce a constraint on the volume of material in an admissible design in a structural optimisation problem, see Equation (7.28).

Mathematical

$(a, b)_A$	Denotes $\int_A a \cdot b \, d\mathbf{x}$.
$\langle a, b \rangle_A$	Denotes $\int_A a \cdot b \, ds$.
$[\cdot]$	Jump operator, see Equation (2.11).
$\{\{\cdot\}\}$	Average operator, see Equation (2.11).
$\text{dist}(a, A)$	Distance function, returns the minimum distance between the point a and the set of points A .
$\text{sign}(a)$	Signum function, returns the sign (\pm) of the argument a .
$\text{span}(A)$	Span; if $A \subset B$, $\text{span}(A)$ is the smallest subspace of B which contains A .
\emptyset	Empty set.
$\mathbf{1}_A(a)$	Indicator function, returns 1 if $a \in A$ and zero otherwise.
$A \setminus B$	The exclusion operator, can be read as ‘ A but not B ’.

Declaration

The work in this thesis is based on research carried out in the Computational Mechanics Group, Department of Engineering, Durham University. No part of this report has been submitted elsewhere for any other degree or qualification and it is all my own work unless referenced to the contrary in the text.

Parts of this work have been published in the following:

Conferences

ACME 24 (2016) - Cardiff

Oral Presentation, ‘Topology optimisation using level set methods and the discontinuous Galerkin Method’ Paper submitted to proceedings.

UKACM 25 (2017) - Birmingham

Oral Presentation, ‘Level set based topology optimisation using high-order reinitialisation methods and the discontinuous Galerkin method’ Paper submitted to proceedings.

ESCO 6 (2018) - Pilsen, Czech Republic

Oral Presentation, ‘A high-order elliptic PDE based level set reinitialisation method using a discontinuous Galerkin discretisation with applications to topology optimisation’ Paper submitted to proceedings.

ECCM-ECFD 6/7 (2018) - Glasgow

Oral Presentation, ‘A high-order elliptic PDE based level set reinitialisation method using a discontinuous Galerkin discretisation’ Paper submitted to proceedings.

UKACM 27 (2019) - London

Oral Presentation, ‘An *hp*-adaptive discontinuous Galerkin level set method’ Paper submitted to proceedings.

Journal Articles

T. Adams, S. Giani, and W. M. Coombs. “A high-order elliptic PDE based level set reinitialisation method using a discontinuous Galerkin discretisation.” *Journal of Computational Physics* 379 (2019): 373-391.

T. Adams, N. McLeish, S. Giani, and W. M. Coombs. “A parabolic level set reinitialisation method using a discontinuous Galerkin discretisation.” *Computers & Mathematics with Applications* (2019).

Copyright © 2020 by Thomas Adams.

“The copyright of this thesis rests with the author. No quotations from it should be published without the authors prior written consent and information derived from it should be acknowledged.”

Acknowledgements

First and foremost, I would like to express deep gratitude towards my doctoral supervisors Dr. Stefano Giani, and Dr. William M. Coombs for their guidance, support and unwavering patience throughout this PhD. I would like to thank Professor Charles Augarde, Professor Graham Coates and Dr. Mohammed Seaid for their time, guidance and constructive criticism with regards to the internal PhD progression review process. I would like to express special thanks towards my friend and colleague Dr. Robert Bird, for our many edifying discussions over the past four years. And finally, I would like to acknowledge that this PhD was supported by the Engineering and Physical Sciences Research Council (grant EP/M507854/1).

Thomas Adams
Durham, March 2020

Chapter 1

Introduction

1.1 Context and overview

Topology optimisation is the most general form of structural optimisation and is concerned with finding the boundary of a given problem domain which can be considered optimal, meaning that it minimises a chosen objective functional whilst satisfying a set of constraints. Such an idea is, for obvious reasons, of great interest to scientists and engineers, but especially nowadays as the prevailing sentiment of our times demand structures and components to be designed with great care taken to increase efficiency and reduce waste. In such a scenario, the ability to work backwards, to start with goals such as these in mind, and then by means independent of a designer compute a design which optimally satisfies those chosen goals, not only can expedite the design process, but may be necessary to lead to the possibly unintuitive designs required to adequately solve such difficult problems. This has led to a growing wealth of research in the area of topology optimisation, particularly over the past 30 years, and with it the development of a family of methods which can be used to solve problems of this type.

One of the first major groups of topology optimisation methods to be developed are known as hard-kill methods [1]. Hard-kill methods are called as such because, after partitioning the problem domain, each element in the mesh, can either be considered as being full of material or empty. The distribution of material over the domain then, can evolve iteratively through the mesh based on various heuristics which define each of the specific methods in the group. In this way the computed boundary, that is the solution to the topology optimisation problem, necessarily must align with the given partition. A similar but more sophisticated group of methods are known as density based topology optimisation methods [2]. Density based methods work by associating a density with each element in the mesh, which becomes the design variable to be solved for. In this way each element is either full of material, empty or contains a material of intermediate density. One of the most popular topology optimisation methods, which found significant use in industry, belongs to the group of density based methods, and is known as the Solid Isotropic Microstructure with Penalisation (SIMP) method [3]. One of the main issues with methods such as these is that they do not allow for accurate or geometrically complex solutions to be found, beyond the density of the mesh. Such criticisms led researchers to develop a group of methods known as boundary variation methods, initial attempts at which represented the

boundary explicitly using polynomial and spline representations [4–6]. These explicit methods however never really found a footing as the optimisation process means moving boundaries and thus remeshing, which is problematic both in terms of computational expense and stability due to mesh distortion.

More recently, a numerical technique known as the level set method, [7], has begun to be employed more widely as a constituent part of various boundary variation topology optimisation methods, see [8, 9] for initial attempts in this regard. The level set method is a numerical technique for representing and tracking evolving interfaces, and can be used as such to track the evolving boundaries of a structure undergoing optimisation. The level set method, however, is an implicit method, which means that the position of the boundaries which are represented by the method are not known explicitly, but instead are implied by the intersection of a hyperplane with a function known as a level set function (and the boundary is thus a level set of the level set function). By virtue of being an implicit method, the level set method is much more computationally efficient than explicit methods, and has advantages over hard-kill and density based methods also, as complex geometries can be captured by simple Cartesian meshes and complex topological changes can be handled with ease. This has led to the growing interest in the level set method for use in this context. Level set methods however, outside the context of topology optimisation, have some known issues. For example, the methods can be prone to numerical instability and the level of accuracy for a given mesh density can often be low; this can be seen in the recent review by Gibou [10] which shows that active areas of research include level set regularisation techniques, to improve stability, as well as methods for reducing mass loss (a phenomenon by which the mass or volume contained by a given level set interface is not maintained over time). Thus whilst such a methodology shows promise, it seems to be that there is room for improvements to be made particularly with respect to the evolution of the optimising boundaries, where level set methods are used for topology optimisation.

A similar but less popular method which has also more recently been applied to boundary variation topology optimisation problems are phase field methods, initial attempts at which can be found in [11]. As with the level set method, the distribution of material in a domain using the phase field method is implied by a function, in this case a phase field function. In the phase field paradigm however, this function is a smoothed Heaviside function and thus the boundary between the two phases has finite thickness. Optimal shapes can be computed using a phase field approach by evolving the phase field function (and thus the implied interface) through the solution of a fourth order Cahn-Hilliard equation which is often simplified into a coupled pair of second order Partial Differential Equations (PDEs). It has been suggested that one reason for the lack of popularity of phase field methods in this context include the inability to precisely determine the position of the interface in the transition region between regions filled with material and which are empty [12], and similarly despite the implicit nature of the method a common criticism of the methods are the high computational costs associated with solving the equations driving the optimisation using such an approach [12–14].

As topology optimisation is a computational technique specific to engineers, it is no surprise that the Finite Element (FE) Method, is a major constituent of many of the proposed method-

ologies for solving topology optimisation problems. Here the FE method finds use not only in discretising the physical equations defining the behaviour of the structures to be optimised but often also, as is implied above, as an integral part of the optimisation process itself. It is, of course, advantageous that a single discretisation method be used for all of the equations in a given methodology. Traditionally however, level set methods have been discretised using Finite Difference (FD) and Finite Volume (FV) type methods as the equations underlying the evolution of the level set function are hyperbolic, a class of problem with which a standard FE discretisation is understood to perform poorly [15].

Given the issues with the level set method described above recent research activity in the area of level set based topology optimisation has been directed towards the investigation of other discretisation techniques such as the eXtended Finite Element Method (XFEM) [16–18] and the Boundary Element Method (BEM) [19, 20], as well as other means for evolving the boundaries defined by a level set function such as using mathematical programming techniques [21]. One interesting family of methods which seems particularly well suited for this role are Discontinuous Galerkin (DG) methods. DG methods are a class of non-conforming FE method, which allow for discontinuous discretisations by formulating problems locally on each element in a mesh, and facilitating interelement communication with the use of flux terms across each element face, much like FV methods. In this way, by combining parts of the traditional FE and FV methods, DG methods can be considered to have advantages over both. With specific regards to level set methods these advantages include; their ability to deal with directional phenomena, their nonlinear stability, their formal high-order accuracy, and more generally due to their compactness the methods’ are advantageous due to their high level of parallelisability and the ease with which one can include *hp*-adaptivity.

This presents the context for the original goal for the work to be presented in this thesis. That goal was to exploit some of the known advantages of the discontinuous Galerkin method in the discretisation of the partial differential equations which drive a level set based topology optimisation methodology. In particular, the high levels of parallelisability of DG methods, as well as, the ease with which the methods can deal with *hp*-adaptive meshes, to develop a methodology which would be able to be applied to some interesting set of example problems. As is often the case with research, however, the aims of the research change with time limits and with the knowledge gained by engaging in the research.

The first stage of the research was to apply a DG discretisation, to the relevant level set equations and demonstrate that the resultant method satisfied the formally arbitrary high-order accuracy expected. One consideration which needed to be taken into account at this stage was that there is a known numerical issue with the level set method known as the tentpole phenomenon [22], by which during the evolution of a level set function, the level sets would drift resulting in steep and flat regions. This is problematic as it can be shown that large variations in the gradient of the level set function will lead eventually to numerical instability and a breakdown in the solution. It is also understood that even if this problem doesn’t cause a breakdown of the solution, avoiding this issue is beneficial in terms of accuracy of the computed solution [23]. In order to deal with issues generated by the variation in the gradient of level

set function, researchers have developed numerous regularisation techniques, often referred to as reinitialisation methods, in order to either ensure that the gradient remains constant (usually equal to unity) after each level set evolution step, or which can regenerate (or generate a new) level set function, with constant gradient, should the variation in the gradient become too large.

Whilst investigating these techniques for regularising the level set function it became apparent that simply translating an existing method into the DG paradigm would not be trivial given the desire that the developed method demonstrate high-order accuracy. For example, with reinitialisation the aim is to compute a level set function with more favourable numerical qualities, i.e. with less variation in the gradient of the level set function over the domain. However as an auxiliary method which occurs between iterations of the evolution of the level set interface, it is important that the position of the interface does not change during reinitialisation. This means that a Dirichlet boundary condition needs to be enforced on the level set interface, which is an implicit surface. The question of how to enforce such a boundary condition to an adequate level of accuracy was one which had not been answered in the literature. Other issues concerning singularities in the computed solution, one can imagine the level set function with constant gradient defining a circular interface taking the form of a cone for example, had likewise gone unaddressed. As such the development of level set reinitialisation techniques became a much more important part of the work than was first anticipated and a significant proportion of the research is directed towards endeavours to adequately solve the level set regularisation problem in the chosen DG paradigm. To this end, in this thesis two novel DG discretised PDE based level set reinitialisation methods are presented both of which demonstrate the desired high-order accuracy, as well as a number of other improvements over existing reinitialisation techniques.

Once the reinitialisation problem had a satisfactory solution, the next stage was to discretise a full level set evolution problem using DG, again with the aims of demonstrating high-order accuracy, utilising as necessary the developed reinitialisation techniques to ensure stability. In order to ensure accuracy, experimentation with various time stepping methods was required, eventually settling on a high-order Runge-Kutta (RK) temporal discretisation. Further, one contribution in this regard is the presentation of a simplified DG discretisation of the level set evolution equations with a novel flux, which can be afforded by the confidence one can have that at any given point the level set function will be a good approximation of a signed distance function due to the developed level set reinitialisation methods. This increases the efficiency of the evolution process, at the expense of reinitialisation. This is a good trade-off however, as the reinitialisation is a requirement anyway if one desires high-order accuracy and numerical stability.

Also explored at this stage are ideas concerning narrow band level set methods. Narrow banding is a technique by which the problem size for both the evolution and reinitialisation problems can be reduced from the entire mesh to a smaller set of elements in the region immediately surrounding the level set interface. This is allowed as the area of interest is the level set interface and any information in the far-field should not have a significant effect on the evolution of the interface. Narrow banding had already been useful when it comes to reinitialisation as one of its benefits is that it can remove singularities which develop in the far-field from the domain

which helps to improve stability. Thus, also explored at this stage of the research was a more critical approach concerning the effective use of narrow band level set methods. In the context of evolution problems, once the interface evolves far enough from its initial position the set of elements comprising the narrow band will change, however the solution on those elements will either have not been appropriately updated whilst those elements were outside or will not exist at all. This led to the development of another novelty presented in this thesis, a technique for extrapolating the level set function outwards from the narrow band to those elements outside the narrow band.

Once this had been achieved, the next stage of the research was an investigation into adaptive mesh refinement for the developed DG level set methodology. Several works have made use of adaptive mesh refinement when solving level set problems in order to improve accuracy and reduce costs, however, where adaptivity has been used it is often employed in a simplistic fashion based on geometric criteria [24–32], and almost never are both h and p adaptivity used in combination [33]. Given the adeptness of DG in this domain, it seemed natural to incorporate an hp -adaptive refinement strategy into the proposed DG based level set method. To this end, presented in this thesis is a novel hp -adaptive mesh refinement strategy, including a novel refinement criterion based once again on the work completed on level set reinitialisation. One potential issue which did arise at this stage, was that when the shape described by the level set interface was more complex, significantly greater computational resources were required to achieve a given level of accuracy. The reasons for this were investigated and it seems that firstly modelling interfaces of any shape to high levels of accuracy on the chosen finite element space using the level set method is inherently difficult and thus resource intensive in terms of memory. As the shape of the interface becomes more complex, however, this problem exaggerates and therefore becomes much more apparent. And secondly the chosen iterative method for the reinitialisation problem has a particularly poor convergence rate which means greater levels of real time are required to achieve a desired level of accuracy, which also increases with the increased complexity of the interface. Whilst some experimentation with other iterative solvers was performed during the research period, unfortunately no alternative stable method was found. As such it might be the case that a potential way of alleviating this issue could be the incorporation of high performance computing techniques as the increased efficiency given by the parallelisation for example may allow the iterative solver to achieve greater levels of accuracy for a given time period and also one may be able to more adequately deal with the memory requirements associated with the desired high levels of resolution.

The technical part of the thesis is concluded with a preliminary exploration of the proposed hp -adaptive DG discretised level set methodology in the context of which it was originally intended, topology optimisation. To this end, presented are details concerning a shape sensitivity analysis approach to optimisation, which can be effectively paired with the level set method [9], in that it can be used to compute an advection velocity field to drive the evolution of a level set interface to a position which minimises a given objective functional. This is combined with an augmented Lagrangian approach [34] for enforcing the constraints on the optimisation problem. The proposed methodology is then applied to a standard benchmark problem, that

is computing the the minimum compliance design of a linear elastic cantilever beam, under a volume constraint.

1.2 Thesis outline

The main body of this thesis consists of 6 chapters. Each of the chapters will consist of a similar form; the chapter will begin with a review of the relevant literature which will contextualise and explain the motivations for each of the chapters, this will be followed by a theory section, and then where applicable numerical examples. A more specific breakdown of the information contained within each chapter is detailed below.

Chapter 2: Discontinuous Galerkin methods

Chapter 2 begins with an overview of numerical methods for the spatial discretisation of PDEs, in particular presenting the history and development of DG and Interior Penalty (IP) methods since their inception in the 1970's. Next the preliminary mathematical nomenclature for DG methods is presented, which is requisite for the discussions which will follow for the remainder of the thesis. After this the literature related to DG methods as applied to the specific classes of equations found in this thesis, first-order advection and second-order quasilinear diffusion, are reviewed with attention paid to error analysis where it has been performed. Furthermore, a generic problem belonging to each of these classes is discretised using the relevant DG variant and the discretisation explained.

Chapter 3: Level set method

Chapter 3 introduces and presents the preliminary mathematical nomenclature for the level set method. In particular three aspects of the level set method are outlined; the formulation of a generic level set evolution problem; level set initialisation and reinitialisation; and narrow banded level set methods. Also presented in this chapter is a literature review concerning existing high-order discretisations of the level set evolution equation with a focus on DG methods where they have been applied.

Chapter 4: Level set reinitialisation

Chapter 4 presents the research completed in the area of level set reinitialisation. This begins with a literature review of the existing methods of level set reinitialisation, again focussing on DG discretisations wherever they occur. This is followed by the proposal of a novel reinitialisation method referred to as the *Elliptic Reinitialisation method*, the development of which is explained in detail and which is shown through experimentation to be of significantly higher accuracy than is capable with competitive methods. A second novelty is presented whereby through reformulation of the Elliptic Reinitialisation method, a *Parabolic Reinitialisation method* is also proposed. The two developed reinitialisation methods are compared and whilst both methods display similar levels of accuracy, the stability of the parabolic formulation is dependent on a problem dependent time step and as such the Elliptic Reinitialisation is decided to be the preferable method. Also presented in this chapter are discussions concerning the enforcement of boundary

conditions on immersed implicit level set interfaces, and the necessity of narrow banding for certain applications of the level set method.

Chapter 5: Level set evolution

Chapter 5 proposes a DG methodology for solving level set evolution problems. In particular the level set equation is simplified, which is made possible by frequent and accurate level set reinitialisation, and then discretised using a DG method using a novel flux. Narrow banding is discussed and with it a novel technique is presented for extrapolating the level set function to elements outside of the narrow band, which is required as the narrow band itself evolves to follow the evolving interface. Anderson acceleration is discussed as a method for improving the convergence of the fixed point iterative method used to solve the reinitialisation and extrapolation problems. The constituents of the methodology; the evolution, reinitialisation, and extrapolation equations and their associated technology are combined and used to solve a number of numerical examples.

Chapter 6: Level set method: adaptive mesh refinement

Chapter 6 proposes a novel strategy and criterion for hp -adaptive mesh refinement in the context of the DG discretised narrow banded level set methodology presented in Chapter 5. The chapter begins with a review of the literature concerning existing refinement strategies for level set methods. Then after presenting the proposed refinement strategy, a number of level set reinitialisation, and level set evolution example problems are solved this time on hp -adaptive meshes.

Chapter 7: Topology Optimisation

Chapter 7 presents a brief foray into topology optimisation using the DG discretised, hp -adaptive, narrow banded level set methodology formulated in the preceding chapters. A literature review is presented discussing a history of the methods used for topology optimisation, and thus where the level set methodology presented thus far would fit in such a context. The requisite theory for using a shape sensitivity approach to solving a minimum compliance problem for linear elastic structures with a constraint on the maximum allowed amount of material is presented which is combined with the proposed level set methodology before solving an example problem involving the design of a cantilever beam under an applied traction.

Chapter 8: Conclusions

Chapter 8 presents a summary of the ideas presented in the thesis, and highlights in particular the novel developments made during the research period. The chapter then proceeds to discuss issues encountered which are yet to be overcome, as well as other areas of interest which were ultimately beyond the scope of the research presented in this thesis given the time constraints of the research period, as parts of a larger discussion on suggested areas of future work.

Whilst mathematical notation will be introduced as and when necessary throughout the thesis, stated here are a number of conventions which will be consistent throughout the presented nomenclature. Vector valued functions and variables always use bold-face notation, for example,

the spatial variable can be denoted $\boldsymbol{x} = \{x, y\}$. Subscripts are used extensively as identifiers between variables and functions which denote the same general meaning but differ in specifics, for example, E_{L^2} and E_{DG} both denote errors with the subscript identifying the specific norm in which the error is computed. In the case that a variable needs both an identifier and an index as a subscript, these are separated by a comma in the order, identifier then index. It is noted that the comma subscript is not used in this thesis, as is sometimes found in the literature, to denote a partial derivative. Indices denoting time step or iteration are always superscripts, and use the symbol m , n or k . In the case of nested iterative methods, multiple superscripts will be used, again separated by a comma. Where algorithms are presented, functions and variables are often given descriptive names as opposed to symbols, in such a case these names are written in camel case and use the Latin Modern Typewriter font, for example, `errorHandle` is a variable containing a handle which points to the appropriate error to be computed. Furthermore, where names are used instead of symbols, operations such as multiplication will always be stated explicitly to avoid confusion.

Chapter 2

Discontinuous Galerkin Methods

2.1 Overview

Finite Element (FE) analysis is an almost ubiquitous numerical tool for approximating the solution to problems in engineering. The standard (i.e. most popular) FE method is known as the Continuous Galerkin (CG) FE method. For a problem with a solution which belongs to a function space, V , the key component of a Galerkin FE method is to look for a solution in a finite dimensional subspace of that space, V_h (where the subscript h refers to a discretisation parameter, often and in this case related to the size of the elements used to discretise the domain on which the problem is to be solved). The classical conforming CGFE method then, takes its name from the restrictions imposed on this approximation space, V_h , which are; that it is conforming i.e. $V_h \subset V$, and in particular that the functions which form a basis of the space are continuous throughout the problem domain (and therefore are continuous across element boundaries).

In 1973, Reed and Hill [35], designed an explicit non-conforming numerical method for solving the neutron transport problem in which there was no requirement on interelement continuity on the approximation space, V_h , for the first time in a finite element context. In their paper, Reed and Hill presented a piecewise continuous method where the problem is formulated locally on each element in the problem domain, and a flux term is used to pass information between adjacent elements. This was compared against a strictly continuous formulation, where it was demonstrated by experiment that the discontinuous method was superior both in terms of accuracy and stability. This was a significant result as traditionally FE methods had performed poorly when attempting to solve first-order hyperbolic Partial Differential Equations (PDEs), compared with other numerical methods of the time, and therefore had typically been ignored by the wider community for use in this context. In 1974, LeSaint and Raviart noted this significance, and produced an analysis of Reed and Hill's method [36] in which they showed that Reed and Hill's method was a generalisation of the CGFE method, and subsequently named Reed and Hill's method, the Discontinuous Galerkin (DG) method.

The original DG scheme of Reed and Hill, for solving a transport problem, was influenced by earlier works on solving transport problems, whereby, as it is known that advection is a directional phenomenon, the appropriate method of passing information between elements is

to make use of an upwind type flux. Diffusion however is a non-directional phenomenon, and as such non-conforming FE methods for diffusion type problems might therefore be built in a similar way to Reed and Hill’s method, except in this case by passing information across element faces using a central/average type flux. Around the same time that Reed and Hill published their article the methods of Nitsche [37] and Babuška [38] which allowed one to weakly impose Dirichlet boundary conditions, were being extended, much in this vein, to allow for analogous methods to be used to enforce interelement continuity. These methods are known as the Interior Penalty (IP) methods. Whilst it is slightly more difficult to pinpoint an original IP method, one early example was presented in the 1973 paper by Babuška and Zlámal [39], in which a penalty type method was used to weakly impose C^1 continuity for the fourth-order biharmonic equation. Other methods which used an approach more analogous to Nitsche’s method includes the 1977 article by Baker [40], which again imposed C^1 continuity on C^0 elements for fourth-order problems, the 1976 work by Douglas and Dupont [41] which penalised the jump in the normal derivative to enforce continuity for second-order elliptic and parabolic PDEs, the 1978 work by Wheeler [42] which included generalisations of the consistency, symmetry and penalty terms of Nitsche’s method across element edges to solve elliptic PDEs, and the 1979 PhD Thesis of Arnold [43] which presented and analysed a similar method to that of Wheeler for nonlinear elliptic and parabolic PDEs. The idea however, that IP methods were DG methods using a different type of flux wasn’t noticed until the 1990’s and as such the development of DG methods and IP methods continued independently in the interim.

After the initial attempts at using DG and IP methods in the 1970’s, further developments were relatively sparse in the following decade. One active area of research at the time however, was methods for solving nonlinear hyperbolic conservation laws. In 1982, Chavent and Solzano [44] published an article which attempted to extend the works of Reed and Hill and LeSaint and Raviart to solve problems of this type, however, one of the main issues with DG methods during this period, was the time discretisation. An implicit solver would require an expensive global nonlinear solve, whereas the first-order explicit Euler method which was used by Chavent and Solzano [44] suffers from a severe time step restriction. In the case of IP methods, Arnold [45] suggests that the reason for the lack of progress may have been to do with the difficulty in finding optimal penalty parameters, as well as the methods never being proven to have significant advantages over a classical CGFE approach.

In the same two decades however, largely driven by the desire to compute accurate solutions to nonlinear hyperbolic systems, there was a vast amount of research completed in the area of high resolution Finite Difference (FD) and Finite Volume (FV) schemes. An important starting point in this context was Godunov’s method [46], which was a first-order finite volume scheme, upon which many of the high resolution schemes were based. The main novel idea in Godunov’s method was to update the solution at each time step by solving exactly a Riemann problem at each cell interface and averaging the solution to these Riemann problems over the domain. In this way Godunov had extended the first-order upwind scheme of Courant, Isaacson and Rees [47] to nonlinear systems of hyperbolic conservation equations. One of the main issues with first-order methods such as Godunov’s method is that they tend to be very diffusive, and therefore

cause discontinuities, which might be physical, to smooth out over time. Higher-order methods such as the Lax-Wendroff scheme [48], however, whilst providing higher resolution in smooth parts of the solution introduce spurious oscillations near to local extrema and particularly in the regions where a solution is discontinuous. In fact Godunov in his article, [46], had proven that higher-order methods, could not both preserve monotonicity (i.e. not introduce oscillations) and also be higher than first-order accurate. The main issue to overcome then was the question of how to resolve discontinuities in a solution over relatively few cells, whilst providing high-order accuracy in smooth regions without introducing erroneous oscillations. Methods presented which aimed to overcome this issue began to be published in the 1970's, with the series of articles by Van Leer [49–53]. In these articles Van Leer introduced the Monotone Upstream-Centered Scheme for Conservation Laws (MUSCL) and with it the idea of flux/slope limiting, by which Godunov's scheme was extended to include adaptive higher-order (in this case linear) approximations to the solution at cell interfaces. The idea was that the higher-order fluxes would provide a higher resolution where the solution was smooth, and the flux could then be limited to prevent oscillations in areas where the solution was sharp. In this way the method, and later higher-order variants for example [54–56], were able to ensure that the total variation (a measure of the oscillation in a solution) was non-increasing (also known as Total Variation Diminishing (TVD)). A related advancement in this respect was introduced in 1981 by Roe [57], who noticed that much of the information gained by solving the set of exact Riemann problems was lost after the solution at cell interfaces was then averaged over the domain. Given that these Riemann problems, especially in the case of nonlinear systems of equations, could be expensive to solve, Roe considered that it may be possible to obtain good results by replacing the Riemann problem with a cheaper to compute approximation. In that initial paper by Roe and similar papers which followed, for example [58, 59], this was achieved by either approximating the Riemann states and then computing the physical flux, or by approximating directly a numerical flux. In 1987, it was shown [60] that schemes which were total variation non-increasing were at most first-order accurate due to the degenerated accuracy in the regions of local extrema (which may be smooth). The solution to this issue was the development of a suitable relaxation of the the TVD property. One approach in this vein, presented in a series of articles by Harten et al. [60, 61], replaced the TVD property by the more lenient Uniformly Non-Oscillatory (UNO) property which allowed the total variation to increase, but maintained that the number of local extrema must be non-increasing, and then eventually by the even more lenient Essentially Non-Oscillatory (ENO) property which allowed for the number of local extrema to increase as well as the total variation, however the increase in the total variation was bounded to be on the order of the cell size, h . A different approach developed by Shu [62], was to replace the TVD property with the Total Variation Bounded (TVB) property by which the total variation is bounded by a positive constant for all time. The schemes developed under the UNO, ENO and TVB frameworks were designed to use arbitrarily high-order interpolation spaces and therefore be able to maintain higher-order accuracy in smooth regions. However, as is noted in [63], high resolution finite volume schemes are not in general formally high-order accurate, in that even on smooth parts of a solution there can be a degradation of accuracy as a result of the coupling

between characteristic components for all but the simplest linear advection problems.

Despite the lack of research directly into DG schemes, by virtue of the discontinuous finite element spaces which can be used in the DG paradigm, information is passed between cells by defining a numerical flux, in much the same way as finite volume methods, and therefore the research on high resolution schemes was able to be naturally incorporated into the finite element framework through DG methods. Thus in the late 1980's the works of Van Leer on slope limiters [52] and Harten on TVD schemes [60] was able to be utilised in the context of DG allowing for the series of articles by Cockburn and Shu [64–68] which presented the Runge-Kutta (RK) DG method for multidimensional systems of nonlinear hyperbolic problems. Furthermore, by the 1990's with the developments in the area of DG methods for problems with non-negligible diffusive parts by authors such as Baumann and Oden [69] and Bassi and Rebay [70], the similarity between IP and DG methods was finally noticed, which then prompted a resurgence of research into IP methods for diffusive PDEs with aims to exploit what was becoming an apparent abundance of advantages of discretising problems using DG methods. Some of the advantages of DG discretisations over both traditional FE schemes and also the aforementioned high resolution FV schemes include; the methods' formal high-order accuracy, their high level of parallelisability, their ability to easily incorporate hp -adaptivity, their ability to deal with complex geometries, their nonlinear stability and their ability to deal with discontinuous solutions. For these reasons, this thesis proceeds in a similar fashion by attempting to extend the domain of DG discretisations further, into the context of level set based topology optimisation methods.

2.2 Mathematical preliminaries

In this section the nomenclature and mathematical preliminaries required for the discussion of a methodology discretised spatially using DG methods will be introduced. In this thesis, a family of partitions, \mathcal{T} , are considered of a domain, $\Omega \subset \mathbb{R}^2$, into disjoint quadrilateral elements, τ . Whilst the problems presented here are restricted to 2 dimensions, it should be noted that the method extends naturally to higher dimensions. Only considered in this work are families of partitions which are shape regular, that is, there exists a constant, C_{reg} , such that

$$\frac{\text{diam}(\tau)}{2r_\tau} \leq C_{\text{reg}}, \quad \forall \tau \in \mathcal{T}, \quad (2.1)$$

where $\text{diam}(\tau)$ denotes the size associated with an element, τ , which in this case is that of its minimum edge length, h_τ , and r_τ denotes the radius of the largest circle inscribed in the element, τ . Each element, τ , is the image of the reference square, $\hat{\tau} = (-1, 1)^2$, under an affine elemental mapping, $f_\tau : \hat{\tau} \rightarrow \tau$. The non-empty intersection, of positive measure, of two neighbouring elements, $\tau_+, \tau_- \in \mathcal{T}$, that is, $e_{\text{int}} = \partial\tau_+ \cap \partial\tau_- \neq \emptyset$, is known as an interior edge. The skeleton of the mesh is denoted, $S(\mathcal{T})$, and is defined as the set of all interior edges. Similarly, the non-empty intersection, of positive measure, of the boundary of an element, $\tau \in \mathcal{T}$, with the boundary of the domain, $\partial\Omega$, that is, $e_{\text{ext}} = \partial\tau \cap \partial\Omega \neq \emptyset$, is called a boundary edge. The set of all boundary edges is denoted, $\mathcal{E}_{\text{ext}}(\mathcal{T})$. The set of all element edges is denoted by, $\partial\mathcal{T} = \bigcup_{\tau \in \mathcal{T}} \partial\tau = S(\mathcal{T}) \cup \mathcal{E}_{\text{ext}}(\mathcal{T})$. Throughout this thesis, 1-irregularly h -refined meshes are allowed for, by which each element

can contain at most one hanging node per edge.

Each element $\tau \in \mathcal{T}$ has an associated polynomial degree, $p_\tau \geq 1$, which is stored in the vector, $\mathbf{p} = \{p_\tau : \tau \in \mathcal{T}\}$. Hence, for a given partition, \mathcal{T} , of Ω , with degree vector, \mathbf{p} , the hp -version of the DG finite element space is defined as

$$V_{\mathbf{p}}(\mathcal{T}) := \{v_h \in L^2(\Omega) : \forall \tau \in \mathcal{T}, v_h|_\tau \circ f_\tau \in \mathcal{Q}_{p_\tau}(\hat{\tau})\}, \quad (2.2)$$

where $\mathcal{Q}_{p_\tau}(\hat{\tau})$ denotes the space of tensor product polynomials on the reference square, $\hat{\tau}$, of degree no more than, p_τ , in each coordinate direction. One advantage of working in a DG paradigm is that there is a lot of flexibility in which set of functions are chosen to form a basis of the space. According to Di Pietro [71], two considerations which one might wish to take when deciding upon this set of basis functions are, orthogonality (choosing an orthogonal basis can help limit the growth of matrix conditioning with polynomial order) and hierarchism (the idea that a basis for a polynomial order p contains the bases for all polynomial degrees less than p , and which therefore allows for a very simple implementation of p -adaptivity). In this case, a further consideration was that the implementation was originally designed to deal with both CG and DG meshes (although for the purposes of this thesis only the DG parts are of concern), and thus it was preferable to choose a basis which could be used in either case. As such, the exact choice of basis functions used in this work are the hierarchical modal bases for a H^1 conforming approximation space on quadrilateral elements as defined in Chapter 2 of [72]. Other choices which form a basis of the space, such as Legendre functions, would likely introduce slight optimisations to the DG implementation in terms of matrix conditioning (due to their orthogonality) and simplicity of implementation, however, as it was deemed unlikely to have a significant effect upon the results of the numerical examples to be presented in this thesis, it was decided that it would unnecessary to make such a change after having already implemented the Lobatto shape functions. It should be noted that defining the interpolation space in such a way does allow one to define a separate order of interpolation in each coordinate direction on a given element, in this work however it will always be the case that the order will be chosen to be equal in each coordinate direction. Furthermore, it will be assumed that \mathbf{p} is of bounded local variation, that is for neighbouring elements $\tau_+, \tau_- \in \mathcal{T}$, there exists a constant, $\rho_p \geq 1$, independent of the mesh, \mathcal{T} , such that

$$\frac{1}{\rho_p} \leq \frac{p_{\tau_+}}{p_{\tau_-}} \leq \rho_p. \quad (2.3)$$

2.3 Literature review

In this thesis, DG spatial discretisations will be applied to two classes of problem; first-order advective PDEs which describe the evolution of the level set function, and second-order linear and quasilinear diffusion equations, which are used in the equations driving the topology optimisation problems (i.e. linear elasticity) and for the regularisation problems associated with the level set method. As such this section will present a review of the relevant literature, as well as an example discretisation, in both of these cases. For a much more complete (and relatively recent)

overview of DG methods the reader is referred to the text [73].

2.3.1 DG methods for diffusive PDEs

Where diffusive terms arise in this thesis, whether they form part of an elliptic or parabolic PDE, in general the diffusion will be non-monotone and nonlinear. That is the diffusive term will be of the form

$$\nabla \cdot (d(u)\nabla u), \quad (2.4)$$

for the solution variable, u , where $d(u)$ is a function defining the behaviour of the diffusion and will therefore be referred to as the diffusion function. As a significant portion of the literature concerns diffusion which is monotonic, here it is stated that monotonicity in this respect can be defined by a diffusion function which satisfies the following condition; for $d \in C(\Omega)$,

$$q_d(u_1 - u_2) \leq d(u_1)u_1 - d(u_2)u_2 \leq Q_d(u_1 - u_2), \quad \text{for } u_1 \geq u_2 \geq 0, \quad (2.5)$$

for some positive constants q_d and Q_d , which, again, is a condition which in general will not be satisfied by the diffusive terms in the equations to be studied in this thesis. The reader is referred to [74] for further details concerning monotonicity.

There are many flavours of DG method which can be employed to deal with equations with non-negligible diffusion, with the difference between each of these methods coming down to how the fluxes, which allows information to pass between neighbouring elements, are chosen. An analysis unifying many of the varieties of DG methods, in this way, used for discretising elliptic problems is presented in [45]. The remainder of this section will outline roughly the development of a number of these methods (with a focus on IP type methods) with reference to error analyses where they have been performed, and will thus conclude on the preferred method of discretisation of diffusive terms in this thesis. This will be followed by an appropriate example discretisation of a diffusion problem in the following section.

In 1982, Arnold [75], first analysed, what would now be known as a Symmetric Interior Penalty Discontinuous Galerkin (SIPG) semi-discretisation of nonlinear parabolic boundary value problems and demonstrated optimal order error estimates in the L^2 and energy norms, assuming a sufficiently large penalty. One issue with IP type DG methods of the time, such as those presented by Arnold [75], Baker [40] and Wheeler [42], was that convergence, stability and matrix conditioning were all dependent on the correct choice of the penalty parameter and furthermore, none of these methods were locally conservative. One attempt at solving these issues was presented by Rivière, Wheeler and Girault, [76], who developed the Nonsymmetric Interior Penalty Discontinuous Galerkin (NIPG) method. This is a nonsymmetric formulation of an IP method, based on the Baumann-Oden (BO) method, [69], and in particular in this work Rivière et al. presented a method to determine an appropriate choice for the penalty parameter in the general case. Rivière and Wheeler, [77], then applied NIPG, along with an implicit time scheme, to nonlinear parabolic problems and derived an optimal error estimate in the $L^2(H^1)$ norm and a suboptimal error estimate in the $L^\infty(L^2)$ norm.

For non-monotone quasilinear diffusion of the form, (2.4), SIPG and NIPG methods were analysed by Gudi and Pani in [78], who derive *a priori* error estimates in the broken H^1 norm

for both SIPG and NIPG, which are shown to be equivalent to NIPG as applied to the linear problem. Furthermore, optimal estimates are derived in the L^2 norm for both SIPG and superpenalised NIPG on uniform regular grids. This was extended to nonlinear parabolic equations with diffusion of the more general form, as stated in (2.6), by Song [79], who investigated the use of SIPG and NIPG spatial discretisations with an explicit Euler temporal discretisation for the full discretisation of nonlinear parabolic equations, and derived suboptimal error estimates in the $l^\infty(L^2)$ and $l^2(H^1)$ norms (where the lower case l^n denotes that the temporal norm is discrete, as the analysis is performed after the discretisation of the time derivative). Song then went on to perform the same analysis for nonlinear parabolic problems discretised with IPDG methods in space and implicit schemes in time [80], in this case deriving optimal *a priori* error estimates in the $l^\infty(L^2)$ and $l^2(H^1)$ norms.

One issue with the standard NIPG and SIPG schemes, is that they cannot be applied directly to the general quasilinear diffusion problem, i.e. diffusion of the form

$$\nabla \cdot \mathcal{D}(u, \nabla u). \quad (2.6)$$

The reason for this is that the choice of flux which results in an SIPG or NIPG scheme contains a stabilisation term which can become impossible to linearise in both the trial and the test functions, see [81] for details. In such a case, an Incomplete Interior Penalty Discontinuous Galerkin (IIPG) discretisation can be useful as was analysed by Dolejší [81]. Another solution to this issue, in the case of monotone quasilinear diffusion, i.e. where the diffusion coefficient satisfies the condition (2.5), was presented by Houston et al. [74], where an IPDG scheme was developed with a new stabilisation term which is linear in both the test and trial functions.

In this thesis it is the authors preference to use an SIPG spatial discretisation, [45]. There are a number of reasons why SIPG can be considered preferable to other DG discretisations of diffusion equations, which are outlined in the Conclusion of [82]. Highlights include the well established optimal rates of convergence in the L^2 norm for SIPG, which are not found for the nonsymmetric DG methods (which includes NIPG, IIPG and BO), as well as the increased efficiency in terms of the linear solve (compared with the nonsymmetric methods) and memory requirements (compared with the Local Discontinuous Galerkin (LDG) method, [83]).

2.3.1.1 DG discretisations for diffusive PDEs

The method of lines is used for all time varying PDEs to be considered in this thesis, which means that the spatial derivatives will be replaced by a discrete approximation, whilst leaving the time variable continuous. In this way, the given PDE, which is variable in space and time, can be approximated by a system of Ordinary Differential Equations (ODEs) to be solved only in the time dimension. Once the approximating system of ODEs has been formulated, this can be solved using any given integration routine for initial value ODEs, to find an approximate solution to the underlying PDE.

In all such cases a DG variant will be used for the spatial discretisation of any given PDE; for the reasons explained in Section 2.1. For all PDEs considered with a non-negligible diffusive term this DG variant will be the SIPG method. In order to demonstrate an SIPG discretisation,

one can consider a Laplace problem of the form

$$\begin{aligned}\nabla \cdot \nabla u(\mathbf{x}) &= 0, & \mathbf{x} \in \Omega, \\ u(\mathbf{x}) &= u_D, & \mathbf{x} \in \partial\Omega_D, \\ \nabla u(\mathbf{x}) \cdot \hat{\mathbf{n}} &= 0, & \mathbf{x} \in \partial\Omega_N,\end{aligned}\tag{2.7}$$

where the solution variable is denoted $u(\mathbf{x})$, which is a function of the spatial variable $\mathbf{x} = \{x, y\}$ (where it is unambiguous this dependency is dropped), and $\hat{\mathbf{n}}$ denotes the unit outward normal. Equation (2.7) has a heterogeneous Dirichlet boundary condition and a homogeneous Neumann boundary condition.

To discretise (2.7) using the SIPG method, one begins in the usual finite element way by multiplying through by a test function, $v \in V$, where V denotes the solution space, and then integrating by parts elementwise which gives

$$\sum_{\tau \in \mathcal{T}} (\nabla u, \nabla v)_\tau - \sum_{\partial\tau \in \partial\mathcal{T}} \langle \nabla u \cdot \hat{\mathbf{n}}, v \rangle_{\partial\tau} = 0, \quad \forall v \in V.\tag{2.8}$$

The notation $(a, b)_\Omega$ denotes $\int_\Omega ab \, d\mathbf{x}$, and $\langle a, b \rangle_{\partial\Omega}$ denotes $\int_{\partial\Omega} ab \, ds$. If one were to use a CG discretisation, then $v \in H^1(\Omega)$, that is the test function is from the space of functions which are weakly differentiable once. This implies the solution is continuous across internal edges, and therefore the integrals computed on the trace of each element sharing these edges would necessarily be equal and opposite. Thus at this stage in a corresponding CG derivation all of the surface integrals on internal edges would cancel out. In the DG paradigm, however, $v \in V = H^1(\Omega) + V_p(\Omega)$, see [84], which is a larger space which admits discontinuous solutions across internal element edges. It is unlikely in such a case that any of these surface integrals will be equivalent approaching a given edge from inside each element sharing that edge and thus all of these integrals must remain. One can decompose the set of surface integrals into the set of integrals over the internal edges, the set of Dirichlet edges which will be denoted, $\mathcal{E}_D(\mathcal{T}) \subset \mathcal{E}_{\text{ext}}(\mathcal{T})$, and the set of Neumann edges denoted, $\mathcal{E}_N(\mathcal{T}) \subset \mathcal{E}_{\text{ext}}(\mathcal{T})$, as follows

$$\begin{aligned}\sum_{\tau \in \mathcal{T}} (\nabla u, \nabla v)_\tau - \sum_{e_{\text{int}} \in \mathcal{S}(\mathcal{T})} \left[\langle \nabla u_{\tau_+} \cdot \hat{\mathbf{n}}_{\tau_+}, v_{\tau_+} \rangle_{e_{\text{int}}} + \langle \nabla u_{\tau_-} \cdot \hat{\mathbf{n}}_{\tau_-}, v_{\tau_-} \rangle_{e_{\text{int}}} \right] \\ - \sum_{e_{\text{ext}} \in \mathcal{E}_D(\mathcal{T})} \langle \nabla u \cdot \hat{\mathbf{n}}, v \rangle_{e_{\text{ext}}} - \sum_{e_{\text{ext}} \in \mathcal{E}_N(\mathcal{T})} \langle \nabla u \cdot \hat{\mathbf{n}}, v \rangle_{e_{\text{ext}}} = 0, \quad \forall v \in V,\end{aligned}\tag{2.9}$$

where $\hat{\mathbf{n}}_\tau$ denotes the unit normal of the edge pointing outwards from the element τ . By noting that $\hat{\mathbf{n}}_{\tau_+} = -\hat{\mathbf{n}}_{\tau_-}$, and by the analogue that $ac - bd = \frac{1}{2}(a+b)(c-d) + \frac{1}{2}(a-b)(c+d)$, Equation (2.9) can be rewritten

$$\begin{aligned}\sum_{\tau \in \mathcal{T}} (\nabla u, \nabla v)_\tau - \sum_{e_{\text{int}} \in \mathcal{S}(\mathcal{T})} \langle \{\{\nabla u\}\}, [v] \rangle_{e_{\text{int}}} - \sum_{e_{\text{int}} \in \mathcal{S}(\mathcal{T})} \langle [\nabla u], \{\{v\}\} \rangle_{e_{\text{int}}} \\ - \sum_{e_{\text{ext}} \in \mathcal{E}_D(\mathcal{T})} \langle \nabla u \cdot \hat{\mathbf{n}}, v \rangle_{e_{\text{ext}}} - \sum_{e_{\text{ext}} \in \mathcal{E}_N(\mathcal{T})} \langle \nabla u \cdot \hat{\mathbf{n}}, v \rangle_{e_{\text{ext}}} = 0, \quad \forall v \in V,\end{aligned}\tag{2.10}$$

where $[[\cdot]]$ and $\{\{\cdot\}\}$ denotes the jump and average operators respectively. For two neighbouring elements, $\tau_+, \tau_- \in \mathcal{T}$, which share an internal edge, the jump and average operators are as defined in [45] and are reproduced here as follows: for an arbitrary scalar valued function, ψ , and vector valued function, Ψ ,

$$[[\psi]] = (\psi_{\tau_+} - \psi_{\tau_-}) \hat{\mathbf{n}}_{\tau_+}, \quad (2.11)$$

$$[[\Psi]] = (\Psi_{\tau_+} + \Psi_{\tau_-}) \cdot \hat{\mathbf{n}}_{\tau_+}, \quad (2.12)$$

$$\{\{\psi\}\} = \frac{1}{2}(\psi_{\tau_+} + \psi_{\tau_-}), \quad (2.13)$$

$$\{\{\Psi\}\} = \frac{1}{2}(\Psi_{\tau_+} + \Psi_{\tau_-}). \quad (2.14)$$

For $u \in H^2(\Omega)$, $[[\nabla u]] = 0$, as it is continuous and therefore one can write

$$\begin{aligned} \sum_{\tau \in \mathcal{T}} (\nabla u, \nabla v)_\tau - \sum_{e_{\text{int}} \in \mathcal{S}(\mathcal{T})} \langle \{\{\nabla u\}\}, [v] \rangle_{e_{\text{int}}} - \sum_{e_{\text{ext}} \in \mathcal{E}_D(\mathcal{T})} \langle \nabla u \cdot \hat{\mathbf{n}}, v \rangle_{e_{\text{ext}}} \\ - \sum_{e_{\text{ext}} \in \mathcal{E}_N(\mathcal{T})} \langle \nabla u \cdot \hat{\mathbf{n}}, v \rangle_{e_{\text{ext}}} = 0, \quad \forall v \in V. \end{aligned} \quad (2.15)$$

The Dirichlet boundary condition in this case is not built into the solution space, V , and therefore needs to be imposed weakly. In this case this will be achieved using Nitsche's method [37]. Ignoring for a moment the internal and Neumann edges, Equation (2.15) consists of the sum of the volume integrals of the gradient term over each element in the partition and the sum of the surface integrals which describes the behaviour of the diffusion at the Dirichlet edges which arises from the Integration By Parts (IBP). These surface integrals on the Dirichlet edges must remain to ensure consistency with the weak solution by virtue of the solution space, V . If the bilinear form were to just consist of these two terms however, then it would not be coercive and there would not exist a unique solution to the problem (2.7). Nitsche's method rectifies this by the inclusion of a penalty term, which penalises the difference between the prescribed solution at the boundary with the computed solution, $(u - u_D)$. It is shown in Nitsche's original paper [37] that for a sufficiently large penalty parameter, μ , the bilinear form over the finite dimensional subspace, $V_{\mathbf{p}}(\mathcal{T})$ is in fact coercive. Whilst it is possible to stop at this stage, Nitsche also includes another surface integral over the Dirichlet edges which is consistent with the weak solution, but which symmetrises the bilinear parts of (2.15) by reversing the arguments of the nonsymmetric IBP surface integral. Whilst the symmetry of the bilinear form is not required it is often desirable as a symmetric bilinear form will result in a symmetric linear system, which is usually cheaper to solve than otherwise. With the inclusion of the two extra terms from Nitsche's method, the weak formulation with appropriate boundary conditions takes the following form

$$\begin{aligned} \sum_{\tau \in \mathcal{T}} (\nabla u, \nabla v)_\tau - \sum_{e_{\text{int}} \in \mathcal{S}(\mathcal{T})} \langle \{\{\nabla u \cdot \hat{\mathbf{n}}\}\}, [v] \rangle_{e_{\text{int}}} - \sum_{e_{\text{ext}} \in \mathcal{E}_D(\mathcal{T})} \langle \nabla u \cdot \hat{\mathbf{n}}, v \rangle_{e_{\text{ext}}} \\ - \sum_{e_{\text{ext}} \in \mathcal{E}_D(\mathcal{T})} \langle u - u_D, \nabla v \cdot \hat{\mathbf{n}} \rangle_{e_{\text{ext}}} + \sum_{e_{\text{ext}} \in \mathcal{E}_D(\mathcal{T})} \mu \langle u - u_D, v \rangle_{e_{\text{ext}}} = 0, \quad \forall v \in V, \end{aligned} \quad (2.16)$$

where μ is a penalty parameter which will be discussed in more detail presently, and where the terms on the Neumann boundary have been cancelled out, due to the homogeneity of the Neumann condition.

The SIPG method enforces interelement continuity in a direct analogue to how Nitsche's method enforces a Dirichlet boundary condition. In this case, of course, rather than attempting to minimise the difference between the prescribed value at the Dirichlet boundary and the computed solution, here the problem is to minimise the difference between the value of the solution on the boundary of an element and the unknown solution on the boundary of the adjacent element, or in other words minimise the jump in the solution across the element edge. This can be written as follows

$$\begin{aligned}
& \sum_{\tau \in \mathcal{T}} (\nabla u, \nabla v)_\tau - \sum_{e_{\text{int}} \in S(\mathcal{T})} \langle \{\{\nabla u \cdot \hat{\mathbf{n}}\}\}, [v] \rangle_{e_{\text{int}}} - \sum_{e_{\text{int}} \in S(\mathcal{T})} \langle \{\{\nabla v \cdot \hat{\mathbf{n}}\}\}, [u] \rangle_{e_{\text{int}}} \\
& + \sum_{e_{\text{int}} \in S(\mathcal{T})} \mu \langle [u], [v] \rangle_{e_{\text{int}}} - \sum_{e_{\text{ext}} \in \mathcal{E}_D(\mathcal{T})} \langle \nabla u \cdot \hat{\mathbf{n}}, v \rangle_{e_{\text{ext}}} - \sum_{e_{\text{ext}} \in \mathcal{E}_D(\mathcal{T})} \langle u, \nabla v \cdot \hat{\mathbf{n}} \rangle_{e_{\text{ext}}} \\
& + \sum_{e_{\text{ext}} \in \mathcal{E}_D(\mathcal{T})} \mu \langle u, v \rangle_{e_{\text{ext}}} = \sum_{e_{\text{ext}} \in \mathcal{E}_D(\mathcal{T})} \langle u_D, \nabla v \cdot \hat{\mathbf{n}} \rangle_{e_{\text{ext}}} + \sum_{e_{\text{ext}} \in \mathcal{E}_D(\mathcal{T})} \mu \langle u_D, v \rangle_{e_{\text{ext}}},
\end{aligned} \tag{2.17}$$

where the comparison with Nitsche's method for enforcing the Dirichlet boundary condition becomes obvious; there is a term reversing the arguments of the IBP surface integral and which therefore renders the bilinear form symmetric and a penalty term which for sufficiently large penalty parameter, μ , ensures the resultant bilinear form is coercive. The parameter μ is a penalty parameter sometimes known as the *discontinuity penalisation parameter* (although it is noted here that the parameter is chosen equivalently for both enforcing continuity and enforcing the Dirichlet boundary condition). The value of the discontinuity penalisation parameter is generally chosen as follows

$$\mu = d(u) \frac{\rho_\mu p^2}{h}, \tag{2.18}$$

where $d(u)$ is a diffusion coefficient (equal to unity for the above linear diffusion problem) and where the constant ρ_μ is a user chosen parameter which is required to be sufficiently large to ensure the Left Hand Side (LHS) of (2.17) is coercive. An optimal choice of penalty parameter is the smallest choice which is large enough to guarantee the coercivity of the associated bilinear form. This is because larger values will negatively effect the conditioning of the resulting linear system particularly for larger polynomial orders. A number of works have attempted to derive optimal penalty parameters using trace inverse inequalities. During the coercivity analysis of IP methods a trace inequality is used which can be stated

$$\int_{\partial\tau} u^2 ds \leq C(p, h) \int_\tau u^2 d\mathbf{x}, \tag{2.19}$$

which bounds the energy of a function u integrated on an element edge with that same function integrated over the element. The lower bound of $C(p, h)$ is the smallest value which ensures

coercivity and therefore implies the optimal choice of the discontinuity penalty parameter. Warburton and Hesthaven in [85] derive a sharp estimate for this constant $C(p, h)$ for simplicial elements of arbitrary polynomial order and dimension, which was then used by Shahbazi [86] and later Epshteyn [87] to derive optimal penalty parameters for the SIPG method applied to diffusion problems. Similarly the constant in the trace inequality is derived for meshes of quadrilateral elements by Burman [88], which is then used to derive penalty parameters by Hillewaert [89], again for diffusion problems. The penalty parameters derived as a result of all of the above analyses are tabulated in Chapter 3 of [89]. Further attempts to derive optimal penalty parameters are relatively sparse beyond this in the literature however. One reason for this could be that where these penalty parameters have been derived they are specific to certain types of elements, basis functions and problems, as is the case in for example [86, 87, 90–92]. Another reason is that where these lower bounds have been derived it is often shown that these values are significantly larger than the critical values determined by experiment for these constants to produce stable results [93], and therefore are not usually optimal in practice. Furthermore, is often simpler to resort to numerical experience for the general case. Based on the experience of researchers such as Houston, [74], it has been suggested that choosing $\rho_\mu = 10$ has consistently shown across a range of problems an appropriate balance between being sufficiently large to weakly enforce continuity accurately but not so large as to significantly affect matrix conditioning. For these reasons and because the value, $\rho_\mu = 10$, is so persistent throughout the literature the same choice is made here and will be used throughout the thesis.

As an aside if one were to rewrite Equation (2.17) introducing the parameter ς as follows

$$\begin{aligned}
& \sum_{\tau \in \mathcal{T}} (\nabla u, \nabla v)_\tau - \sum_{e_{\text{int}} \in \mathcal{S}(\mathcal{T})} \langle \{\{\nabla u \cdot \hat{\mathbf{n}}\}\}, [v] \rangle_{e_{\text{int}}} - \varsigma \sum_{e_{\text{int}} \in \mathcal{S}(\mathcal{T})} \langle \{\{\nabla v \cdot \hat{\mathbf{n}}\}\}, [u] \rangle_{e_{\text{int}}} \\
& + \sum_{e_{\text{int}} \in \mathcal{S}(\mathcal{T})} \mu \langle [u], [v] \rangle_{e_{\text{int}}} - \sum_{e_{\text{ext}} \in \mathcal{E}_D(\mathcal{T})} \langle \nabla u \cdot \hat{\mathbf{n}}, v \rangle_{e_{\text{ext}}} - \varsigma \sum_{e_{\text{ext}} \in \mathcal{E}_D(\mathcal{T})} \langle u, \nabla v \cdot \hat{\mathbf{n}} \rangle_{e_{\text{ext}}} \\
& + \sum_{e_{\text{ext}} \in \mathcal{E}_D(\mathcal{T})} \mu \langle u, v \rangle_{e_{\text{ext}}} = \varsigma \sum_{e_{\text{ext}} \in \mathcal{E}_D(\mathcal{T})} \langle u_D, \nabla v \cdot \hat{\mathbf{n}} \rangle_{e_{\text{ext}}} - \sum_{e_{\text{ext}} \in \mathcal{E}_D(\mathcal{T})} \mu \langle u_D, v \rangle_{e_{\text{ext}}},
\end{aligned}
\tag{2.20}$$

then it is possible to return an NIPG discretisation of the problem (2.7) by choosing $\varsigma = -1$, and an IIPG discretisation by choosing $\varsigma = 0$.

A number of the diffusion equations which will be encountered in this work are quasilinear, that is they take the form

$$\begin{aligned}
\nabla \cdot (d(u(\mathbf{x})) \nabla u(\mathbf{x})) &= 0, & \mathbf{x} \in \Omega, \\
u(\mathbf{x}) &= u_D, & \mathbf{x} \in \partial\Omega_D, \\
d(u(\mathbf{x})) \nabla u(\mathbf{x}) \cdot \hat{\mathbf{n}} &= 0, & \mathbf{x} \in \partial\Omega_N,
\end{aligned}
\tag{2.21}$$

where the diffusion coefficient $d(u)$ is a nonlinear function. Where these quasilinear diffusion equations occur they will have a heterogeneous Dirichlet boundary condition (identical to the linear case in Equation (2.7)), and a nonlinear Neumann boundary condition (which by virtue

of the Neumann condition having a zero Right Hand Side (RHS) can also be treated identically to the linear homogeneous Neumann condition in Equation (2.7)). An SIPG discretisation of (2.21) can thus be stated as follows by direct analogue with the linear case above

$$\begin{aligned}
& \sum_{\tau \in \mathcal{T}} (d(u) \nabla u, \nabla v)_\tau - \sum_{e_{\text{int}} \in S(\mathcal{T}) \cap \mathcal{E}_D} \langle \{ \{ d(u) \nabla u \cdot \hat{\mathbf{n}} \} \}, [v] \rangle_{e_{\text{int}}} \\
& - \sum_{e_{\text{int}} \in S(\mathcal{T}) \cap \mathcal{E}_D} \langle \{ \{ d(u) \nabla v \cdot \hat{\mathbf{n}} \} \}, [u] \rangle_{e_{\text{int}}} + \sum_{e_{\text{int}} \in S(\mathcal{T}) \cap \mathcal{E}_D} \mu \langle [u], [v] \rangle_{e_{\text{int}}} \\
& = \sum_{e_{\text{ext}} \in \mathcal{E}_D(\mathcal{T})} \langle u_D, d(u) \nabla v \cdot \hat{\mathbf{n}} \rangle_{e_{\text{ext}}} + \sum_{e_{\text{ext}} \in \mathcal{E}_D(\mathcal{T})} \mu \langle u_D, v \rangle_{e_{\text{ext}}}, \quad \forall v \in V, \quad (2.22)
\end{aligned}$$

where for a given element, $\tau_+ \in \mathcal{T}$, which shares an edge with the domain boundary, $\partial\Omega$, the jump and average operators are defined as,

$$[[\psi]] = \psi_{\tau_+} \hat{\mathbf{n}}_{\tau_+}, \quad (2.23)$$

$$\{ \{ \Psi \} \} = \Psi_{\tau_+}. \quad (2.24)$$

Replacing the infinite dimensional solution space, V , with the finite dimensional approximation, $V_{\mathbf{p}}(\mathcal{T})$, defined in (2.2), the variational formulation of the quasilinear diffusion equation, (2.21), discretised using a SIPG discretisation can be stated: find $u_h \in V_{\mathbf{p}}(\mathcal{T})$ such that

$$B_{\text{dif}}(u_h; u_h, v_h) = J_{\text{dif}}(v_h), \quad \forall v_h \in V_{\mathbf{p}}(\mathcal{T}), \quad (2.25)$$

where the form $B_{\text{dif}}(w_h; u_h, v_h)$ which is bilinear in u_h and v_h can be stated

$$\begin{aligned}
B_{\text{dif}}(w_h; u_h, v_h) &= (d(w_h) \nabla u_h, \nabla v_h)_{\mathcal{T}} - \langle \{ \{ d(w_h) \nabla u_h \} \}, [v_h] \rangle_{S(\mathcal{T}) \cap \mathcal{E}_D} \\
& - \langle \{ \{ d(w_h) \nabla v_h \} \}, [u_h] \rangle_{S(\mathcal{T}) \cap \mathcal{E}_D} + \mu(w_h) \langle [u_h], [v_h] \rangle_{S(\mathcal{T}) \cap \mathcal{E}_D}, \quad (2.26)
\end{aligned}$$

and the form $J_{\text{dif}}(v_h)$ which is linear in v_h can be stated

$$J_{\text{dif}}(v_h) = \langle u_D, d(u_h) \nabla v_h \cdot \hat{\mathbf{n}} \rangle_{\mathcal{E}_D} + \mu \langle u_D, v_h \rangle_{\mathcal{E}_D}, \quad (2.27)$$

where for brevity, the following notation has been used

$$(\cdot, \cdot)_{\mathcal{T}} := \sum_{\tau \in \mathcal{T}} (\cdot, \cdot)_{\tau}, \quad \text{and} \quad \langle \cdot, \cdot \rangle_{\mathcal{E}} := \sum_{e \in \mathcal{E}} \langle \cdot, \cdot \rangle_e, \quad (2.28)$$

a convention which will continue for the remainder of the thesis. It should be noted that this variational formulation stated here also generalises to the linear case (2.7) by appropriate choice of the diffusion coefficient.

2.3.2 DG methods for advective PDEs

As mentioned previously, DG methods were first conceived in the context of linear hyperbolic conservation laws, in particular the neutron transport equation with the seminal paper by Reed

and Hill [35]. The following year an analysis of Reed and Hill’s method, was presented by LeSaint and Raviart [36] who proved a convergence rate in the L^2 norm of h^p on general triangulations and for a basis of tensor product polynomials of order, p , on Cartesian meshes, a rate of convergence of h^{p+1} . Later, it was proven by Johnson and Pitkäranta [94] that on general triangulations the rate of convergence for the method was $h^{p+\frac{1}{2}}$ in the L^2 norm, an estimate that was demonstrated numerically to be sharp by Peterson [95]. On “semiuniform” triangulations Richter [96] proved that the optimal rate of convergence for the method was of the order h^{p+1} . Cockburn et al. [97] later showed that the assumptions on the uniformity and conformity of meshes chosen by Richter could be extended and that the optimal order of convergence h^{p+1} , could be achieved on any mesh made of simplices each of which has a unique outflow face.

One of the first extensions of the DG method to nonlinear conservation laws was presented in 1982 by Chavent and Solzano [44]. Rather than solving a global implicit system, the ability to solve element by element as in Reed and Hill’s original DG method can be preserved through the use of explicit time stepping methods. As such, in their work a DG discretisation in space is combined with an explicit Euler method in time, for studying one dimensional water flooding problems. Like all explicit methods, in order to ensure numerical stability (bounded growth of truncation errors over time), the explicit Euler discretisation must satisfy a Courant-Friedrichs-Lewy (CFL) condition. A physical interpretation of this CFL condition is that in order to ensure numerical stability, a wave emanating from the boundary of an element is not allowed evolve further than one element width in any given time step, and thus for a given maximum wave speed and minimum element size one can determine an appropriate time step to ensure stability. When applied to nonlinear hyperbolic conservation laws, however, one issue with the explicit Euler method in particular is that it can be shown via a Von Neumann stability analysis that there is a nonlinear relationship between the required time step size and the given mesh size, h , for a constant Courant number, on the order of $h\sqrt{h}$, [98]. The result of this is that such a discretisation suffers from a severe time step restriction which can be prohibitively expensive for hyperbolic problems. This was also shown in a follow up work by Chavent and Cockburn [99] (in which the method presented in [44] is expanded to include a slope limiter) which presents a similar stability analysis showing a nonlinear relationship between the time step size and grid spacing for a constant Courant number.

These issues were overcome in a series of papers by Cockburn and Shu [64–68], which presented the RKDG method. The first of these articles, [64], expanded on the method of Chavent and Cockburn by modifying the slope limiter which allowed for the recovery of accuracy in the regions surrounding local extrema, and by replacing the time discretisation by a second-order TVD RK scheme. The resulting scheme was proven linearly stable for fixed Courant number, and was shown to be formally second-order accurate in space and time. Through the series of articles that followed the method was then extended to arbitrary-order by using RK schemes of order $p + 1$ for DG discretisations of order p [65], to one dimensional systems [66], to multiple dimensions, with appropriate generalisations of the proposed slope limiter [67] and finally to multiple dimensional systems [68]. Analysis of DG methods for nonlinear conservation laws is still an active area of research. One important result is the local entropy inequality presented

by Jiang and Shu [100], which implies for problems discretised using DG in space, that the L^2 norm of the solution does not increase over time, even without limiters. More recently an RKDG discretisation of nonlinear conservation laws using a second-order TVD RK method in time was analysed and quasi-optimal *a priori* error estimates were obtained [101]. Similarly, analysis of a DG scheme in space with a third order TVD RK scheme in time for solving nonlinear conservation laws with possibly discontinuous solutions was presented [102] and sub-optimal *a priori* error estimates were proven for general monotone fluxes, whilst optimal convergence was shown for upwind schemes on problems which were sufficiently smooth.

2.3.2.1 DG discretisations for advective PDEs

A general first-order hyperbolic advection equation can be stated

$$\begin{aligned} \frac{\partial u(\mathbf{x}, t)}{\partial t} + \mathbf{b} \cdot \nabla u(\mathbf{x}, t) &= 0, & \mathbf{x} \in \Omega, \\ u(\mathbf{x}, t) &= u_-, & \mathbf{x} \in \partial\Omega_-, \\ u(\mathbf{x}, 0) &= u^0, & \mathbf{x} \in \Omega, \end{aligned} \quad (2.29)$$

where \mathbf{b} denotes the advection velocity vector, u_- denotes the solution along the boundary, $\partial\Omega_-$ denotes the inflow part of the boundary, and u^0 denotes the initial condition. The inflow part of the boundary is defined as

$$\partial\Omega_- = \{\mathbf{x} \in \partial\Omega : \mathbf{b} \cdot \hat{\mathbf{n}} < 0\}. \quad (2.30)$$

On the natural boundary u_- will be prescribed, whereas for an internal inflow edge on element, τ_+ , u_- denotes the trace of u on the edge of element τ_+ , from inside the element τ_- . To discretise (2.29) one once again begins in the standard finite element way; multiplying by a test function, $v \in V$, and integrating by parts which gives

$$\sum_{\tau \in \mathcal{T}} \left(\frac{\partial u(\mathbf{x}, t)}{\partial t}, v \right)_{\tau} - \sum_{\tau \in \mathcal{T}} (u(\mathbf{x}, t), \nabla \cdot (\mathbf{b}v))_{\tau} + \sum_{\partial\tau \in \partial\mathcal{T}} \langle \mathbf{b} \cdot \hat{\mathbf{n}} u(\mathbf{x}, t), v \rangle_{\partial\tau} = 0, \quad \forall v \in V. \quad (2.31)$$

As for the diffusion problem one can then decompose the surface integrals over the element edges into integrals over the internal edges, $S(\mathcal{T})$, and integrals over the inflow edges, $\partial\Omega_-$, as follows

$$\begin{aligned} \sum_{\tau \in \mathcal{T}} \left(\frac{\partial u(\mathbf{x}, t)}{\partial t}, v \right)_{\tau} - \sum_{\tau \in \mathcal{T}} (u(\mathbf{x}, t), \nabla \cdot (\mathbf{b}v))_{\tau} + \sum_{e_{\text{int}} \in S(\mathcal{T})} \langle \{\{ \mathbf{b}u(\mathbf{x}, t) \}\}, \llbracket v \rrbracket \rangle_{e_{\text{int}}} \\ + \sum_{e_{\text{ext}} \in \partial\Omega_-} \langle \mathbf{b} \cdot \hat{\mathbf{n}} (u(\mathbf{x}, t) - u_-), v \rangle_{e_{\text{ext}}} = 0, \quad \forall v \in V. \end{aligned} \quad (2.32)$$

The flux term which naturally arises on the internal edges, $\{\{ \mathbf{b}u \}\}$, is known to be too dissipative, and therefore can lead to non-physical oscillations in the solution. For this reason, the flux across internal edges is usually approximated by a numerical flux, $\hat{\mathbf{b}} \simeq \{\{ \mathbf{b}u \}\}$, which is chosen such that stability can be ensured (this is equivalent to choosing an appropriate approximate Riemann solver). One common approach to solve this issue is to replace the average value of the solution

over the edge with a numerical flux which takes the following form

$$\{\{ \mathbf{b}u \}\} \simeq \hat{\mathbf{b}} = \{\{ \mathbf{b}u \}\} + c_{\partial\tau} \llbracket u \rrbracket. \quad (2.33)$$

In this way, as shown in [103], the standard upwind flux formulation can be returned by simply choosing the parameter $c_{\partial\tau}$ as follows

$$c_{\partial\tau} = \frac{1}{2} |\mathbf{b} \cdot \hat{\mathbf{n}}|. \quad (2.34)$$

Another common numerical flux, the Lax-Friedrichs (LF) flux can similarly be defined by choosing the parameter $c_{\partial\tau}$ as

$$c_{\partial\tau} = \frac{1}{2} \|\mathbf{b} \cdot \hat{\mathbf{n}}\|_2, \quad (2.35)$$

where $\|\cdot\|_2$ denotes the Euclidean norm. It should be noted that for linear advection problems the LF flux will reduce to the equivalent upwind formulation [104]. Upwind fluxes are generally preferred for dealing with pure advection problems as it is a directional phenomenon, and upwind fluxes ensure that the information can only flow in the upwind direction. Lax-Friedrichs type fluxes include an additional central flux type term which is a form of artificial viscosity and as such the LF flux is typically preferred for the discretisation of advection-diffusion type problems. Whilst an appropriately chosen flux for a given problem is requisite to ensure stability, evidence is presented by Cockburn and Shu [105] which suggests that for high-order DG discretisations the effect on accuracy of various fluxes is less significant. This is confirmed by Wheatley et al. [106], in the context of smooth problems however it is also shown in their article that in the context of highly discontinuous solutions, careful consideration of the chosen flux can result in cheaper to compute and more accurate simulations.

The variational formulation of equations of the type (2.29) can therefore be stated: find $u \in V$ for each $t = (0, T)$ such that

$$\left(\frac{\partial u(\mathbf{x}, t)}{\partial t}, v \right)_{\mathcal{T}} - (u(\mathbf{x}, t), \nabla \cdot (\mathbf{b}v))_{\mathcal{T}} + \langle \hat{\mathbf{b}}, \llbracket v \rrbracket \rangle_{S(\mathcal{T})} + \langle \mathbf{b} \cdot \hat{\mathbf{n}} (u(\mathbf{x}, t) - u_-), v \rangle_{\partial\Omega_-} = 0, \quad \forall v \in V. \quad (2.36)$$

Thus the DG approximation of the variational formulation can be stated: find $u_h \in V_{\mathbf{p}}(\mathcal{T})$ for each $t = (0, T)$ such that

$$\left(\frac{\partial u_h}{\partial t}, v_h \right)_{\mathcal{T}} + B_{\text{adv}}(u_h, v_h) = J_{\text{adv}}(v_h), \quad \forall v_h \in V_{\mathbf{p}}(\mathcal{T}), \quad (2.37)$$

where

$$B_{\text{adv}}(u_h, v_h) = \langle \hat{\mathbf{b}}, \llbracket v_h \rrbracket \rangle_{S(\mathcal{T})} + \langle \mathbf{b} \cdot \hat{\mathbf{n}} u_h, v_h \rangle_{\partial\Omega_-} - (u_h, \nabla \cdot (\mathbf{b}_h v_h))_{\mathcal{T}}, \quad (2.38)$$

and

$$J_{\text{adv}}(v_h) = - \langle \mathbf{b} \cdot \hat{\mathbf{n}} u_-, v_h \rangle_{\partial\Omega_-}. \quad (2.39)$$

2.4 Summary

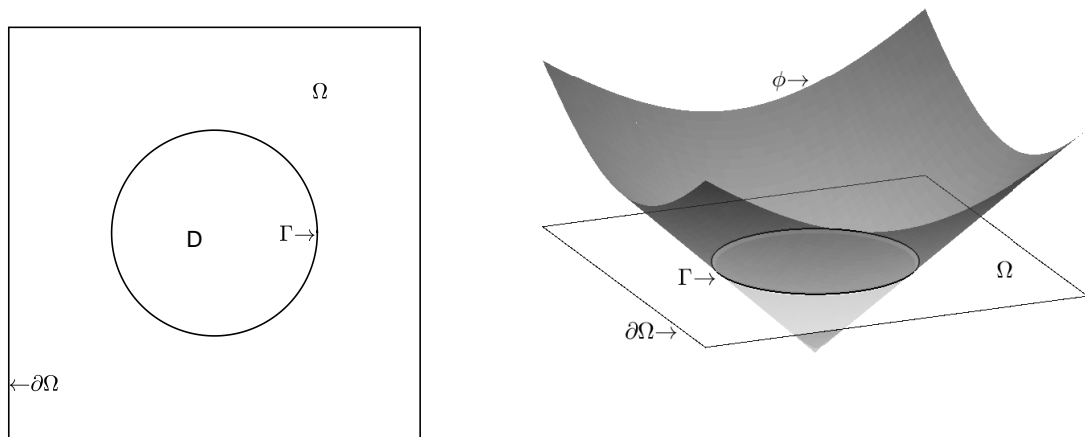
This chapter has provided an introduction to DG methods, including a review of the wider literature concerning numerical methods for PDEs, a review of the literature specific to DG discretisations of the problems to be encountered in the remainder of this thesis, and the presentation of the nomenclature necessary for discussions which follow, including example discretisations. This is the first of two introductory chapters which aim to equip the reader with the context necessary for the research to follow. The next chapter is the second of these introductory chapters, which presents an introduction to and presentation of the preliminaries concerning the level set method, such that beyond these introductory chapters, Chapter 2 and Chapter 3, the thesis can proceed to use appropriate DG discretisations for the relevant equations comprising a level set methodology, which will then form part of a level set based topology optimisation method.

Chapter 3

The Level Set Method

3.1 Overview

Broadly speaking one could divide methods for structural topology optimisation into two classes which could be defined as; one where the position of the internal free boundaries (interfaces) are treated as the design variable, and a second where the density of the material inside each of a set of discrete control volumes is the design variable. Methods belonging to either of these classes have their advantages and disadvantages, however, assuming one is to choose a method belonging to the first of these two classes, a technique will be required for representing and tracking the evolving interfaces. In this context the most used tool is the level set method which was originally developed by Osher and Sethian, [7], in 1988. Level set methods are an implicit method of interface capturing, meaning that the position of the internal boundaries are implied by a function which can be tracked or evolved in an Eulerian manner. By virtue of the implicit nature of the method topological changes can be handled naturally by the method, as opposed to explicit methods whereby the merging of multiple interfaces or the splitting of an interface would require complicated algorithms to detect and deal with such events. This is also advantageous as it means that the method is efficient, as there is no need to discretise the interface (or otherwise make explicit its position), or to compute any kind of remeshing which might be necessary with other interface fitting type methods. For these reasons the level set method has become a popular tool for solving topology optimisation problems. The seminal works on level set based topology optimisation were presented in the 2000's. The first of these was an article by Sethian and Weigmann, [107], which presented a method for the optimal design of elastic structures where a level set method is used to represent the internal boundaries which were evolved by an *ad hoc* measure based on the Von Mises stress. In 2001, Osher and Santosa [8] presented a similar method, however with a more mathematically rigorous method for computing the advection velocity field used to drive the evolution of the interface towards its optimal position, based on functional derivatives of the underlying optimisation problem. Then in 2004, Allaire et al. [9] generalised further this approach of using shape sensitivity analysis to compute an appropriate interface advection velocity. A more detailed literature review concerning contemporary methods for solving topology optimisation problems will be presented in Chapter 7.



(a) The problem domain with interface, Γ .

(b) The level set function, ϕ , intersecting the problem domain, Ω .

Figure 3.1: A circular interface on a square domain, defined using the level set method.

3.2 Mathematical preliminaries

The idea behind the level set method, is to implicitly represent an interface, Γ , by the intersection between a real scalar valued function known as a level set function, ϕ , and the hyperplane which is the problem domain, Ω . By choosing the problem domain to be the zero-plane, the interface will then be defined as the zero level set, of the level set function, i.e. the position of the interface is along the set of points at which the level set function equals zero. This can be stated as

$$\begin{aligned}
 \phi(\mathbf{x}, t) &> 0, & \text{if } \mathbf{x} \in \Omega \setminus D, \\
 \phi(\mathbf{x}, t) &= 0, & \text{if } \mathbf{x} \in \Gamma, \\
 \phi(\mathbf{x}, t) &< 0, & \text{if } \mathbf{x} \in D,
 \end{aligned}
 \tag{3.1}$$

where the level set interface, Γ , divides the problem domain, Ω , into two subdomains, $\Omega \setminus D$ and D . An example of this is presented in Figure 3.1 which demonstrates graphically how the level set method can be used to represent a circular interface on a square domain. The level set function itself, $\phi = \phi(\mathbf{x}, t)$, is a function of space, $\mathbf{x} = \{x, y\}$, and pseudotime, t .

The level set interface, Γ , can then be tracked or evolved by appropriately computing an advection velocity field, \mathbf{b} , over the problem domain, and using this to drive a scalar transport problem, which evolves the level set function and thus implicitly the interface. This transport problem takes the form of a pure advection equation which in this context is sometimes called the level set equation, and can be stated as

$$\frac{\partial \phi(\mathbf{x}, t_e)}{\partial t_e} + \mathbf{b} \cdot \nabla \phi(\mathbf{x}, t_e) = 0,
 \tag{3.2}$$

where t_e is pseudotime as related to the evolution equation.

3.2.1 Level set initialisation and reinitialisation

The initial condition for the level set evolution problem, $\phi(\mathbf{x}, 0)$, is a user decided quantity. Two considerations need to be taken into account when making this choice: the initial position of the interface, and the desired numerical qualities of the level set function. The initial position of the level set interface is problem dependent and will likely be obvious or at least strongly informed by the problem to be solved. Using an example from structural optimisation, when computing the minimum compliance of a structure it is known that one should consider a large number and regular distribution of holes (internal boundaries) in the initial condition. This is because the more holes there are, the greater the complexity of the internal structures, and furthermore, for a sufficiently stable method no new holes can be generated during level set evolution. Therefore the more holes implied by the initial level set function, the closer the converged solution can be to the global minimum. With regards to the desired numerical qualities of the level set function however, it can be noted that at each point along the level set interface, the interface can only be transported along the normal pointing away from the interface at that point. A natural choice then, is to initialise the level set function as a signed distance function to the desired position of the initial interface. A level set function, $\phi(\mathbf{x}, t_e)$, which is a signed distance function can be defined as,

$$\phi(\mathbf{x}, t_e) = \pm \text{dist}(\mathbf{x}, \Gamma), \quad (3.3)$$

where $\text{dist}(\mathbf{x}, \Gamma)$ is the minimum distance from the point \mathbf{x} to the interface, Γ . Using the notation from (3.1), the sign of the level set function, (\pm) , could then be defined as positive for $\mathbf{x} \in \Omega \setminus D$ and negative for $\mathbf{x} \in D$. For reference, the example level set function shown in Figure 3.1(b), is a signed distance function to the interface. One property of a signed distance function, which will be referred to as the *signed distance property*, is that a signed distance function will satisfy an Eikonal equation, which can be stated as

$$|\nabla \phi(\mathbf{x}, t_e)| - 1 = 0. \quad (3.4)$$

Satisfaction of this signed distance property is desirable for the evaluation of level set problems as it has been shown that large variations in the gradient of the level set function, can both cause and facilitate the propagation of numerical instabilities during the solution of the level set evolution problem, [108]. Furthermore, it is known that strongly satisfying the signed distance property is required when one desires high levels of accuracy [23]. For these reasons, the signed distance function is usually considered the best choice for the initialisation of a level set problem in terms of desirable numerical qualities and will in all cases be the choice made for the problems considered in this thesis.

Whilst the initial level set function is a choice made by the user, it is also beneficial to ensure that the level set function maintains satisfaction of the signed distance property throughout its evolution. In the general case however, after any given iteration in the solution of the evolution problem, the level set function is likely to have evolved such that it no longer satisfies the signed distance property. For this reason, researchers have developed numerous techniques which allow one, at any point during level set evolution, i.e. for a known (implicitly) fixed interface position,

to *reinitialise* the level set function as a signed distance function. This topic is an important one in the context of the research to be presented in this thesis and as such the details will be deferred to Chapter 4 which will present a literature review concerning level set reinitialisation as well as novel reinitialisation techniques for DG based level set methods.

3.2.2 Narrow banded level set methods

Whether for purposes of accuracy, stability or both, the maximum distance that the level set interface can evolve during a given iteration will be limited. When discretising the problem temporally using explicit methods, the maximum amount of interface movement will be limited by the CFL condition, which will be such that the interface can move from the element within which it currently resides to one of its neighbours (where a neighbour of element, τ , is defined as any element with which τ shares a mesh node). Since the interface cannot at any given iteration evolve beyond this, and furthermore as information pertaining to the level set function beyond these elements also has no effect on the evolution of the interface, the PDEs to be solved concerning the level set function, need only be solved on elements ‘near’ to the interface. This is known as a narrow band approach, [109], and is a common part of level set methodologies (and one which will be adopted in this work) as it allows one to reduce the computational expense of level set problems.

Computational efficiency, however, isn’t the only benefit of using a narrow band approach. One of the issues with choosing the level set function to be a signed distance function, as stated above, is that if the zero isocontour of the level set function, $\Gamma(\phi)$, has at least one loop surrounding a simply-connected subdomain, D for example, there will always be a singularity which occurs in the level set function. This can be observed at the peak of the conic level set function describing the circular interface in Figure 3.1. An added benefit of the adoption of a narrow band approach in such an instance is that for a ‘sufficiently refined’ mesh almost all of these areas would be far enough away from the level set interface so as to fall outside of the narrow band. It is known that high-order accuracy is a function of the smoothness of the solution when discretising a problem using DG methods, [110], and since these singularities will always occur, the use of a narrow band approach is therefore necessary to allow one to demonstrate optimal orders of convergence when solving level set problems using DG methods. (Although it should be stated that in the case that the interface itself is non-smooth this of course would not be possible.)

3.3 Literature review: discontinuous Galerkin level set methods

Research in the area of level set methods, discretised using discontinuous Galerkin methods is one still in its relative infancy and thus the literature is relatively sparse. One notable area of interest to researchers in this regard involves the application of level set methods to multiple phase immiscible incompressible flows. Over the last few decades, the most popular method for discretising level set problems of this type spatially have been finite difference ENO or Weighted Essentially Non-Oscillatory (WENO) type schemes. One of the known issues with such discretisations however, is that there can be a significant amount of numerical dissipation which leads to a phenomenon known as mass loss. This type of issue has been shown to be particularly

significant in regions where the interface is a sharp corner or is otherwise discontinuous. A number of works have suggested that the use of DG methods might be a potential solution to this issue, as DG methods are able to resolve sharp corners more effectively [111], and similarly due to the fact that DG methods have been shown to be both more accurate and more efficient in time than comparable ENO/WENO schemes, [112]. This idea led to early works on DG based level set methods for two phase flow, for example by Osher and Yan, [24] and Marchandise [113]. Both of these works take advantage of an incompressibility assumption which allows one to rewrite the advective form of the level set equation as presented in Equation (3.2) in the form of a hyperbolic conservation law, as follows

$$\frac{\partial \phi(\mathbf{x}, t_e)}{\partial t_e} + \nabla \cdot (\mathbf{b} \phi(\mathbf{x}, t_e)) = 0. \quad (3.5)$$

When the evolution equation is written in this form the level set method is usually referred to as the conservative level set method. As is demonstrated in [114], the conservative form of the level set equation can introduce erroneous offset in the location of the level set interface, and as such is not a formulation which would be appropriate for the purposes of the work to be presented in this thesis. However, as a significant amount of the work done on DG level set methods concerns interface problems between incompressible fluids, much of the research in this area is on the discretisation of the conservative level set equation. In [24], Osher applies two DG spatial discretisations to the conservative level set equation, (3.5); the standard upwind DG scheme for hyperbolic problems, as well as a DG method using a Lax-Friedrichs flux. It is found in [24], that both types of flux considered are stable and both demonstrate similar levels of accuracy when solving the conservative level set equation. The temporal discretisation used in [24] is an explicit Strong Stability Preserving (SSP) RK method. Thus the full discretisation used is the same RKDG scheme that is presented in [68], by Cockburn and Shu. Similarly, in [113], a quadrature-free upwind DG formulation is used to discretise (3.5) spatially, with an explicit RK method of order, $p + 1$, applied in time on meshes of maximum polynomial order, p . In this work it was again found that DG methods outperformed ENO methods in terms of accuracy and efficiency.

The combination of an explicit RK method in time, with a DG method in space using either an upwind or Lax-Friedrichs flux is a theme that persists throughout the literature in the context of discretising the equations which form the level set methodology. For example, similar schemes are presented in [115–121]. In [115], it was shown that such a method was accurate on the order of h^{p+1} on meshes using order p Legendre polynomials as a basis. Furthermore, a LF flux was compared with the upwind flux in [119], where it was found that the LF flux was preferable in the case where it is known that the solution to the level set evolution equation is smooth. Some examples of variations on this theme include methodologies adopting implicit backward differencing time schemes, in works such as [25, 122–125]. Similarly, a number of works use a DG discretisation for the temporal variables, which is known as a Space-Time Discontinuous Galerkin (STDG) method, which is another implicit time scheme, for example in [33, 126]. In almost all cases, the motivation for using implicit temporal schemes is to avoid (or at least reduce)

time stepping restrictions usually as a result of the stiffness of the underlying problem which is driving the evolution of the level set interface; this is beneficial as none of the aforementioned works take advantage of the high levels of parallelisation which is possible using an explicit in time formulation.

Despite a wealth of literature concerning analysis of DG methods discretising advection-diffusion type problems, of which the relevant are outlined in the introduction to [124], the literature concerning error analysis for DG level set methods is also relatively sparse. The 2006 article by Osher, [24], presents a bound on the L^2 error of the order $h^{p+\frac{1}{2}}$, for meshes of elements of maximum order p , for the method contained within. In [124], an upwind DG in space, Crank-Nicolson in time, discretisation of the advective level set equation is analysed. For the full discretisation an error estimate is derived of the order, $h^{p+\frac{1}{2}} + \Delta t_e^2$, on meshes of elements with maximum polynomial order p . In [33], an analysis is performed on the STDG discretisation of the level set equation, and an error estimate is derived bounding the error, for problems where the spatial solution exists in the Sobolev space, H^s , on the order of $h^{2\min\{p+1,s\}-2} + \Delta t_e^{2q}$, for a spatial DG scheme of maximum order p and a temporal DG scheme of maximum order q .

3.4 Summary

After the presentation of the DG preliminaries in Chapter 2, Chapter 3 has followed in a similar vein presenting context concerning the level set method as a technique for topology optimisation, as well as a review of the literature concerning level set methods which take advantage of DG discretisations. Also presented in this chapter are the mathematical preliminaries regarding the level set method which includes details concerning level set initialisation/reinitialisation, level set evolution, and narrow banded level set methods. As level set evolution, for the purposes presented in this thesis, requires reinitialisation, the next chapter proceeds by presenting research concerning level set reinitialisation in the DG paradigm.

Chapter 4

Level Set Reinitialisation

In Section 3.2.1 it was established that it is beneficial to ensure that the initial level set function is chosen such that it is a signed distance function to the desired initial position of the interface. For the same reasons, it is generally considered desirable to ensure that the level set function is a signed distance function for all time during level set evolution. It is unlikely, however, that after the level set function is advected subject to a given velocity field, \mathbf{b} , for any non-zero amount of pseudotime, that the level set function will continue to maintain satisfaction of the signed distance property. In order to rectify this, after any given iteration of the evolution equation, one can consider a separate problem by which there is a known (implicitly) interface position, $\Gamma(\phi(\mathbf{x}, t_e))$, but a level set function which no longer sufficiently satisfies the Eikonal equation, (3.4), and therefore which needs to be *reinitialised* as a signed distance function.

The level set reinitialisation problem can therefore be stated: at any time during the solution of the evolution equation, t_e , given the level set function at that point in time, $\tilde{\phi}_h^0 = \phi(\mathbf{x}, t_e)$, find a new level set function, $\phi(\mathbf{x}, t_e)$, which is a signed distance function to the original position of the level set interface, $\Gamma(\tilde{\phi}_h^0)$. This can be stated mathematically as finding a solution to the Eikonal equation, stated in Equation (3.4), relative to the following Dirichlet boundary condition

$$\tilde{\phi}_h(\mathbf{x}, t_r) = 0, \quad \mathbf{x} \in \Gamma(\tilde{\phi}_h^0), \quad (4.1)$$

where t_r denotes pseudotime related to reinitialisation problem. Throughout this thesis, the notation, $\tilde{\phi}$, denotes the solution variable to the reinitialisation problem, which can be contrasted to, ϕ , which denotes the solution variable to the level set evolution equation. In this way, the initial condition to the reinitialisation problem, $\tilde{\phi}_h^0$ will be equal to the pre-reinitialisation level set function at the time that the reinitialisation routine is called, $\phi(\mathbf{x}, t_e)$, and the converged solution to the reinitialisation routine, $\tilde{\phi}_h^\infty = \tilde{\phi}_h(\mathbf{x}, \infty)$, will then replace the inputted level set function in the evolution problem at that time step, $\phi(\mathbf{x}, t_e)$.

Whilst the desire (and sometimes requirement) for level set reinitialisation exists to ensure stability and accuracy when solving interface problems using the level set method, level set reinitialisation is oftentimes considered a necessary evil. The reason for this is that the requirement to solve a second problem, on top of the level set evolution problem, both reduces computational efficiency, and also adds a potential additional source of errors into the method. This has lead

to an abundance of research in the area of level set reinitialisation in order to minimise these potential issues. DG based level set methodologies, however, are still in their relative infancy, as was discussed in the literature review in Section 3.3, and as such the translation of existing reinitialisation methods, or the development of new reinitialisation methods in the context of DG discretisations of the level set method is relatively sparse. As such in order to satisfy the stated aims of this thesis, that is in order to develop a robust, efficient, high-order accurate DG based level set methodology for solving topology optimisation problems, it was necessary to develop a robust, efficient, and high-order accurate DG based level set reinitialisation method.

To this end, during this PhD two journal articles were published presenting DG based level set reinitialisation methods, [127] and [128]. The contribution of this chapter of the thesis will be the presentation of the work published in those two articles in one unified framework. As such the remainder of this chapter will comprise the following subsections. First of all, in Section 4.1, the literature surrounding level set reinitialisation will be reviewed with a particular focus on where DG discretisations have been adopted and the shortfalls of those proposed solutions. Section 4.2 presents the derivation of the strong form of a particular type of reinitialisation method based on the minimisation of the least squares residual of the Eikonal equation, (3.4), which can then be discretised in a number of ways. Section 4.3, and its subsections formulate the Eikonal minimising reinitialisation method as an Elliptic PDE, before discussing some of the intricacies which need to be considered in order to ensure the resulting Elliptic Reinitialisation method is robust and high-order accurate. Section 4.4, uses the discussion presented in Section 4.3 to make the necessary modifications to the Eikonal Minimising Elliptic Reinitialisation method to present the author’s preferred discretisation of the Elliptic Reinitialisation method. Also presented in Section 4.4 are a number of numerical examples demonstrating the efficacy of the proposed formulation of the Elliptic Reinitialisation method. Section 4.5, presents the Parabolic Reinitialisation method, which again uses the discussion presented in Section 4.3 to modify the Eikonal Minimising Reinitialisation method, however this time the resulting method takes the form of a Parabolic PDE. A number of possible discretisations of the Parabolic Reinitialisation method are presented which are compared both with each other and with the Elliptic Reinitialisation method through a number of numerical experiments, which can also be found in Section 4.5. The chapter is then concluded in Section 4.6 with the author’s reasons for deciding that the Elliptic Reinitialisation method, as presented in Section 4.4, is the preferred Reinitialisation method moving forward.

4.1 Literature review: level set reinitialisation

There is a relatively large literature surrounding level set reinitialisation and thus there already exist many methods which are capable of reinitialising a level set function as a signed distance function. In this section, this literature will be reviewed, with a focus on where DG discretisations have been applied to such methods. It should be noted first and foremost that all of the reinitialisation techniques discussed below are equivalent in what they aim to do, generating a signed distance function to the level set interface at any given point during level set evolution. However, the methods to be discussed vary in terms of their computational efficiency, stability

and accuracy, especially when using a DG discretisation.

In general, reinitialisation methods fall into one of two categories: geometric methods, and PDE based methods. Geometric methods as their name suggests, attempt to reinitialise the level set function at a set of discrete points by measuring the distance from these points to the level set interface, whilst preserving the polarity of the level set function pre-reinitialisation to generate a signed distance function. PDE based methods, also aptly named, involve generating a signed distance function, by solving a PDE. PDE based methods can be further divided into two subcategories. One type of PDE based reinitialisation method, referred to henceforth as *pure reinitialisation* methods, require a separate PDE (a pure reinitialisation problem) to be solved, the solution of which will be a signed distance function to the interface. This is in contrast to the second type of PDE based reinitialisation method which will be referred to as *all-in-one* methods, which modify the level set evolution equation, (3.2), itself to include a constraint enforcing that the level set function always satisfies the Eikonal equation, (3.4), such that both the evolution of the interface and reinitialisation of the level set function are computed simultaneously.

The original idea of reinitialising the level set function as a signed distance function to overcome stability issues was first considered in 1993 by Chopp [22], who developed a geometric, brute force approach to reinitialisation. This approach works by first explicitly discretising the interface, $\Gamma_h(\tilde{\phi}_h^0) \sim \Gamma(\tilde{\phi}_h^0)$, and then at a set of discrete points in the problem domain (in this case, mesh nodes) defining the new value of the level set function, $\tilde{\phi}_h$, as equal to the minimum distance from that point, to the discrete interface multiplied by the sign of the original level set function at that point, which can be stated as

$$\tilde{\phi}_h(\mathbf{x}) = \text{sign} \left(\tilde{\phi}_h^0(\mathbf{x}) \right) \text{dist} \left(\mathbf{x}, \Gamma_h \left(\tilde{\phi}_h^0(\mathbf{x}) \right) \right), \quad (4.2)$$

where $\text{sign}(\cdot)$ denotes the signum function.

Whilst Chopp's geometric reinitialisation approach is attractive in its simplicity, there are a number of issues with such geometric reinitialisation methods. One of the major advantages of using the level set method for solving interface problems, in terms of computational efficiency, is the implicit nature of the evolving interface. Not only is this advantage surrendered by discretising the interface, but the expense required to both discretise the interface and compute the minimum distance at each node in the mesh, to the discrete interface, increases with mesh density, with the number of points used to discretise the interface and with the length of the interface itself. This is evidenced in the original article, [22], where it is noted that the complexity of this reinitialisation method is $\mathcal{O}(n^6)$. Furthermore, adaptation of such an approach to meshes of non-conforming elements also poses some additional difficulties. As there is no longer a requirement of continuity across element edges, the level set interface can be discontinuous and thus the distance from a given point to the interface can become ill-defined. Similarly, the sign of the level set function at each degree of freedom for a given node will not necessarily be well-defined, particularly if that given node is near to the interface. Both of these potential issues will either cause strong discontinuities to develop in the level set function, ultimately leading to regions where the signed distance property is not satisfied, or lead to a smoothing

of the level set function, which will cause erroneous movement in the position of the interface post-reinitialisation. Lastly, when using a geometric reinitialisation method, the approximation of the interface on each element will be at most linear, and any benefits arising from the high-order approximations possible through the use of DG methods will be surrendered each time the reinitialisation routine is called.

One of the most popular methods of reinitialisation is a PDE based, pure reinitialisation method, introduced by Sussman *et al.* in 1994, [129], most often referred to as the *Hyperbolic Reinitialisation* method. This method involves solving the following hyperbolic PDE

$$\frac{\partial \tilde{\phi}(\mathbf{x}, t_r)}{\partial t_r} - \text{sign}(\tilde{\phi}^0) \left(1 - |\nabla \tilde{\phi}(\mathbf{x}, t_r)|\right) = 0, \quad (4.3)$$

where t_r denotes pseudotime as related to the reinitialisation problem, the steady state solution to which will be achieved once the level set function provides a sufficient approximate solution to the Eikonal equation, (3.4). The multiplication by the sign of the pre-reinitialisation level set function, $\tilde{\phi}^0$, acts as a weak Dirichlet boundary condition on the interface of the level set function, and thus the solution will be a signed distance function to the interface, $\Gamma(\tilde{\phi}^0)$.

The main issue with the Hyperbolic Reinitialisation method is that the (potentially poor) characteristics of the original level set function, $\tilde{\phi}^0$, can be propagated during the reinitialisation process. This most often presents as a ‘smearing’ of the level set interface [130], or in other words erroneous movement of the level set interface during reinitialisation. Mousavi [131], presented a solution to (4.3), discretised spatially using a DG method and temporally using a third-order Runge-Kutta scheme. In that work, Mousavi outlines clearly the difficulties encountered in trying to produce a stable solution using these methods of discretisation; the results presented show that at some point in pseudotime the solution will always gradually begin to diverge. Independent work done by the author of this thesis, instead using an explicit Euler discretisation in time, found a similar issue when trying to solve the Hyperbolic Reinitialisation problem using a DG discretisation in space. Mousavi [131] found that it was possible to create a method which was practically viable by including an artificial viscosity term, sufficiently smoothing the signum function and using a severe time step restriction. Such a solution to the reinitialisation problem isn’t ideal however, as a large number of iterations are required to return a signed distance function everywhere in the domain, which could be considered prohibitively expensive. Karakus *et al.* [132], found similar issues with the Hyperbolic Reinitialisation method and attempted to overcome these potential shortcomings by taking advantage of the high level of parallelisation possible with DG methods to speed up the computation of the resulting reinitialisation method.

Gomes and Faugeras [133], demonstrated that the solution to a Hamilton-Jacobi equation, such as the level set evolution equation, would not in general be a signed distance function. As such, they proposed modifying the evolution equation as follows

$$\frac{\partial \phi(\mathbf{x}, t_e)}{\partial t_e} = b(\mathbf{x} - \phi(\mathbf{x}, t_e) \nabla \phi(\mathbf{x}, t_e)), \quad (4.4)$$

such that it was no longer a Hamilton-Jacobi equation, and thus developed an evolution equation

the solution of which would be a signed distance function. This became the first all-in-one type method. Whilst theoretically such a formulation should force the level set function to maintain its signed distance properties, in practice it was observed that once discretised there could still be a drift in the level sets leading to a loss of the signed distance property over time [134]. This idea however, prompted other all-in-one type methods whereby the evolution equation is modified to include a signed distance constraint, such that the evolution problem and the Eikonal equation are solved simultaneously. For example, Weber *et al.* [134], set up the level set evolution problem as an optimisation problem driven by an error functional which both minimises deviations in the desired interface movement and also deviations from the signed distance property. A similar solution was presented by Li *et al.* [135] whereby the level set evolution problem was reframed as an optimisation problem with an energy driving the evolution and a penalty term restricting deviation from a signed distance function. This lead to a formulation of the evolution equation, which Li *et al.* later named, Distance Regularised Level Set Evolution (DRLSE), [136], which can be stated as

$$\frac{\partial \phi(\mathbf{x}, t_e)}{\partial t_e} = \underbrace{b|\nabla \phi(\mathbf{x}, t_e)|}_{\text{advection term}} + \underbrace{\gamma_S \nabla \cdot \left(\nabla \phi(\mathbf{x}, t_e) - \frac{\nabla \phi(\mathbf{x}, t_e)}{|\nabla \phi(\mathbf{x}, t_e)|} \right)}_{\text{signed distance constraint}}, \quad (4.5)$$

where γ_S is a penalty parameter.

Basting and Kuzmin [137], took the distance regularisation part of the DRLSE, and considered it as a pure reinitialisation problem; which can be stated as

$$\frac{\partial \tilde{\phi}(\mathbf{x}, t_r)}{\partial t_r} = \nabla \cdot \left(\nabla \tilde{\phi}(\mathbf{x}, t_r) - \frac{\nabla \tilde{\phi}(\mathbf{x}, t_r)}{|\nabla \tilde{\phi}(\mathbf{x}, t_r)|} \right), \quad (4.6)$$

which is a parabolic PDE. Basting [137] then reformulated the problem as a quasilinear elliptic PDE to be solved iteratively by simplifying out the time derivative, and also including an appropriate boundary condition, which can be stated as

$$\nabla \cdot \left(\nabla \tilde{\phi}(\mathbf{x}) - \frac{\nabla \tilde{\phi}(\mathbf{x})}{|\nabla \tilde{\phi}(\mathbf{x})|} \right) + \gamma_D \tilde{\phi}(\mathbf{x}) = 0, \quad (4.7)$$

where γ_D is a penalty parameter, used to enforce the Dirichlet boundary condition on the level set interface.

A novel yet similar approach was developed by Parolini [138], whereby the level set function on elements intersected by level set interface could be computed exactly given that the elements were both discontinuous and linear. This would be achieved by solving on each intersected element

$$\nabla \tilde{\phi}_h(\mathbf{x}) = \frac{\nabla \tilde{\phi}_h^0(\mathbf{x})}{|\nabla \tilde{\phi}_h^0(\mathbf{x})|}. \quad (4.8)$$

A continuous approximation of the level set function along those elements intersected by the interface could then be generated by L^2 projecting the solution onto a continuous linear finite element space. This approach is called the *local interface projection* method. The solution to the

local interface projection on the band of elements cut by the interface is then used as a boundary condition to solve one of the other reinitialisation methods to generate a signed distance function in the far-field region (i.e in the set of elements which aren't intersected by the level set interface). For example, Parolini uses the Hyperbolic Reinitialisation method to reinitialise the level set function in the far-field. The utility of the work done by Parolini in the context of this thesis is limited, as their work required continuous and linear finite elements. The idea of local interface projection however, has been recently extended to high-order DG discretisations by Zhang and Yue, [121]. By using a DG discretisation Zhang and Yue solve directly for the gradient of the level set function, and recover the solution using the known position of level set interface pre-reinitialisation, thus accurately preserving the position of the interface. In the far-field region, Zhang and Yue use a DG spatial discretisation of the Hyperbolic Reinitialisation problem, (4.3), to recover the signed distance function. From their results it can be seen that some of the issues described earlier seem to persist even when enforcing the boundary condition on the interface using the local interface projection method. For example it was necessary to develop a novel numerical flux to ensure stability, a limiter was required in the case that the interface was singular, a sufficiently smooth signum function was required, and a severe time step restriction was also necessary to ensure stability. Furthermore, the error data (and thus the accuracy of the whole method) presented was dependent on a number of parameters, where little guidance is presented in appropriate choices for these parameters in the general case.

The works of Li [136], and Basting and Kuzmin [137], which begin by attempting to solve a minimisation problem driven by the residual to the Eikonal equation, constrained by an appropriate boundary condition is the starting point for the work to be presented in the remainder of this chapter. In fact, Equation (4.6) which is originally presented in [137], is one possible formulation of the strong form of what will be referred to as the *Parabolic Reinitialisation* method, to be considered henceforth. Whilst Basting and Kuzmin [137] never solve the Parabolic Reinitialisation problem directly, instead simplifying the problem to the quasilinear elliptic equation, (4.7), in the sections to follow a number of discretisations of both parabolic and elliptic formulations of this underlying approach to reinitialisation will be considered using a DG discretisation of the spatial operators in all cases. As well as adopting a DG approach, and presenting a number of new formulations of this approach to reinitialisation, a number of other questions needed to be addressed in order to meet the aim of developing an efficient, robust, and high-order DG based level set reinitialisation method. Specifically these questions included: how to optimally enforce boundary conditions on an implicit surface given these aims; what modifications needed to be included to ensure that experimental orders of convergence align with the theoretically optimal rates of convergence in the relevant norms; and how to reformulate the minimisation problem such that the method is stable for any initial condition (which is an issue in the works [136, 137, 139] particularly where the gradient of the initial condition is small, $|\nabla \tilde{\phi}_h^0| < 0.5$). It should be noted that during the completion of this work, another paper was published by Utz *et al.* [139], which presented a DG discretisation of Equation (4.7), and thus a similar level set reinitialisation method to be presented below, however, in that work none of these potential issues were considered.

4.2 Eikonal minimisation based reinitialisation: strong form

As first presented by Basting and Kuzmin in [137], the derivation of this type of reinitialisation equation begins by aiming to minimise the least squares residual of the Eikonal equation, (3.4), that is

$$\min_{\tilde{\phi} \in H^1(\Omega)} \left(R_1(|\nabla \tilde{\phi}(\mathbf{x})|) \right), \quad (4.9)$$

where

$$R_1(|\nabla \tilde{\phi}(\mathbf{x})|) = \int_{\Omega} P_1(|\nabla \tilde{\phi}(\mathbf{x})|) \, d\mathbf{x}, \quad (4.10)$$

and where

$$P_1(|\nabla \tilde{\phi}(\mathbf{x})|) = \frac{1}{2} \left(|\nabla \tilde{\phi}(\mathbf{x})| - 1 \right)^2. \quad (4.11)$$

With the inclusion of the Dirichlet boundary condition (4.1), it can be seen that the signed distance function to the interface minimises (4.9), as in such a case the value of the functional would be equal to zero, which is clearly a global minimum; therefore at least one minimiser to the problem exists. A well known method for minimising an objective functional is to find the steady-state solution to the gradient flow equation, see [140], that is for functional, $R_1(|\nabla \tilde{\phi}(\mathbf{x})|)$,

$$\frac{\partial \tilde{\phi}(\mathbf{x}, t_r)}{\partial t_r} + \frac{\partial R_1(\tilde{\phi}(\mathbf{x}, t_r))}{\partial \tilde{\phi}(\mathbf{x}, t_r)} = 0. \quad (4.12)$$

The notation $\frac{\partial R_1}{\partial \tilde{\phi}}$ denotes the functional derivative of $R_1(\tilde{\phi})$ with respect to the solution variable, $\tilde{\phi}$. This derivative can be computed by its relation to the first variation of the functional $R_1(\tilde{\phi})$, in the direction of the arbitrary function, v , that is (dropping the dependency on \mathbf{x} and t_r for clarity)

$$\begin{aligned} \int_{\Omega} \frac{\partial R_1(\tilde{\phi})}{\partial \tilde{\phi}} v \, d\mathbf{x} &= \left. \frac{dR_1(\tilde{\phi} + \epsilon v)}{d\epsilon} \right|_{\epsilon=0} \\ &= \left. \frac{d}{d\epsilon} \int_{\Omega} \frac{1}{2} \left(|\nabla(\tilde{\phi} + \epsilon v)| - 1 \right)^2 \, d\mathbf{x} \right|_{\epsilon=0}, \\ &= \left. \frac{d}{d\epsilon} \int_{\Omega} \frac{1}{2} \left(|\nabla \tilde{\phi} + \epsilon \nabla v| - 1 \right)^2 \, d\mathbf{x} \right|_{\epsilon=0}, \\ &= \left. \frac{d}{d\epsilon} \int_{\Omega} \frac{1}{2} \left(|\nabla \tilde{\phi} + \epsilon \nabla v|^2 - 2|\nabla \tilde{\phi} + \epsilon \nabla v| + 1 \right) \, d\mathbf{x} \right|_{\epsilon=0}, \\ &= \int_{\Omega} \left(\nabla \tilde{\phi} \cdot \nabla v + \epsilon \nabla v \cdot \nabla v - \frac{\nabla \tilde{\phi} \cdot \nabla v + \epsilon \nabla v \cdot \nabla v}{|\nabla \tilde{\phi} + \epsilon \nabla v|} \right) \, d\mathbf{x} \Big|_{\epsilon=0}, \\ &= \int_{\Omega} \left(\nabla \tilde{\phi} \cdot \nabla v - \frac{\nabla \tilde{\phi} \cdot \nabla v}{|\nabla \tilde{\phi}|} \right) \, d\mathbf{x}. \end{aligned} \quad (4.13)$$

Using the more standard notation, this is equivalent to the Gateaux derivative of the functional, $R_1(\tilde{\phi})$, and can therefore be stated

$$\delta R_1(\tilde{\phi}, v) = \int_{\Omega} d_1(|\nabla \tilde{\phi}|) \nabla \tilde{\phi} \cdot \nabla v \, d\mathbf{x}, \quad (4.14)$$

where, $d_1(|\nabla\tilde{\phi}|)$, denotes a nonlinear diffusion functional, which can be stated in this case as

$$d_1(|\nabla\tilde{\phi}|) = 1 - \frac{1}{|\nabla\tilde{\phi}|}. \quad (4.15)$$

Combining the above gradient flow with the homogeneous Dirichlet boundary condition on the level set interface, (4.1), and an appropriate Neumann boundary condition on the natural boundary leads to a strong formulation of the reinitialisation problem which can be expressed as

$$\begin{aligned} \frac{\partial\tilde{\phi}(\mathbf{x}, t_r)}{\partial t_r} + \nabla \cdot \left(d_1(|\nabla\tilde{\phi}(\mathbf{x}, t_r)|) \nabla\tilde{\phi}(\mathbf{x}, t_r) \right) &= 0, \quad \mathbf{x} \in \Omega, \\ \tilde{\phi}(\mathbf{x}, t_r) &= 0, \quad \mathbf{x} \in \Gamma(\tilde{\phi}^0), \\ d_1(|\nabla\tilde{\phi}(\mathbf{x}, t_r)|) \nabla\tilde{\phi}(\mathbf{x}, t_r) \cdot \hat{\mathbf{n}} &= 0, \quad \mathbf{x} \in \partial\Omega. \end{aligned} \quad (4.16)$$

The first equation forming (4.16) is a parabolic equation, with quasilinear diffusion. As such the formulation, (4.16), that is where the diffusion is defined by Equation (4.15), i.e. $d_1(|\nabla\tilde{\phi}|)$, will henceforth be referred to as the *Eikonal Minimising Parabolic Reinitialisation* method. As the diffusion will be positive where $|\nabla\tilde{\phi}| > 1$ and negative where $|\nabla\tilde{\phi}| < 1$, the unique viscosity solution is a signed distance function satisfying the equilibrium point $|\nabla\tilde{\phi}| = 1$, [137]. The signed distance function which is the solution to (4.16) is uniquely defined by the homogeneous Dirichlet boundary condition along the pre-reinitialisation level set interface, $\Gamma(\tilde{\phi}^0)$. The third equation forming, (4.16), comprises a nonlinear Neumann boundary condition on the domain boundary by which the gradient of the solution at the domain boundary must also satisfy the Eikonal equation.

To get the elliptic formulation of this problem as in [127, 137, 139], one simply makes the assumption that the steady-state solution will be found when the transient term is zero, that is

$$\frac{\partial\tilde{\phi}(\mathbf{x}, \infty)}{\partial t_r} = 0. \quad (4.17)$$

This leads to the generic strong formulation of the *Eikonal Minimising Elliptic Reinitialisation* method which can be stated

$$\begin{aligned} \nabla \cdot \left(d_1(|\nabla\tilde{\phi}(\mathbf{x})|) \nabla\tilde{\phi}(\mathbf{x}) \right) &= 0, \quad \mathbf{x} \in \Omega, \\ \tilde{\phi}(\mathbf{x}) &= 0, \quad \mathbf{x} \in \Gamma(\tilde{\phi}^0(\mathbf{x})), \\ d_1(|\nabla\tilde{\phi}(\mathbf{x})|) \nabla\tilde{\phi}(\mathbf{x}) \cdot \hat{\mathbf{n}} &= 0, \quad \mathbf{x} \in \partial\Omega. \end{aligned} \quad (4.18)$$

4.3 Eikonal Minimising Elliptic Reinitialisation method

It is simpler to begin by discussing the elliptic formulation of the reinitialisation method as it does not include the requirement for temporal discretisation. Also, it is more chronologically accurate as the work on the Elliptic Reinitialisation method was completed first, before considering the parabolic formulation. For the remainder of this section, the solution variable to the reinitialisation problem does not evolve in time, and thus is only a function of space i.e. $\tilde{\phi} = \tilde{\phi}(\mathbf{x})$. Furthermore where appropriate, when discussing the level set function $\tilde{\phi}(\mathbf{x})$, the

dependence on \boldsymbol{x} is dropped for conciseness.

4.3.1 Picard linearisation and spatial discretisation

Equation (4.18) is a quasilinear diffusion equation, with a homogeneous Dirichlet boundary condition on the level set interface, and a ‘homogeneous’ quasilinear Neumann boundary condition, and therefore in order to return a non-zero solution it is necessary to apply an appropriate linearisation. In this work, a Picard linearisation is applied to the terms which are nonlinear with respect to, $\nabla \tilde{\phi}$, which allows one to rewrite the diffusion part of the strong formulation (4.18) as

$$\nabla \cdot \nabla \tilde{\phi}^m = \nabla \cdot \frac{\nabla \tilde{\phi}^{m-1}}{|\nabla \tilde{\phi}^{m-1}|} \quad \text{in } \Omega, \quad (4.19)$$

where the superscript denotes the m^{th} iteration. The linearised strong form, can then be discretised spatially using the SIPG method which leads to a variational formulation which can be stated as; find $\tilde{\phi}_h^m \in V_{\mathbf{p}}(\mathcal{T})$, as $m \rightarrow \infty$, such that the following weak form statement of equilibrium is satisfied

$$B_{\text{ER}}(\tilde{\phi}_h^m, v_h) = J_{\text{ER},1}(|\tilde{\phi}_h^{m-1}|; \tilde{\phi}_h^{m-1}, v_h), \quad \forall v_h \in V_{\mathbf{p}}(\mathcal{T}), \quad (4.20)$$

where the bilinear form, $B_{\text{ER}}(\tilde{\phi}, v)$, is defined as

$$B_{\text{ER}}(\tilde{\phi}, v) = \left(\nabla \tilde{\phi}, \nabla v \right)_{\mathcal{T}} - \left\langle \{\{\nabla \tilde{\phi}\}\}, \llbracket v \rrbracket \right\rangle_{S(\mathcal{T})} - \left\langle \llbracket \tilde{\phi} \rrbracket, \{\{\nabla v\}\} \right\rangle_{S(\mathcal{T})} + \mu \left\langle \llbracket \tilde{\phi} \rrbracket, \llbracket v \rrbracket \right\rangle_{S(\mathcal{T})}, \quad (4.21)$$

the form $J_{\text{ER},n}(|\nabla \tilde{\phi}|; \tilde{\phi}, v)$ is defined as

$$J_{\text{ER},n}(|\nabla \tilde{\phi}|; \tilde{\phi}, v) = \left((1 - d_n(|\nabla \tilde{\phi}|)) \nabla \tilde{\phi}, \nabla v \right)_{\mathcal{T}} - \left\langle \left\{ \left\{ (1 - d_n(|\nabla \tilde{\phi}|)) \nabla \tilde{\phi} \right\} \right\}, \llbracket v \rrbracket \right\rangle_{S(\mathcal{T})}. \quad (4.22)$$

The subscript n associated with the form $J_{\text{ER},n}$ is used to differentiate between various diffusion functions which can be used to define the Elliptic Reinitialisation problem; in this case, $n = 1$, corresponding to the diffusion function, d_1 , as defined in Equation (4.15), however, a discussion into other possible diffusion functions will be presented in Section 4.3.4. The penalty parameter, μ , in Equation (4.21) is chosen as, $\mu = 10p^2/h$, see Section 2.3.1.1 for details. One further note at this stage, is that the term on the RHS is integrated by parts which is why there is a volume and surface integral in Equation (4.22), however, the additive zero terms which enforce continuity, again see Section 2.3.1.1, need only be applied to the solution variable and therefore only appear on the LHS.

4.3.2 Integration on an immersed implicit interface

Before one can discuss the methods for imposing a boundary condition on an immersed implicit interface, one requires a method for computing the line integral of a function over that interface. There are three general approaches found in the literature: explicit reconstruction of the interface through mesh refinement [141]; implicit reconstruction of the interface using an approximate Dirac delta function such as in the original immersed boundary method, [142]; and methods which generate a new quadrature rule over the area of an element (for 2D problems), which is

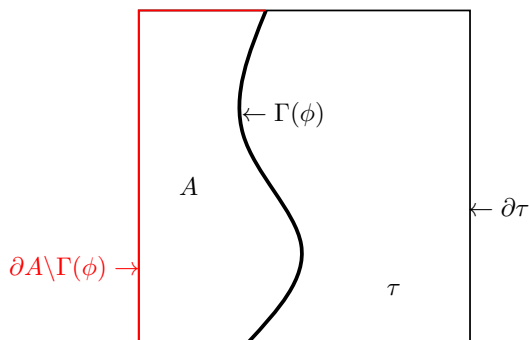


Figure 4.1: A quadrilateral element, τ , intersected by a level set interface, $\Gamma(\phi)$, which divides the element into two subdomains A and $\tau \setminus A$.

equivalent to integrating an arbitrary function over the implicit interface [143, 144].

Methods involving r -adaptivity (that is methods which allow for the movement of element nodes) to explicitly reconstruct the interface are not appropriate in the context of this work, as the Eulerian nature of the level set method allows one to take advantage of the use of simple Cartesian meshes. Methods of this type also suffer from extreme computational expense, especially when the desired level of accuracy is high. Methods involving the use of an approximate Dirac delta function, allow one to replace the line integral over the interface with an equivalent surface integral weighted by the Dirac delta function (again for the 2D problems to be considered in this work). Whilst this method is simple to implement, and has found use in other works, even prompting research into high-order approximations of the delta function [145], the method depends on the global cancellation of errors over the domain, which will have limited accuracy when working with level set functions which are only piecewise continuous. The final group of methods, however, are able to demonstrably provide arbitrarily high-order elementwise approximations of integrals on implicit interfaces. The preferred method of the author is one such method which was presented by Müller *et al.* in [144], and is henceforth referred to throughout the text as *Müller's method*.

Perhaps the easiest way to explain Müller's method is by analogue with the method by which one implicitly reconstructs a level set interface using a Dirac delta function which is mentioned previously and with which the reader might be more familiar. In this way, one can approximate the integral of an arbitrary function u over an interface, $\Gamma(\phi)$, implied by a level set function as follows

$$\int_{\Gamma(\phi)} u \, ds \approx \int_{\tau} u \delta_s(\phi) \, d\mathbf{x} \quad (4.23)$$

where $\delta_s(\cdot)$ denotes an appropriate smooth approximation of the Dirac delta function, and the domains of integration are as presented in Figure 4.1. That is one can transform the surface integral over the interface, to a volume integral over the domain (τ in this case) where the function u has a weighting at each integration point, the value of which will be large near to the interface (where $\phi = 0$) and which quickly decays to zero elsewhere. Müller's method acts much in this vein, allowing one to transform a surface integral such as that stated in the LHS of

(4.23) to an integral over the volume of an element by computing an appropriate set of weights at each of a set of quadrature points over the volume of that element. How this is achieved, however, is quite different.

The first piece of technology needed to compute these new weights in Müller's method is a technique for computing quadrature rules (i.e. a set of points and weights). In [144], Müller *et al.* do this by computing the solution of a system of moment fitting equations, [146]. That is for a set of N_I integrals of the set of functions $\{u\}_{i=1}^{N_I}$ over a domain, τ , one can solve the following

$$\begin{bmatrix} u_1(\boldsymbol{\xi}_1) & \cdots & u_1(\boldsymbol{\xi}_{N_\xi}) \\ \vdots & \ddots & \vdots \\ u_{N_I}(\boldsymbol{\xi}_1) & \cdots & u_{N_I}(\boldsymbol{\xi}_{N_\xi}) \end{bmatrix} \begin{Bmatrix} w_1 \\ \vdots \\ w_{N_\xi} \end{Bmatrix} = \begin{Bmatrix} \int_\tau u_1 ds \\ \vdots \\ \int_\tau u_{N_I} ds \end{Bmatrix}, \quad (4.24)$$

where $\boldsymbol{\xi}$ are the abscissae of the new quadrature rule, N_ξ is the number of quadrature points (a number which can be chosen by the user), and w are the weights of the new quadrature rule. It is assumed that the set of functions $\{u\}_{i=1}^{N_I}$ comprises a polynomial basis of \mathbb{R}^2 of degree p . The maximum order of the basis functions, determines the number of equations, N_I , to be solved and it is noted by Müller *et al.* [144] that care should be taken to ensure that the number of quadrature points, N_ξ , is chosen such that the resulting linear system is underdetermined, i.e. $N_\xi > N_I$. The quadrature rule generated by (4.24) for the set of functions $\{u\}_{i=1}^{N_I}$ over the given domain, can then be used to compute the integral of any function over the domain provided that function can be sampled at the quadrature points. In general moment fitting equations such as those stated in (4.24) are nonlinear, in that both the weights and the abscissae of the quadrature rule are to solved for, however, Müller *et al.* instead choose to fix the set of points such that they align with the standard 2D Gauss quadrature abscissae of appropriate order. By fixing the position of the points, $\boldsymbol{\xi}$, the system becomes linear and only the weights, w , need to be solved for on each element which allows for the efficiency of which Müller's method is capable.

In order to generate this set of weights then by which one can compute the integral of any function over a given implicit interface using the standard Gauss quadrature abscissae over the volume of a domain, one needs to compute a set of integrands over the interface, to form the RHS of a set of moment fitting equations. This is difficult as the position of the interface is unknown by virtue of the implicit nature of the level set method. As such in [144], Müller *et al.* consider a set of divergence free basis functions, $\{\boldsymbol{\beta}'_i\}_{i=1}^{N_I}$, by which one can transform the integral over the unknown position of the interface, to an equivalent integral over the known boundary of the domain using the divergence theorem. This can be demonstrated as follows

$$\begin{aligned} \int_{\Gamma(\phi)} \boldsymbol{\beta}'_i \cdot \mathbf{n}_\phi ds &= \int_{\partial A} \boldsymbol{\beta}'_i \cdot \hat{\mathbf{n}} ds - \int_{\partial A \setminus \Gamma(\phi)} \boldsymbol{\beta}'_i \cdot \hat{\mathbf{n}} ds, \\ &= \int_A \nabla \cdot \boldsymbol{\beta}'_i dx - \int_{\partial A \setminus \Gamma(\phi)} \boldsymbol{\beta}'_i \cdot \hat{\mathbf{n}} ds, \\ &= 0 - \int_{\partial A \setminus \Gamma(\phi)} \boldsymbol{\beta}'_i \cdot \hat{\mathbf{n}} ds, \\ &= - \int_{\partial \tau} \theta(\phi) \boldsymbol{\beta}'_i \cdot \hat{\mathbf{n}} ds, \end{aligned} \quad (4.25)$$

where $\theta(\cdot)$ denotes the Heaviside function, and \mathbf{n}_ϕ denotes the normal of the level set function which can be computed as follows

$$\mathbf{n}_\phi = \frac{\nabla\phi}{|\nabla\phi|}. \quad (4.26)$$

It should be noted again that the notation in (4.25) for the domains of integration are in reference to Figure 4.1.

In order to generate the desired set of weights then, one simply needs to compute the solution to the following set of moment fitting equations

$$\begin{bmatrix} \beta'_1(\boldsymbol{\xi}_1) \cdot \mathbf{n}_\phi(\boldsymbol{\xi}_1) & \cdots & \beta'_1(\boldsymbol{\xi}_{N_\xi}) \cdot \mathbf{n}_\phi(\boldsymbol{\xi}_{N_\xi}) \\ \vdots & \ddots & \vdots \\ \beta'_{N_I}(\boldsymbol{\xi}_1) \cdot \mathbf{n}_\phi(\boldsymbol{\xi}_1) & \cdots & \beta'_{N_I}(\boldsymbol{\xi}_{N_\xi}) \cdot \mathbf{n}_\phi(\boldsymbol{\xi}_{N_\xi}) \end{bmatrix} \begin{Bmatrix} w_1 \\ \vdots \\ w_{N_\xi} \end{Bmatrix} = \begin{Bmatrix} - \int_{\partial\tau} \theta(\tilde{\phi}_h^0) \beta'_1 \cdot \hat{\mathbf{n}} \, ds \\ \vdots \\ - \int_{\partial\tau} \theta(\tilde{\phi}_h^0) \beta'_{N_I} \cdot \hat{\mathbf{n}} \, ds \end{Bmatrix}. \quad (4.27)$$

The basis functions, β , are chosen as the monomial basis functions, where the derivatives, β' , are orthonormalised using a Gram-Schmidt procedure. A method for generating the values of these bases can be found in Appendix A. For the remainder of this thesis unless explicitly stated otherwise, for a mesh of maximum order, p_{\max} , the order of the bases, β' , will thus be chosen as $p_{\max} + 1$ for all elements, in order to ensure the interface is maximally resolved given its current representation by the mesh. This means that the number of degrees of freedom on each element can be computed as follows, $N_I = (p_{\max} + 2)(p_{\max} + 5)/2$. Then, as the abscissae are chosen to align the standard 2D Gauss quadrature points, to ensure that $N_\xi > N_I$, $N_\xi = (p_{\max} + 3)^2$ quadrature points are used to compute the integrals in (4.27).

Müller's method as presented in [144], is as stated above the preferred integration method of the author, however, there are two modifications to the method as stated in the article [144] which were found to be required to ensure accuracy and stability. Firstly, it was found that the accuracy of this method depends heavily on the accuracy with which one is able to compute the integral terms on the RHS of (4.27). The Heaviside function, $\theta(\tilde{\phi})$, in each of the integrals is present such that the integral is computed only along the part of the edge where the level set function is positive, $\tilde{\phi} > 0$. When using a standard 1D Gauss quadrature along element edges, the discontinuity present in the Heaviside function is smoothed to such an extent that it becomes difficult to predict whether a given quadrature rule will be sufficient to ensure that the method is sufficiently accurate, without using a (potentially prohibitively) high-order quadrature rule. As such for edges intersected by the interface, a Newton/bisection method is used to find the intersection point(s) and a standard quadrature rule is used to integrate over this newly defined interval, with stopping criterion, $|\tilde{\phi}| < 1e - 15$, i.e. when the root is smaller than the stated tolerance. Secondly, as the number of quadrature points is chosen to ensure that the system is underdetermined, the linear system will likely be rank deficient and ill-conditioned. Whilst it is noted by Muller *et al.* that any appropriate least squares solver will suffice, here the numerically

stable singular value decomposition approach is used (whereby any singular values, S , deemed too small, that is $S < \max(S)/10^{12}$, are removed to further improve stability) which is a slight difference to the method as stated in [144]. As a final note, whilst generally robust, Müller’s method is problem dependent and small perturbations in the relative position between the mesh and the immersed surface will have an influence on the accuracy for a given problem.

4.3.3 Boundary conditions on implicit interfaces

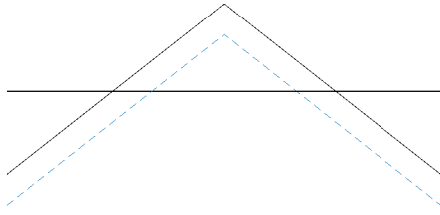
To enforce a Dirichlet boundary condition on an implicit surface is not trivial. The literature highlights four main approaches for the imposition of Dirichlet boundary conditions on implicit surfaces; the penalty method [38], Nitsche’s method [37], the method of Lagrange multipliers [147], and methods involving enrichment or modification of shape functions, for example [148].

In the works by Basting and Kuzmin [137] and Utz *et al.* [139], on a similar PDE based reinitialisation method, the Dirichlet boundary condition on the level set interface is enforced using a penalty method. As such the weak formulation would be stated as, find $\tilde{\phi}_h^m \in V_{\mathbf{p}}(\mathcal{T})$, as $m \rightarrow \infty$ such that

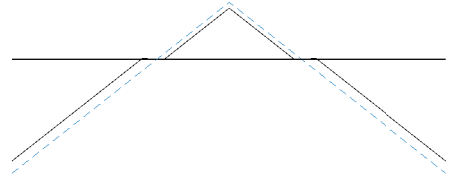
$$B_{\text{ER}}(\tilde{\phi}_h^m, v_h) + \underbrace{\gamma_D \langle \tilde{\phi}_h^m, v_h \rangle_{\Gamma(\tilde{\phi}^0)}}_{\text{penalty term}} = J_{\text{ER},1}(|\nabla \tilde{\phi}_h^{m-1}|; \nabla \tilde{\phi}_h^{m-1}, v_h), \quad \forall v_h \in V_{\mathbf{p}}(\mathcal{T}), \quad (4.28)$$

where γ_D is a penalty parameter, henceforth referred to as the *interface penalisation parameter*.

The main advantage of penalty methods is their simplicity, in this work however, difficulty was encountered in deciding how to choose an appropriate value of the interface penalisation parameter, γ_D . Babuška *et al.* [149], note that when using a penalty method, that if the value of the penalty parameter is chosen to be too large or too small, it can significantly decrease the accuracy of the underlying method. For the Eikonal Minimising Reinitialisation methods, choosing the interface penalty parameter outside the range of admissible values can lead to two possible issues, which can be demonstrated through a simple numerical example. An initial level set function, $\tilde{\phi}^0 = 1.5|x| + 1$, is L^2 projected onto a mesh of 40 square elements on $\Omega = (-2, 2) \times (0, 0.4)$, such that $h = 0.2$. This function is then reinitialised using the formulation of the Eikonal Minimising Elliptic Reinitialisation method (4.28). The solution is considered to have converged when $|\tilde{\phi}^m - \tilde{\phi}^{m-1}| < 10^{-8}$; that is when the relative change in the level set function between the two most recent iterations is smaller than a tolerance. It should be noted here that throughout this section, as the interface of the original level set function, $\Gamma(\tilde{\phi}^0)$, is in general immersed within an element, all integrals computed over a level set interface will be computed using Müller’s method, the details of which were discussed in Section 4.3.2. The results of this experiment for two values of the penalty parameter are displayed in Figure 4.2. If the penalty parameter is too small then there is no longer a unique solution and Equation (4.28) holds such that the solution found satisfies the Eikonal equation, but the level set function is no longer sufficiently constrained as a rigid body in space, which appears as a movement of the interface as can be seen in Figure 4.2(a). If the value of the interface penalisation parameter is too large, there will be boundary locking, [150], in elements intersected by the interface, as is demonstrated in Figure 4.2(b).

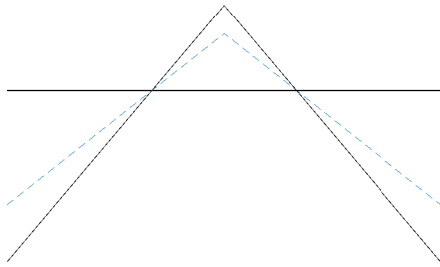


(a) Converged solution with $\gamma_D = 0$.

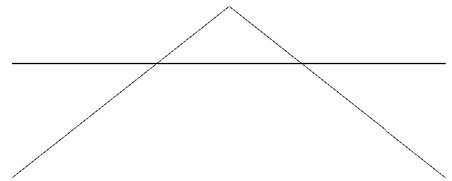


(b) Converged solution with $\gamma_D = 10^6$.

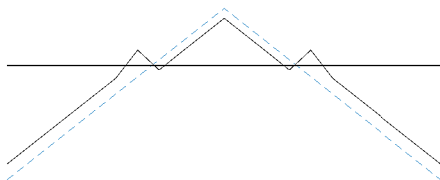
Figure 4.2: Effect of the value of the penalty parameter, γ_D , on the solution of a reinitialisation problem at the level set interface. The solid line shows the level set function, the dashed line shows the analytical solution, and the horizontal line shows the $\phi = 0$ plane.



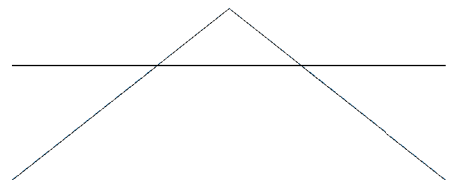
(a) Pre-reinitialisation level set function.



(b) Converged solution using linear elements with $\gamma_D = 50$.



(c) Converged solution using quintic elements with $\gamma_D = 1250$.



(d) Converged solution using quintic elements with $\gamma_D = 50$.

Figure 4.3: Examples showing problem dependency of the penalty parameter, γ_D . The solid line shows the level set function, the dashed line shows the analytical solution, and the horizontal line shows the $\phi = 0$ plane.

In [139], evidence is presented which lends support to the idea that an appropriate choice for the value of the interface penalisation parameter for a given mesh, is equal to the discontinuity penalisation parameter, μ , that is, $\gamma_D = \mu$ (see Section 2.3.1.1 for information concerning the choice of μ). It can be observed that the interface penalisation parameter is problem dependent, however, it is not necessarily apparent that it is related to the mesh size in the same way as the discontinuity penalisation parameter. For example, repeating the numerical experiment from the previous paragraph, with a mesh of linear elements, the interface penalisation parameter would therefore be computed, $\gamma_D = 10p^2/h = 50$. As evidenced at a glance by the solution in Figure 4.3(b), choosing the penalty parameter in this way is appropriate in this case. Repeating the experiment once again but this time increasing the order of the elements to $p = 5$ causes an increase in this value to $\gamma_D = 1250$; Figure 4.3(c) shows that such a value is too large and causes locking/spurious oscillations in the elements intersected by the level set interface and therefore is not an appropriate choice. However, repeating the experiment a third time, again using quintic elements, but choosing the interface penalty parameter as, $\gamma_D = 50$, allows one to return a solution which no longer displays locking at the boundary as shown in Figure 4.3(d). Without presenting the evidence, the same is true when changing the number of elements used to discretise the problem. This implies that the problem itself has a significant (and difficult to quantify) influence on the range of admissible values for the interface penalisation parameter. The difficulty in identifying *a priori* the admissible range of values for the interface penalisation parameter for a given problem led to the exploration of other possible methods for the imposition of a Dirichlet boundary condition on an implicit surface.

Nitsche's method is similar to the penalty method in that there is a penalty term which imposes the prescribed value on the boundary. Without re-presenting the evidence, the same arguments against using the penalty method described above were found to also be true of Nitsche's method when applied to the implicit interface. The methods involving the modification of the shape functions require *a priori* knowledge of the position of the interface, whereas the methodology here deals with evolving and implied interfaces only, and therefore methods such as these are also not appropriate in the context of this work.

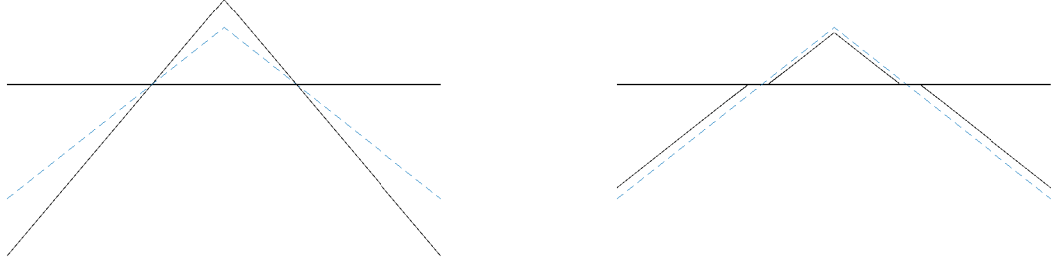
The method of Lagrange multipliers involves the reformulation of the weak form (4.20) such that a new unknown, the Lagrange multiplier, λ_D , is to be solved for, in addition to the level set function, $\tilde{\phi}$, which constrains the level set function along the level set interface. The weak form of the Eikonal Minimising Elliptic Reinitialisation problem can thus be reformulated: find $\tilde{\phi}_h^m \in V_p(\mathcal{T})$ and $\lambda_D \in \mathfrak{L}$, as $m \rightarrow \infty$ such that

$$B_{\text{ER}}(\tilde{\phi}_h^m, v_h) + \underbrace{\langle \lambda_D, v_h \rangle_{\Gamma(\tilde{\phi}^0)}}_{\text{LM term}} = J_{\text{ER},1}(|\nabla \tilde{\phi}_h^{m-1}|; \nabla \tilde{\phi}_h^{m-1}, v_h), \quad \forall v_h \in V_p(\mathcal{T}), \quad (4.29)$$

and

$$\langle \tilde{\phi}_h^m, \zeta \rangle_{\Gamma(\tilde{\phi}^0)} = 0, \quad \forall \zeta \in \mathfrak{L}. \quad (4.30)$$

One of the difficulties of using the method of Lagrange multipliers, is choosing the correct interpolation space, \mathfrak{L} , for the Lagrange multipliers. One natural choice is choosing the space,



(a) Pre-reinitialisation level set function.

(b) Converged Solution.

Figure 4.4: Effect of using too large of an interpolation space for the Lagrange multipliers to enforce a Dirichlet boundary condition. The solid line shows the level set function, the dashed line shows the analytical solution, and the horizontal line shows the $\phi = 0$ plane.

\mathfrak{L} , as follows

$$\mathfrak{L}(\mathcal{T}_\Gamma) = \text{span}\{\mathcal{Q}_{p_\tau}(\tau)\}, \quad (4.31)$$

where

$$\mathcal{T}_\Gamma = \{\tau \in \mathcal{T} : \tau \cap \Gamma(\tilde{\phi}^0) \neq \emptyset\}, \quad (4.32)$$

that is, \mathcal{T}_Γ denotes the subset of elements in \mathcal{T} which are intersected by the level set interface, Γ . Choosing the Lagrange multiplier space in this way i.e. consisting of the same basis functions as the finite element space (see Equation 2.2), means that one needs to solve for one Lagrange multiplier per degree of freedom on any element intersected by the interface.

When choosing the Lagrange multiplier interpolation space, an appropriate choice is a space which is rich enough such that it contains the approximate solution, but not so large as to overconstrain the problem. It is a known phenomenon, [151], that boundary locking or spurious oscillations can occur when the Lagrange multiplier space is too large, as may be the case when, $V_p(\mathcal{T})$ and $\mathfrak{L}(\mathcal{T}_\Gamma)$ are chosen to be of equal order. Once again, the same numerical experiment defined earlier in this section is computed, this time using a Lagrange multiplier approach to enforce the boundary condition, with the Lagrange multiplier space defined as in (4.31). The results of this experiment can be seen in Figure 4.4, which shows that choosing the spaces $V_p(\mathcal{T})$ and $\mathfrak{L}(\mathcal{T}_\Gamma)$ to be of equal order does in fact lead to boundary locking.

In order to avoid this issue the order of the space $\mathfrak{L}(\mathcal{T}_\Gamma)$ needs to be reduced. It was observed that using a rule such as

$$\mathfrak{L}(\mathcal{T}_\Gamma) = \text{span}\{\mathcal{Q}_{p_\tau-1}(\tau)\}, \quad (4.33)$$

i.e. for a finite element space of order p , choosing the Lagrange Multiplier space, to be the space of polynomials of order $p - 1$, led to similar issues with locking and spurious oscillations. The only case where this wasn't true was choosing $\mathfrak{L}(\mathcal{T}_\Gamma)$ as the space of piecewise constant functions; which was observed to be true regardless of the order of the finite element space.

A more appropriate choice then, for any order of approximation space $V_p(\mathcal{T})$, is to reduce the order of the Lagrange multiplier space to the space of piecewise constant functions with one degree of freedom per element intersected by the interface. This can be stated as

$$\mathfrak{L}(\mathcal{T}_\Gamma) = \text{span}_{\tau \in \mathcal{T}_\Gamma} \{\mathbf{1}_\tau\}, \quad (4.34)$$

where $\mathbf{1}_\tau$ is the indicator function defined as follows

$$\mathbf{1}_\tau(\mathbf{x}) := \begin{cases} 1 & \text{if } \mathbf{x} \in \tau, \\ 0 & \text{if } \mathbf{x} \notin \tau. \end{cases} \quad (4.35)$$

This choice of space means that for each element, $\tau \in \mathcal{T}_\Gamma$, the integral of the level set function over the portion of the interface contained within that element, averages to be zero over the element. In other words, this reduction in the order of the constraint space allows some movement to occur at the interface (limited by the size of the element), which is a sufficient relaxation to remove the boundary locking observed above and allows the boundary condition to be satisfied without affecting the signed distance property.

The preferred method of the author therefore for enforcing a Dirichlet boundary condition on an implicit interface, is to use a Lagrange multiplier approach, where the Lagrange multiplier space is chosen to be the space of piecewise constant functions.

4.3.4 Modified objective functionals for the minimisation based reinitialisation problem

In order to avoid further derivations, it is noted that in general for an objective functional which could be stated

$$\min_{\tilde{\phi} \in H^1(\Omega)} \left(R_n(|\nabla \tilde{\phi}|) \right), \quad (4.36)$$

where

$$R_n(|\nabla \tilde{\phi}|) = \int_{\Omega} P_n(|\nabla \tilde{\phi}|) \, d\mathbf{x}, \quad (4.37)$$

the Gateaux derivative can be stated

$$\delta R_n(\tilde{\phi}, v) = \int_{\Omega} d_n(|\nabla \tilde{\phi}|) \nabla \tilde{\phi} \cdot \nabla v \, d\mathbf{x}, \quad (4.38)$$

with diffusion functional

$$d_n(|\nabla \tilde{\phi}|) = \frac{1}{|\nabla \tilde{\phi}|} \frac{\partial P_n(|\nabla \tilde{\phi}|)}{\partial (|\nabla \tilde{\phi}|)}. \quad (4.39)$$

One potential problem which can be seen immediately with the Eikonal minimisation based formulations (4.16) and (4.18) is that the resulting nonlinear diffusion functional, $d_1(|\nabla \tilde{\phi}|)$, becomes singular as $|\nabla \tilde{\phi}| \rightarrow 0$. This issue was also noted by authors Li [136] and Basting [137], who completed work on level set reinitialisation using a similar approach. To combat this issue these authors suggest modifying the objective functional driving the minimisation problem, (4.36), such that what is optimised is the least squares residual to the Eikonal equation

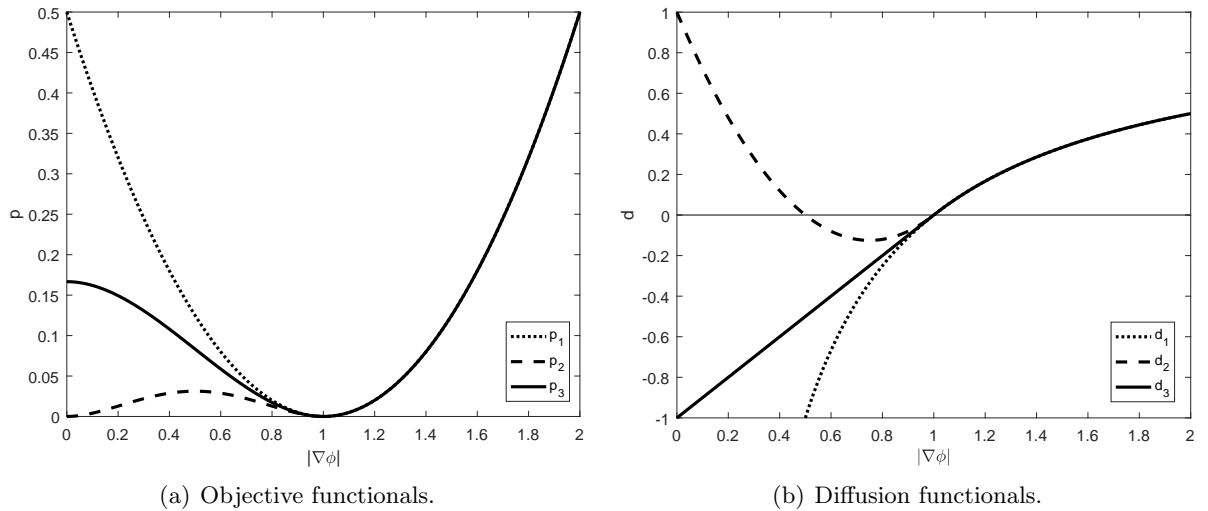


Figure 4.5: Three different objective functionals and their corresponding diffusion rates.

everywhere except in the region where $|\nabla\tilde{\phi}|$ is small. For example, [137] presents the potential functional

$$P_2(|\nabla\tilde{\phi}|) = \begin{cases} \frac{1}{2}(|\nabla\tilde{\phi}| - 1)^2 & \text{if } |\nabla\tilde{\phi}| > 1, \\ \frac{1}{2}|\nabla\tilde{\phi}|^2(|\nabla\tilde{\phi}| - 1)^2 & \text{if } |\nabla\tilde{\phi}| \leq 1, \end{cases} \quad (4.40)$$

which leads to a diffusion term

$$d_2(|\nabla\tilde{\phi}|) = \begin{cases} 1 - \frac{1}{|\nabla\tilde{\phi}|} & \text{if } |\nabla\tilde{\phi}| > 1, \\ 1 - (3|\nabla\tilde{\phi}| - 2|\nabla\tilde{\phi}|^2) & \text{if } |\nabla\tilde{\phi}| \leq 1. \end{cases} \quad (4.41)$$

Figure 4.5(a) shows a plot of the objective functional, P_2 and Figure 4.5(b) shows a plot of the associated diffusion functional, d_2 . It can be observed in Figure 4.5(a) that for the objective functional, P_2 , there are two solutions to the minimisation problem, one corresponding to the Eikonal equation, and a second at $|\nabla\tilde{\phi}| = 0$. Furthermore, it can be seen in Figure 4.5(b), that for the corresponding diffusion functional, d_2 , that where the gradient is small, i.e. $|\nabla\tilde{\phi}| < 0.5$, the diffusion is positive, which corresponds to forcing the level set function towards the solution at $|\nabla\tilde{\phi}| = 0$. This is referred to as a double well potential, in that there are two equivalent global minima. In the work done by Basting and Kuzmin, where such a modification is made, the choice makes sense as it is their desire to use a truncated level set function, where the gradient of the level set function in the far-field region should be equal to zero. In the work to be presented in this thesis such an approach is not applicable as it wouldn't satisfy the requirement for robustness in the method. For example, it is conceivable that a level set function with a shallow gradient near to the interface might need to be reinitialised, and using a double well potential could lead to the signed distance property not being satisfied in this region, as will be shown numerically presently. Li *et al.* in [136], present a similar double well potential, however the criticisms to Basting and Kuzmin's double well potential apply also in that case.

In order to overcome these issues, here a new objective functional is proposed which both avoids the singularity at $|\nabla\tilde{\phi}| = 0$ and always has negative diffusion for $|\nabla\tilde{\phi}| < 1$. One such

functional could be stated as

$$P_3(|\nabla\tilde{\phi}|) = \begin{cases} \frac{1}{2}(|\nabla\tilde{\phi}| - 1)^2 & \text{if } |\nabla\tilde{\phi}| > 1, \\ \frac{(|\nabla\tilde{\phi}|)^3}{3} - \frac{(|\nabla\tilde{\phi}|)^2}{2} + \frac{1}{6} & \text{if } |\nabla\tilde{\phi}| \leq 1, \end{cases} \quad (4.42)$$

which leads to a diffusion term

$$d_3(|\nabla\tilde{\phi}|) = \begin{cases} 1 - \frac{1}{|\nabla\tilde{\phi}|} & \text{if } |\nabla\tilde{\phi}| > 1, \\ 1 - (2 - |\nabla\tilde{\phi}|) & \text{if } |\nabla\tilde{\phi}| \leq 1. \end{cases} \quad (4.43)$$

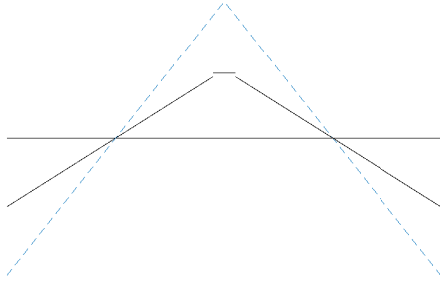
It should be stated that any function which satisfies these conditions, conceptually should suffice. Figure 4.5 shows that the objective functional, P_3 , does indeed satisfy both of these conditions.

Figure 4.6 demonstrates by example the relative performance of these different objective functionals. For this numerical experiment, an initial level set function, $\tilde{\phi}^0 = -(|x|/2) + 0.5$, is projected onto a mesh of 38 square elements on the domain $\Omega = (-2, 2) \times (0, 8/19)$ with $h = 4/19$, such that a singularity falls at the centre of the 2 central elements. Using a mesh of linear elements, both components of the gradient throughout these elements will therefore be close to zero, and everywhere else in the mesh the gradient can also be considered small, i.e. $|\nabla\tilde{\phi}| \leq 0.5$. The initial projection of the level set function can be seen in Figure 4.6(a). For these examples, the initial level set function is reinitialised using the discretisation presented in Equations (4.29) and (4.30), that is the elliptic formulation, linearised using Picard's method, using the Lagrange multiplier method to enforce the Dirichlet boundary condition. The only modification required to include any of these new diffusion functionals is to modify the RHS of the weak form as follows

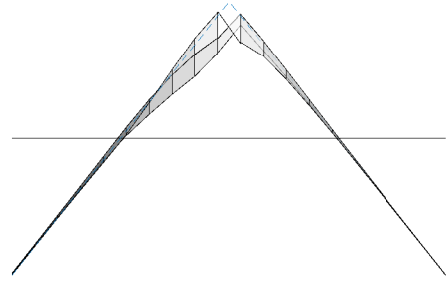
$$B_{\text{ER}}(\tilde{\phi}_h^m, v_h) + \langle \lambda_D, v_h \rangle_{\Gamma(\tilde{\phi}^0)} = J_{\text{ER},n}(|\nabla\tilde{\phi}_h^{m-1}|; \nabla\tilde{\phi}_h^{m-1}, v_h), \quad \forall v_h \in V_{\mathbf{p}}(\mathcal{T}), \quad (4.44)$$

where, $n = 1, 2, 3$, corresponds to the diffusion in Equations (4.15), (4.41) and (4.43) respectively. Again the criterion defining convergence for these numerical experiments can be stated, $|\tilde{\phi}^m - \tilde{\phi}^{m-1}| < 10^{-8}$.

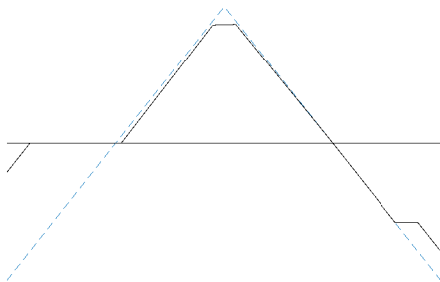
When using the Eikonal minimising objective functional, P_1 , it can be observed that the solution immediately begins to oscillate and does not converge. Figure 4.6(b) shows the level set function after 50 iterations when using P_1 . It can be seen that the attempt to correct the almost zero gradients in the centre element, leads to an overcorrection causing the level set function to twist as it tries to force the gradient back to unity, after which the solution breaks down and continues to diverge over time. Figure 4.6(c) shows the converged solution when using Basting and Kuzmin's proposed objective functional, P_2 . It can be seen that there are no longer overshoots as a result of the initial 'zero' gradients, however, some parts of the level set function converge to the solution at $|\nabla\tilde{\phi}| = 0$. Figure 4.6(d) shows the converged solution using the objective functional, P_3 . The limited diffusion for small gradients does slow convergence in regions where the gradient is small, however, it also removes any overshoots or oscillations, and thus the level set function at steady-state is congruent with the analytical solution as far as



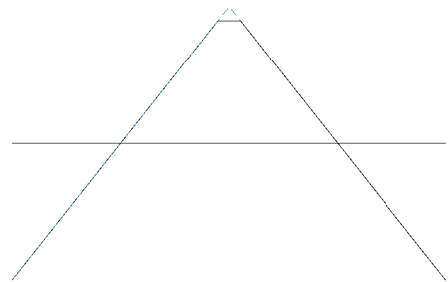
(a) Pre-reinitialisation level set function.



(b) Diverging solution after 50 iterations using objective function, P_1 .



(c) Converged solution using objective functional, P_2 .



(d) Converged solution using objective functional, P_3 .

Figure 4.6: Converged solutions to a simple problem demonstrating how each of the different objective functionals behave in the presence of small gradients. The solid line shows the level set function, the dashed line shows the analytical solution, and the horizontal line shows the $\phi = 0$ plane.

possible given the coarseness of the mesh. Therefore the preferred objective functional in this work is that defined as P_3 .

4.4 Elliptic Reinitialisation method

The discussion presented in Sections 4.3.1, 4.3.2, 4.3.3, and 4.3.4, yields a number of preferable modifications to the Eikonal Minimising Elliptic Reinitialisation method. For the remainder of this thesis the reinitialisation method referred to as the *Elliptic Reinitialisation* method (which can be contrasted to that referred to as the Eikonal Minimising Elliptic reinitialisation method), will refer to the method to be described in this section. Specifically, the objective functional defining the Elliptic Reinitialisation problem is that defined in Equation (4.42), i.e. P_3 . The resulting quasilinear diffusion equation is linearised using Picard's method, which can be stated

$$\begin{aligned} \nabla \cdot \nabla \tilde{\phi}^m &= \nabla \cdot \left((1 - d_3(|\nabla \tilde{\phi}^{m-1}|)) \nabla \tilde{\phi}^{m-1} \right), & \mathbf{x} \in \Omega, \\ \tilde{\phi}^m &= 0, & \mathbf{x} \in \Gamma(\tilde{\phi}^0), \\ d_3(|\nabla \tilde{\phi}^m|) \nabla \tilde{\phi}^m \cdot \hat{\mathbf{n}} &= 0, & \mathbf{x} \in \partial\Omega. \end{aligned} \quad (4.45)$$

The homogeneous Dirichlet boundary condition is enforced using the Lagrange multiplier approach, with an interpolation space consisting of piecewise constant functions. Discretising (4.45) using SIPG, the resulting variational formulation of the Elliptic Reinitialisation method can be stated as: find $\tilde{\phi}_h^m \in V_{\mathbf{p}}(\mathcal{T})$ and $\lambda_D \in \mathfrak{L}(\mathcal{T}_\Gamma)$, as $m \rightarrow \infty$ such that

$$B_{\text{ER}}(\tilde{\phi}_h^m, v_h) + \langle \lambda_D, v_h \rangle_{\Gamma(\tilde{\phi}^0)} = J_{\text{ER},3}(|\nabla \tilde{\phi}_h^{m-1}|; \nabla \tilde{\phi}_h^{m-1}, v_h), \quad \forall v_h \in V_{\mathbf{p}}(\mathcal{T}), \quad (4.46)$$

and

$$\langle \tilde{\phi}_h^m, \zeta \rangle_{\Gamma(\tilde{\phi}^0)} = 0, \quad \forall \zeta \in \mathfrak{L}(\mathcal{T}_\Gamma). \quad (4.47)$$

Thus choosing $\{v_j\}_{j=1}^{N_h}$ to form a basis of $V_{\mathbf{p}}$, and $\{\zeta_j\}_{j=1}^{N_{LM}}$ to form a basis of \mathfrak{L} one can write

$$\tilde{\phi}_h = \sum_{j=1}^{N_h} \tilde{\phi}_j v_j, \quad \lambda_D = \sum_{j=1}^{N_{LM}} \lambda_{D,j} \zeta_j, \quad (4.48)$$

where N_h denotes the number of solution variables per element, and N_{LM} denotes the number of Lagrange multipliers per element. The linear system for the Elliptic Reinitialisation method can thus be stated as: find $\tilde{\phi}_h^m \in V_{\mathbf{p}}(\mathcal{T})$, and $\lambda_D \in \mathfrak{L}(\mathcal{T}_\Gamma)$ as $m \rightarrow \infty$, such that

$$\begin{bmatrix} K_{\text{ER}} & \Lambda^\top \\ \Lambda & 0 \end{bmatrix} \begin{Bmatrix} \tilde{\phi}^m \\ \boldsymbol{\lambda}_D \end{Bmatrix} = \begin{Bmatrix} F_{\text{ER}} \\ 0 \end{Bmatrix}, \quad (4.49)$$

where $\tilde{\phi} = \{\tilde{\phi}_1, \tilde{\phi}_2, \dots, \tilde{\phi}_{N_h}\}^\top$ and $\boldsymbol{\lambda}_D = \{\lambda_{D,1}, \lambda_{D,2}, \dots, \lambda_{D,N_{LM}}\}^\top$ are unknown coefficient vectors to be solved for which correspond to the solution variables $\tilde{\phi}_h$ and λ_D respectively via Equation (4.48). Furthermore, $K_{\text{ER}} = (k_{\text{ER},ij})$ is the stiffness matrix, $\Lambda = (a_{ij})$ is the Lagrange Multiplier matrix, and $F_{\text{ER}} = (f_{\text{ER},i})$ is the force vector with entries defined as

$$k_{\text{ER},ij} = B_{\text{ER}}(v_j, v_i), \quad (4.50)$$

$$a_{ij} = \langle v_j, \zeta_i \rangle_{\Gamma(\tilde{\phi}^0)}, \quad (4.51)$$

and

$$f_{\text{ER},i} = J_{\text{ER},3}(|\nabla \tilde{\phi}^{m-1}|; \nabla \tilde{\phi}^{m-1}, v_i). \quad (4.52)$$

4.4.1 Elliptic Reinitialisation method: numerical examples

In order to demonstrate the efficacy of the proposed Elliptic Reinitialisation method a number of numerical examples will be presented. Each of these example problems begins with a level set function with a given interface position but with a gradient which varies over the domain (in each case chosen specifically to include areas where the norm of the gradient is both less than and greater than unity), being projected onto a mesh of square elements of uniform size and uniform polynomial order. The level set function can then be reinitialised and error data collected using the error measures to be presented in the following section, Section 4.4.1.1. Each example will then be repeated for a series of meshes of different h , for each p allowing for convergence data to be presented for the proposed method.

4.4.1.1 Error measures

Where a known solution, ϕ , exists, the error between that and the computed solution, ϕ_h , is given in the L^2 norm which can be stated as

$$E_{L^2}^2(\phi_h, \phi, \mathcal{T}) = \sum_{\tau \in \mathcal{T}} \int_{\tau} (\phi_h - \phi)^2 \, d\mathbf{x}, \quad (4.53)$$

the L^∞ norm which can be stated as

$$E_{L^\infty}(\phi_h, \phi, \mathcal{T}) = \max_{\mathbf{x} \in \mathcal{T}} |\phi_h - \phi|, \quad (4.54)$$

and the DG norm which can be stated as

$$E_{\text{DG}}^2(\phi_h, \phi, \mathcal{T}) = \sum_{\tau \in \mathcal{T}} \int_{\tau} (\nabla(\phi_h - \phi))^2 \, d\mathbf{x} + \sum_{e_{\text{int}} \in \mathcal{S}(\mathcal{T})} \mu \int_{e_{\text{int}}} \llbracket \phi_h - \phi \rrbracket^2 \, d\mathbf{x}. \quad (4.55)$$

When the analytical solution is not known, there are two additional error measures which can demonstrate the efficacy of the reinitialisation method. The first is an error measure which measures globally, the degree to which the computed solution satisfies the Eikonal equation, that is

$$E_{\text{SD}}^2(\phi_h, \mathcal{T}) = \sum_{\tau \in \mathcal{T}} \int_{\tau} (|\nabla \phi_h| - 1)^2 \, d\mathbf{x}, \quad (4.56)$$

which will be referred to as the *signed distance error measure*. The second of these, is a measure of the movement of the interface in the L^2 norm, which is evaluated by integrating the difference between the computed and desired value of the solution along the original position of the interface, that is

$$E_{\text{Int}}^2(\phi_h, \Gamma(\phi^0)) = E_{L^2}^2(\phi_h, 0, \Gamma(\phi^0)) = \int_{\Gamma(\phi^0)} \phi_h^2 \, d\mathbf{x}, \quad (4.57)$$

which will be referred to as the *interface error measure*.

It should be noted that for all of the numerical experiments presented in this section Müller's method [144] is used to compute the integral along the interface, with the maximum order of the divergence free basis functions, β' , equal to 10. This is higher than would be required in practice for the problems to be presented, however, it allows as much as possible one to remove the error associated with the mesh/problem dependency of the integration method and thus better evaluate the reinitialisation method. Furthermore, for the numerical examples to be presented, the Picard iteration is considered to have converged when $\left|E_{SD}(\tilde{\phi}^m) - E_{SD}(\tilde{\phi}^{m-1})\right| < 10^{-8}$.

4.4.1.2 h-convergence study: circular interface

The first experiment presented is the reinitialisation of a level set function, $\tilde{\phi}^0$, which can be described analytically by the quadric

$$\tilde{\phi}^0 = x^2 + y^2 - 1, \quad (4.58)$$

in the domain $\Omega = (-2, 2)^2$. This corresponds to a circular interface centred at the origin. The signed distance function with the same interface, and therefore the analytical solution to the reinitialisation of (4.58) can thus be stated

$$\tilde{\phi} = \sqrt{x^2 + y^2} - 1. \quad (4.59)$$

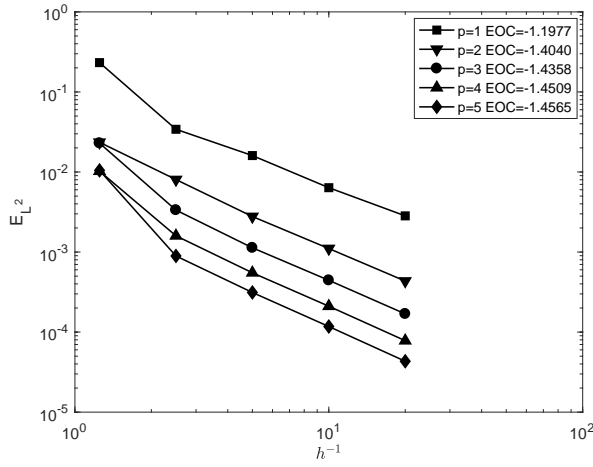
For this problem, the position of the level set interface can also be described analytically as follows, for $0 \leq \vartheta \leq 2\pi$,

$$\begin{aligned} x &= \cos(\vartheta), \\ y &= \sin(\vartheta). \end{aligned} \quad (4.60)$$

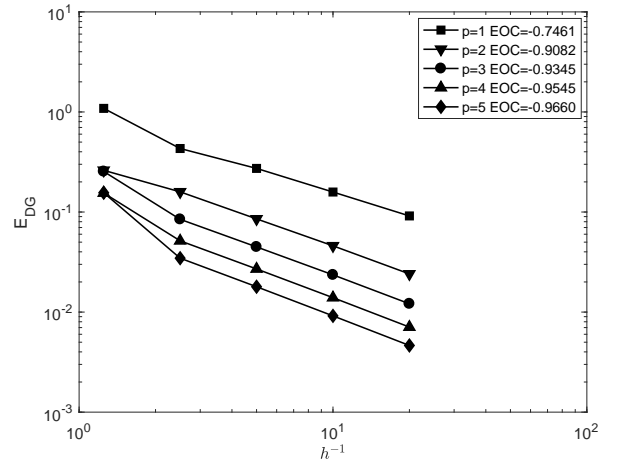
This allows the interface error measure to be computed using the trapezium rule, to remove any error associated with the methods for integrating over an implicit surface. An h -convergence study is performed by computing the reinitialisation of the level set function, initialised as the L^2 projection of (4.58), on a sequence of Cartesian meshes with square elements of size, $h = 0.8, 0.4, 0.2, 0.1, 0.05$, of uniform polynomial order, $p = 1, 2, 3, 4, 5$. Error data will be presented using each of the norms defined in Section 4.4.1.1.

For elliptic problems discretised using SIPG, theoretically optimal convergence rates in the L^2 norm are known to be h^{p+1} , and in the DG norm, h^p , [45], assuming the problem is sufficiently smooth. It has also been shown that the theoretically optimal rate of convergence in L^∞ norm is proportional to $\ln(h^{-1})^{\bar{s}} h^{p+1}$, where $\bar{s} = 1$ for $p = 1$, and $\bar{s} = 0$ otherwise, [152]. In [110], it is shown that for a problem which lacks sufficient smoothness, the rate of convergence will be equal to the linear case for all p . The signed distance error measure acts similarly to the H^1 seminorm, computing the difference between measures of the gradient of the solution and such, it would be reasonable to expect optimal convergence rates to be equivalent to optimal convergence in the H^1 seminorm, which is known to be h^p , once again assuming sufficient smoothness.

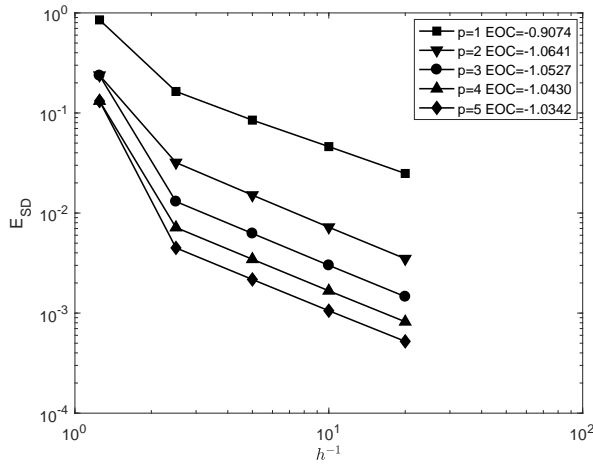
The analytical solution for this problem, as defined in (4.59), is singular at the origin, and



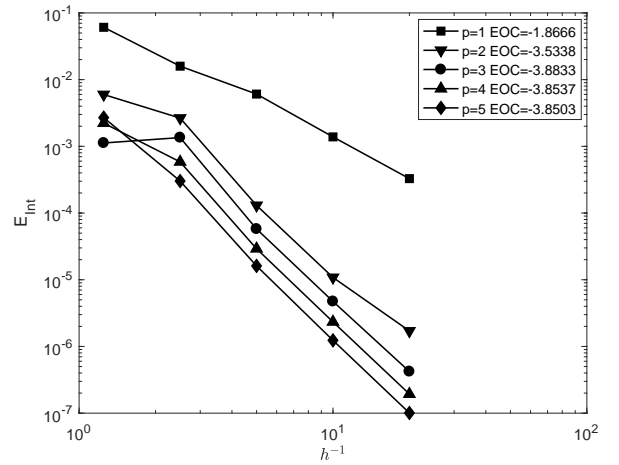
(a) Convergence in the L^2 norm.



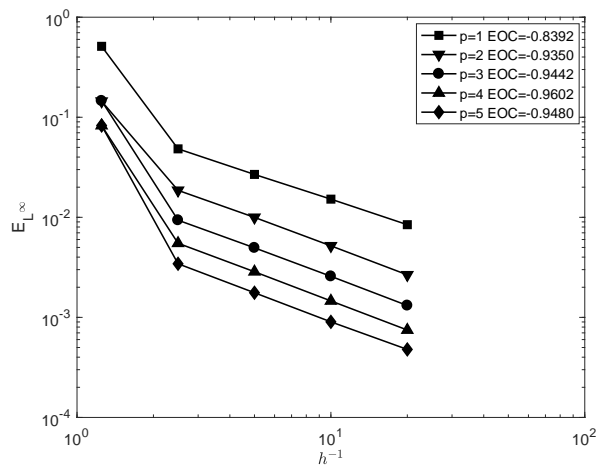
(b) Convergence in the DG norm.



(c) Convergence using the signed distance error measure.



(d) Convergence using the interface error measure.



(e) Convergence in the L^∞ norm.

Figure 4.7: Error data and convergence rates for the circular interface problem in the domain $\Omega = (-2, 2)^2$, solved using the Elliptic Reinitialisation method.

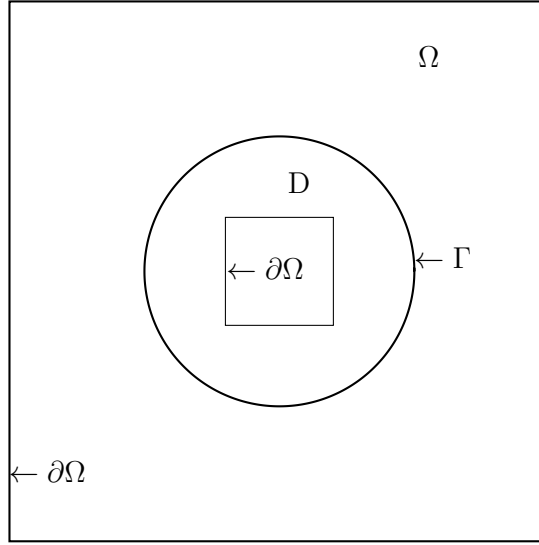


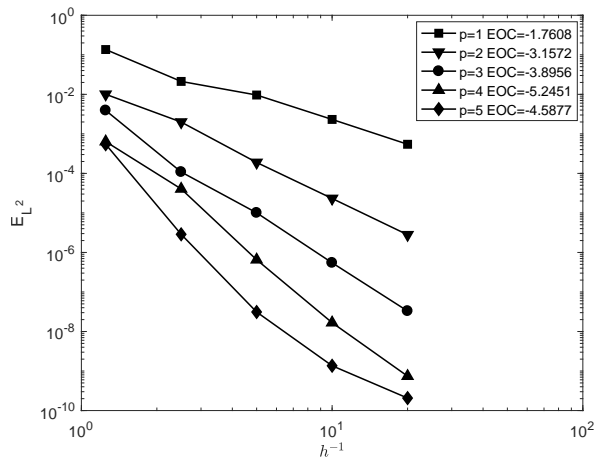
Figure 4.8: Domain configuration for the circular interface reinitialisation problem where the singular part has been removed from the domain, i.e. $\Omega = (-2, 2)^2 \setminus (-0.4, 0.4)^2$.

therefore it should be expected, given the contents of the previous paragraph, that for this problem the experimental order of convergence in the L^2 norm is h^2 , in the DG norm and signed distance error norms is h^1 , and in the L^∞ norm is around $\ln(h^{-1})h^2$, for all p . The results of the h -convergence study are presented in Figure 4.7 and demonstrate that beyond the initial pre-asymptotic datum the experimental orders of convergence, using the four aforementioned error measures, are congruent with those expected for a non-smooth problem. It should be noted that the quoted orders of convergence for all measures and polynomial orders are computed using the difference between the results for $h = 0.4$ and $h = 0.05$.

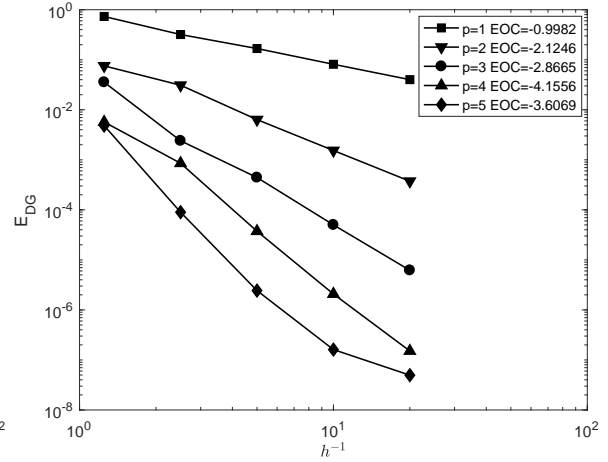
It is unknown, to the knowledge of the author, how the interface error measure should behave with mesh refinement. For this experiment, it can be observed that there is an increase in the experimental order of convergence between the meshes where $p = 1$ and $p = 2$, however, this remains constant for any $p > 2$. For the purposes of the following discussion, it is useful to observe that the presence of a singularity in the mesh, constrains the rate at which the L^2 error in the solution along the interface decreases when using high-order elements.

4.4.1.3 h-convergence study: circular interface with narrow band

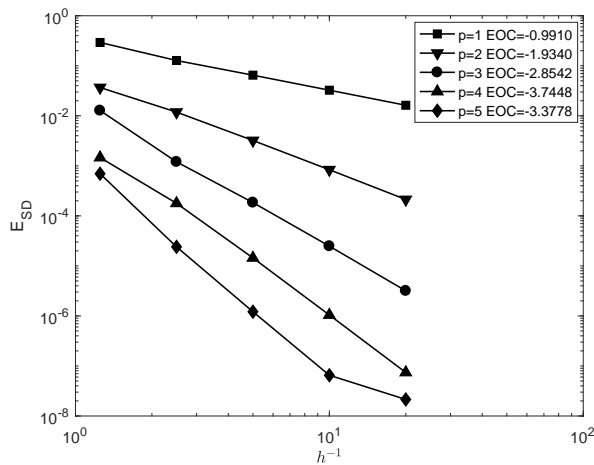
For the previous example problem, the analytical solution is singular, and thus the experimental orders of convergence as expected were limited when using meshes of higher-order elements. In order to return rates of convergence which align with the theoretically optimal then, one needs to change the domain over which the reinitialisation is computed such that it contains the interface but not the singular regions of the solution. As discussed in Section 3.2.2, this can be achieved by adopting the use of a narrow band approach. For this somewhat trivial example, the singularity is known to occur at the origin and thus a naive implementation of a narrow band approach, is to simply repeat the previous experiment in the domain, $\Omega = (-2, 2)^2 \setminus (-0.4, 0.4)^2$, such that the singularity is removed. The relative position of the interface and the domain boundaries can be seen in Figure 4.8.



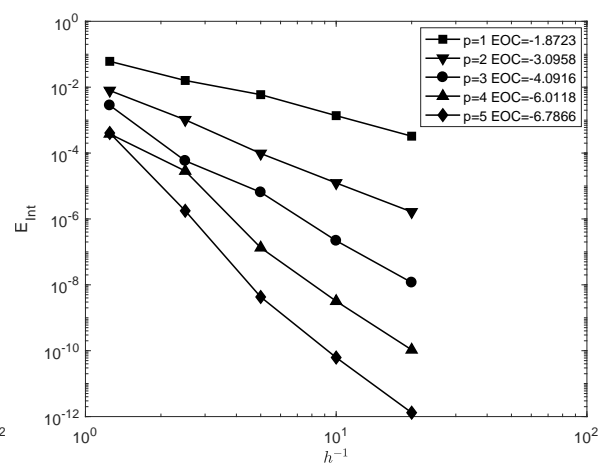
(a) Convergence in the L^2 norm.



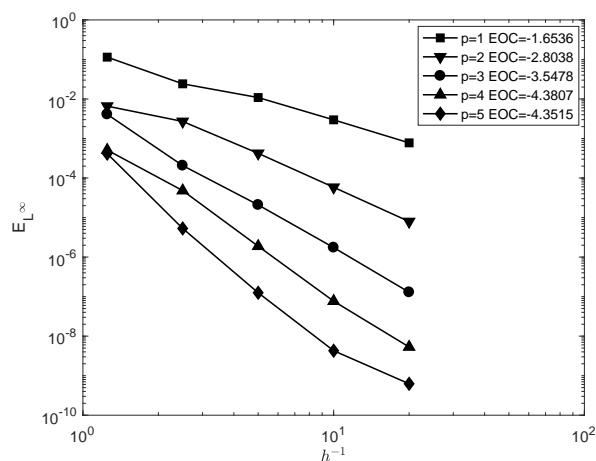
(b) Convergence in the DG norm.



(c) Convergence using the signed distance error measure.



(d) Convergence using the interface error measure.



(e) Convergence in the L^∞ norm.

Figure 4.9: Error data and convergence rates for the circular interface reinitialisation problem on the domain $\Omega = (-2, 2)^2 \setminus (-0.4, 0.4)^2$, solved using the Elliptic Reinitialisation method.

Here then, the h -convergence study from Section 4.4.1.2 is repeated on the new domain leading to the results shown in Figure 4.9. As expected, by removing the origin from the problem domain, the solution everywhere inside the domain is now smooth enough to display experimental convergence rates which align with the theoretically optimal rates in all of the relevant norms. The convergence rates using the interface error measure for this experiment seem to decrease on the order of h^{p+1} , which suggests that one might expect this to be the optimal rate of convergence for this error measure. It should be noted that the quoted orders of convergence for all measures and polynomial orders are computed using the difference between the results for $h = 0.4$ and $h = 0.05$. It can also be noted here that for this example the number of iterations taken to reach the convergence criterion is often few; for this simple example, for the mesh with $h = 0.05$ and $p = 5$, just 6 iterations are required, although there is considerable variance in the required number of iterations to converge across the range of h and p tested.

4.4.1.4 h -convergence study: smooth star interface

Two more example problems will be presented of a more arbitrary nature than the simple circle example, thus the rule used to define the width of the narrow band for these examples is as follows: remove from the mesh any element which has a minimum absolute nodal value greater than four times the size of the smallest element, h_{min} . For these example problems, the error data presented will be using the signed distance and interface error measures only, and in both cases the presented errors will be normalised by the area of the domain inside the narrow band. Also, for these examples the interface error will be computed using Müller's method (as opposed to the trapezium rule for the previous examples), and as such the error computed will be a measure of the movement of the interface from its initial projection as opposed to the distance from the analytical solution (although in practice, calculating the error in these two ways gives similar results except for the coarsest meshes tested).

The first of the arbitrary interfaces has an initial level set function which can be defined everywhere by

$$\tilde{\phi}^0 = x^2 + y^2 - \left(1 + 0.2 \sin \left(6 \arctan \left(\frac{y}{x}\right)\right)\right), \quad (4.61)$$

on a domain of maximum size $\Omega = (-2, 2)^2$. The function (4.61) describes an interface centred at the origin which takes the shape of a smooth six pointed star, as shown in Figure 4.10(a), and which is therefore referred to henceforth as the *smooth star interface reinitialisation problem*. It should be noted that for each possible mesh in the convergence study defined by element size, h and element order p , the initial projection of (4.61) onto Ω will be different and thus the portion of the domain which remains inside the narrow band (that is the size of the narrow band) will therefore also be different. For this experiment, the h -convergence study will be computed on a sequence of Cartesian meshes with square elements of size $h = 0.4, 0.2, 0.1, 0.05, 0.025$, for meshes of uniform polynomial order, $p = 1, 2, 3$.

Figure 4.11(a) shows the error and convergence data for the smooth star interface reinitialisation problem using the signed distance error measure. The first two data points for all polynomial orders show linear convergence; this is because the criterion used to define the narrow band for those meshes, is yet to be sufficient to remove the more complicated singular region

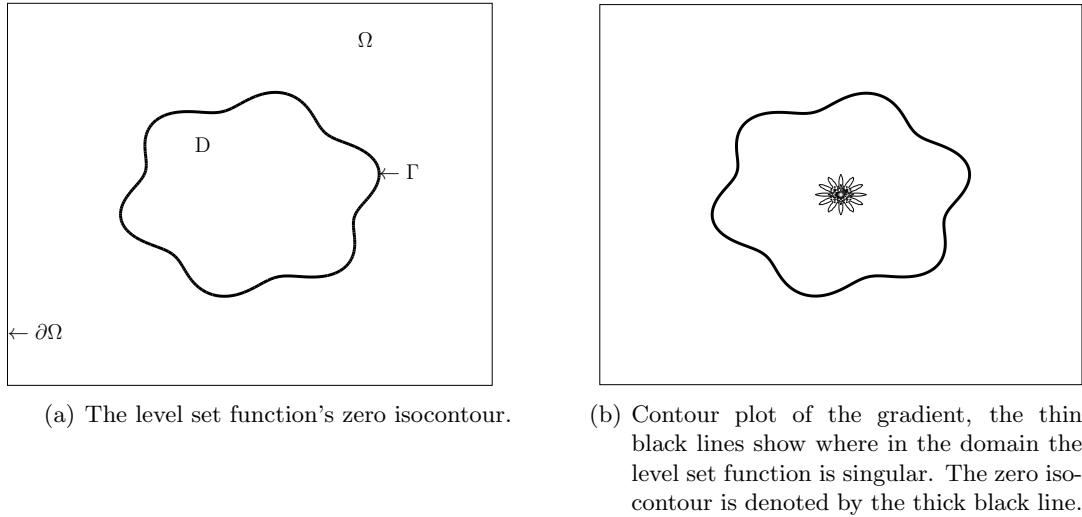


Figure 4.10: Domain configuration for the smooth star interface reinitialisation problem, initialised as stated in Equation (4.61), where $\Omega = (-2, 2)^2$.

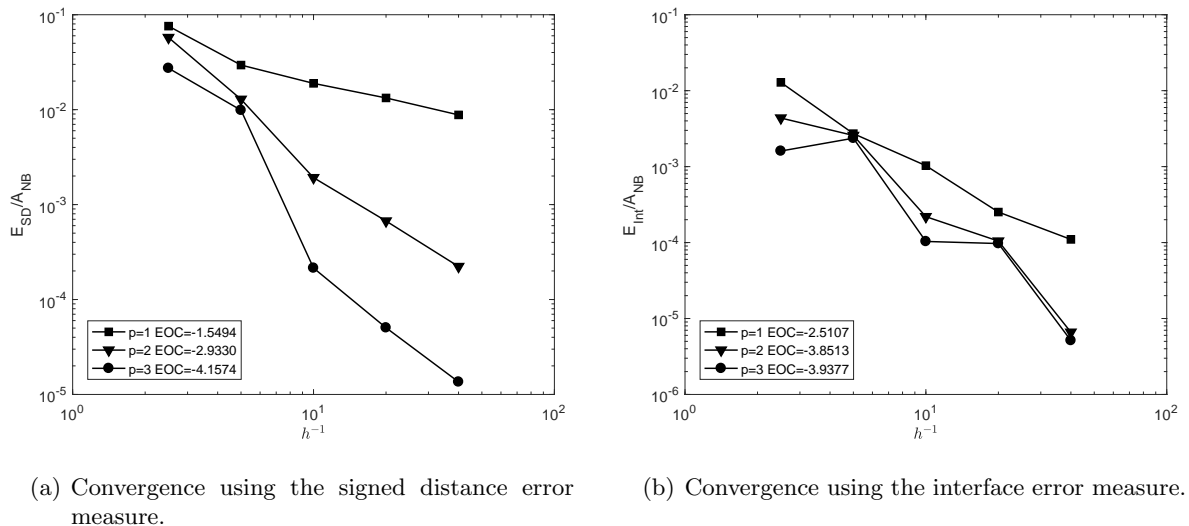


Figure 4.11: Error data normalised by narrow band area, A_{NB} , and associated convergence rates for the smooth star interface reinitialisation problem, initialised as stated in Equation (4.61), in the domain $\Omega = (-2, 2)^2$, with narrow band, solved using the Elliptic Reinitialisation method.

of the solution from the domain, see Figure 4.10(b). As h becomes smaller, the narrow band becomes narrower and the singular part is removed, beyond which the experimental order of convergence aligns with that which is expected to be optimal. The quoted experimental orders of convergence for both error measures in this example, and for all polynomial orders, are computed using the difference between the results for $h = 0.1$ and $h = 0.025$.

The rate of convergence for the interface error increases slightly between the meshes of linear and quadratic elements, however increasing the polynomial order of the elements beyond that, no longer results in an increase in the accuracy of the solution at the interface, despite the improving solution in the signed distance error measure. This reiterates that which was found for the non-smooth circular interface problem in Section 4.4.1.2.

4.4.1.5 h-convergence study: multiple interfaces

The final example to be presented consists of multiple nested interfaces of various curvatures, which more closely resembles a level set function which one might encounter in practice. This will be referred to henceforth as the *multiple interface reinitialisation problem*. The initial level set function at a point is defined as the maximum value of one of three analytical functions, i.e.

$$\tilde{\phi}^0 = \max(\tilde{\phi}_k^0), \quad k = 1, 2, 3, \quad (4.62)$$

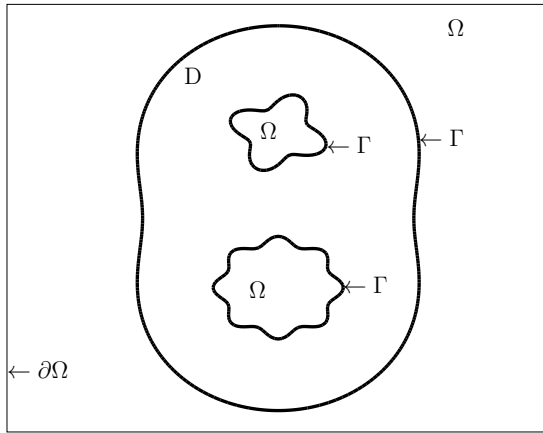
where

$$\begin{aligned} \tilde{\phi}_1^0 &= 1.5 \left(\sqrt{x^2 + y^2} - \left(1 + 0.8 \sin \left(\arctan \left(\frac{y}{x} \right) \right)^2 \right) \right), \\ \tilde{\phi}_2^0 &= -2 \left(\sqrt{x^2 + y^2} - \left(0.3 - 0.075 \sin \left(4 \arctan \left(\frac{y - 0.8}{x} \right) \right) \right) \right), \\ \tilde{\phi}_3^0 &= -2 \left(\sqrt{x^2 + y^2} - \left(0.48 - 0.08 \sin \left(4 \arctan \left(\frac{y - 0.65}{x} \right) \right)^2 \right) \right). \end{aligned} \quad (4.63)$$

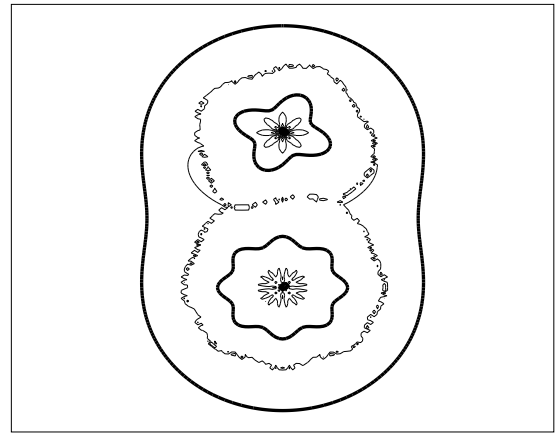
The level set interface defined by (4.62), can be seen in Figure 4.12(a). For this example problem, an h -convergence study is computed on a sequence of Cartesian meshes with square elements of size $h = 0.4, 0.2, 0.1, 0.05, 0.025, 0.0125$, for meshes of uniform polynomial order $p = 1, 2, 3$. As for the previous example, the convergence results are given using only the signed distance and interface error measures, once again normalised by the area of the domain comprising the narrow band at the point when the error is computed.

Looking at the error and convergence data computed using the signed distance error measure in Figure 4.13(a), it can once again be seen that until the mesh is sufficiently refined and therefore the narrow band sufficiently narrow, there are singularities present in the solution and the experimental order of convergence for all polynomial orders, p , is equivalent to the linear case. That is the case for all meshes with element size, $h \geq 10^{-1}$. Beyond this point, the experimental orders of convergence align with the theoretically optimal for the signed distance error measure. The quoted orders of convergence for both error measures, and for all polynomial orders are computed for this example using the difference between the results for $h = 0.05$ and $h = 0.0125$.

The error and convergence data computed using the interface error measure is displayed

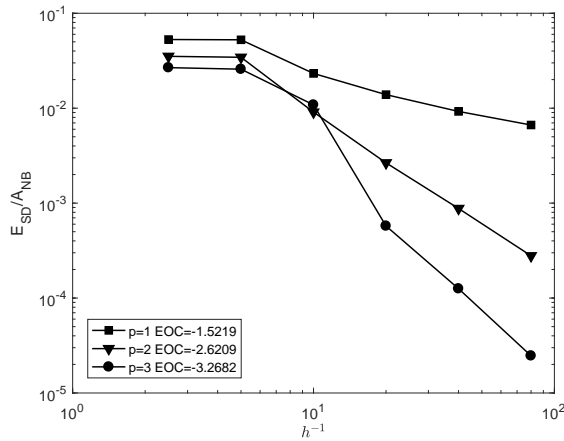


(a) The level set function's zero isocontour.

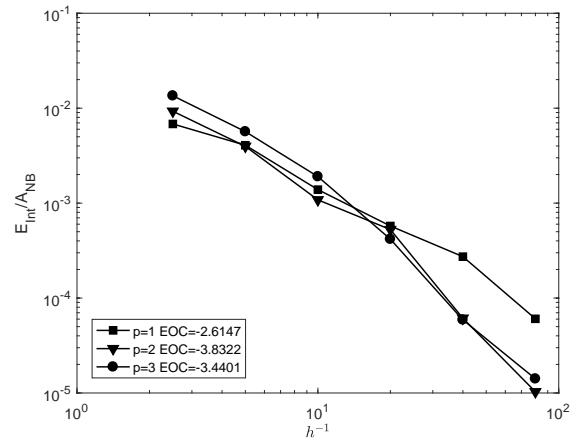


(b) Contour plot of the gradient, the thin black lines show where in the domain the level set function is singular. The zero isocontour is denoted by the thick black line.

Figure 4.12: Domain configuration for the multiple interface problem where, $\Omega = (-2, 2)^2$.



(a) Convergence using the signed distance error measure.



(b) Convergence using the interface error measure.

Figure 4.13: Error data normalised by narrow band area, A_{NB} , and associated convergence rates for the multiple interfaces interface reinitialisation problem, initialised as stated in Equation (4.61), in the domain $\Omega = (-2, 2)^2$, with narrow band, solved using the Elliptic Reinitialisation method.

in Figure 4.13(b). It shows that for a given element size, h , the error is almost equivalent, regardless of polynomial order, p , with a small increase in accuracy between the meshes where $p = 1$ and $p = 2$, as was also the case for the previous example. As has been the case for all of the presented examples, it is difficult to explain the behaviour of this error measure for this problem. It could be the case that this behaviour is a result of using Müller's method to compute the integral over the interface which enforces the Dirichlet boundary condition. In their original paper [144], Müller *et al.* are able to demonstrate high-order accuracy, however, the error measure stated is an average of errors for the same problem computed multiple times after small perturbations to the position of the domain (which would change the relative position of the mesh and the implicit interface). This is done presumably because in the general case they were not able to demonstrate high-order accuracy for a given interface on a given domain, which might be what is happening here. Despite this, of the methods tested for computing such an integral, it was the only one which was able to demonstrate robustly that its accuracy improved with mesh refinement. Thus ultimately beyond this comments are restricted to the following; for all examples the demonstrated movement of the level set function at the interface is small (especially in comparison to other reinitialisation methods, see [137] for example), and furthermore can be decreased predictably by controlling the element size with order $\sim h^2$.

Another point of note for this particular example is that for the denser higher-order meshes, the number of iterations required to satisfy the convergence criterion grows large, for this problem when $p = 3$ it takes an average of 920 iterations. However, it can also be noted that, for the mesh where $h = 0.0125$ and $p = 3$ tested, it takes just 5 iterations to improve the gradient solution by 3 orders of magnitude, and 34 iterations for an improvement of 4 orders of magnitude. This suggests that in practical situations, it would be up to the user to decide where to strike the balance between expense and accuracy.

4.5 Parabolic Reinitialisation method

The knowledge gained from the investigations into boundary conditions and modified objective functionals using the elliptic method can be applied directly to the diffusion part of the Parabolic Reinitialisation formulation. In this section then, all that is left to investigate is the influence of various discretisations of the time derivative.

4.5.1 Parabolic reinitialisation method: spatial semi-discretisation

Using the modified Eikonal formulation, i.e. the potential function, P_3 , from Equation (4.42), the strong form of the Parabolic Reinitialisation method can be stated as

$$\begin{aligned} \frac{\partial \tilde{\phi}(\mathbf{x}, t_r)}{\partial t_r} + \nabla \cdot \left(d_3(|\nabla \tilde{\phi}(\mathbf{x}, t_r)|) \nabla \tilde{\phi}(\mathbf{x}, t_r) \right) &= 0, \quad \mathbf{x} \in \Omega, \\ \tilde{\phi}(\mathbf{x}, t_r) &= 0, \quad \mathbf{x} \in \Gamma(\tilde{\phi}^0), \\ d_3(|\nabla \tilde{\phi}(\mathbf{x}, t_r)|) \nabla \tilde{\phi}(\mathbf{x}, t_r) \cdot \hat{\mathbf{n}} &= 0, \quad \mathbf{x} \in \partial\Omega. \end{aligned} \tag{4.64}$$

Using the method of Lagrange multipliers to enforce the Dirichlet boundary condition, and applying an SIPG discretisation in space leads to a variational formulation of the problem

(4.64) to be stated as follows; find $\tilde{\phi}_h(\mathbf{x}, t_r) \in V_{\mathbf{p}}(\mathcal{T})$ and $\lambda_D(\mathbf{x}, t_r) \in \mathfrak{L}(\mathcal{T}_\Gamma)$ from $t_r = (0, \infty)$, such that

$$\left(\frac{\partial \tilde{\phi}_h(\mathbf{x}, t_r)}{\partial t_r}, v_h \right)_{\mathcal{T}} + B_{\text{PR}}(|\tilde{\phi}_h(\mathbf{x}, t_r)|; \tilde{\phi}_h(\mathbf{x}, t_r), v_h) + \langle \lambda_D, v_h \rangle_{\Gamma(\tilde{\phi}^0)} = 0, \quad \forall v_h \in V_{\mathbf{p}}(\mathcal{T}), \quad (4.65)$$

and

$$\left\langle \tilde{\phi}_h(\mathbf{x}, t_r), \zeta \right\rangle_{\Gamma(\tilde{\phi}^0)} = 0, \quad \forall \zeta \in \mathfrak{L}(\mathcal{T}_\Gamma), \quad (4.66)$$

where the notation $B_{\text{PR}}(|\nabla \tilde{\phi}|; \tilde{\phi}, v)$ is introduced for conciseness to denote the form

$$\begin{aligned} B_{\text{PR}}(|\nabla \tilde{\phi}|; \tilde{\phi}, v) &= \left(d_3(|\nabla \tilde{\phi}|) \nabla \tilde{\phi}, \nabla v \right)_{\mathcal{T}} - \left\langle \{d_3(|\nabla \tilde{\phi}|) \nabla \tilde{\phi}\}, \llbracket v \rrbracket \right\rangle_{S(\mathcal{T})} \\ &\quad - \left\langle \{d_3(|\nabla \tilde{\phi}|) \nabla v\}, \llbracket \tilde{\phi} \rrbracket \right\rangle_{S(\mathcal{T})} + \mu \left\langle \llbracket \tilde{\phi} \rrbracket, \llbracket v \rrbracket \right\rangle_{S(\mathcal{T})}, \end{aligned} \quad (4.67)$$

which is bilinear in $\tilde{\phi}$ and v . Discretising (4.64) spatially in this way (and also dropping the dependency on \mathbf{x}) leads to a linear system of nonlinear ODEs which can be stated as

$$M \frac{\partial \tilde{\phi}(t_r)}{\partial t_r} + K_{\text{PR}}(|\tilde{\phi}(t_r)|) \tilde{\phi}(t_r) + \Lambda^\top \boldsymbol{\lambda}_D(t_r) = 0, \quad (4.68)$$

$$\Lambda \tilde{\phi}(t_r) = 0, \quad (4.69)$$

where $M = (m_{ij})$ is the mass matrix, with entries defined as

$$m_{ij} = (v_j, v_i)_{\mathcal{T}}, \quad (4.70)$$

and $K_{\text{PR}}(|\tilde{\phi}(t_r)|) = (k_{\text{PR},ij}(|\tilde{\phi}(t_r)|))$ is the nonlinear stiffness matrix with entries defined as

$$k_{\text{PR},ij}(|\tilde{\phi}(t_r)|) = B_{\text{PR}}(|\tilde{\phi}(t_r)|, v_j, v_i), \quad (4.71)$$

and $\Lambda = (a_{ij})$ is the Lagrange Multiplier matrix with entries defined in Equation (4.51).

4.5.2 Parabolic reinitialisation method: full discretisations

Explicit [79], semi-implicit [153] and fully implicit [77, 80] time discretisations have all found use when solving parabolic problems with nonlinear diffusive terms discretised spatially using IPDG methods, see Section 2.3.1. Explicit methods, despite their simplicity, often demonstrate severe time step restrictions when combined with IPDG spatial discretisations and have thus found limited use in the literature. However, as is noted in [79], explicit discretisations could become competitive due to their ability to be parallelised. Implicit methods, on the other hand have found preference in the literature due to their superior stability, however, the requirement to solve a nonlinear system at each time step could similarly be considered prohibitively expensive. Semi-implicit methods, which use a suitable combination of explicit and implicit linearisations for a given problem, can demonstrate improved stability in comparison with explicit methods, whilst only requiring a linear system to be solved at each time step, however, the efficacy of such

an approach is of course problem dependent, and therefore difficult to analyse *a priori*. As it is not immediately obvious which is most suitable for the given problem all three types of temporal discretisation will be investigated below. More specifically, three different full discretisations will be explored: an SIPG in space, Explicit Euler (EE) in time discretisation (SIPG-EE); an SIPG in space, Semi-Implicit (SI) method in time (SIPG-SI); and an SIPG in space, Implicit Euler (IE) in time discretisation (SIPG-IE).

4.5.2.1 Fully explicit

For both forward and backward Euler methods, the same notation will be used for the discretisation of the time derivative, that is

$$\frac{\partial \tilde{\phi}_h(t_r)}{\partial t_r} \approx \frac{\tilde{\phi}_h^m - \tilde{\phi}_h^{m-1}}{\Delta t_r}, \quad (4.72)$$

where Δt_r denotes the time step, and $\tilde{\phi}^m = \tilde{\phi}(t_r^m)$ denotes the value of $\tilde{\phi}$ at the m^{th} time step, i.e. $t_r^m = m\Delta t_r$. Linearising the diffusive terms explicitly, the SIPG-EE full discretisation can thus be stated as follows: find $\tilde{\phi}_h^m \in V_{\mathbf{p}}(\mathcal{T})$, and $\lambda_D^m \in \mathcal{L}(\mathcal{T}_\Gamma)$, as $t_r^m \rightarrow \infty$ such that

$$\begin{aligned} \left(\tilde{\phi}_h^m, v_h \right)_{\mathcal{T}} + \Delta t_r \langle \lambda_D^m, v_h \rangle_{\Gamma(\tilde{\phi}^0)} &= \left(\tilde{\phi}_h^{m-1}, v_h \right)_{\mathcal{T}} - \Delta t_r B_{\text{PR}}(|\tilde{\phi}_h^{m-1}|; \tilde{\phi}_h^{m-1}, v_h), \\ \forall v_h \in V_{\mathbf{p}}(\mathcal{T}), \end{aligned} \quad (4.73)$$

and

$$\Delta t_r \left\langle \tilde{\phi}_h^m, \zeta \right\rangle_{\Gamma(\tilde{\phi}^0)} = 0, \quad \forall \zeta \in \mathcal{L}(\mathcal{T}_\Gamma). \quad (4.74)$$

Using the matrix notation introduced in Sections 4.4 and 4.5.1, the linear system to be solved at each time step can thus be stated as

$$\begin{bmatrix} M & \Delta t_r \Lambda^\top \\ \Delta t_r \Lambda & 0 \end{bmatrix} \begin{Bmatrix} \tilde{\phi}^m \\ \lambda_D^m \end{Bmatrix} = \begin{Bmatrix} (M - \Delta t_r K_{\text{PR}}(|\tilde{\phi}^{m-1}|)) \tilde{\phi}^{m-1} \\ 0 \end{Bmatrix}. \quad (4.75)$$

4.5.2.2 Semi-implicit

The idea behind semi-implicit methods is to deal with the nonlinear parts of the problem explicitly, whilst dealing with the linear parts implicitly. As such various semi-implicit formulations of (4.65)-(4.66) are possible. Here we choose to apply an explicit linearisation to the nonlinear diffusion coefficient, $d(|\nabla \tilde{\phi}|)$, such that it is a function of the solution at the previous time step, t_r^{m-1} . A backwards Euler method can then be used to discretise the time derivative present in the resulting system of linear ODEs. The SIPG-SI full discretisation, can thus be stated: find $\tilde{\phi}_h^m \in V_{\mathbf{p}}(\mathcal{T})$, and $\lambda_D^m \in \mathcal{L}(\mathcal{T}_\Gamma)$, as $t_r^m \rightarrow \infty$ such that

$$\left(\tilde{\phi}_h^m, v_h \right)_{\mathcal{T}} + \Delta t_r B_{\text{PR}}(|\tilde{\phi}_h^{m-1}|; \tilde{\phi}_h^m, v_h) + \Delta t_r \langle \lambda_D^m, v_h \rangle_{\Gamma(\tilde{\phi}^0)} = \left(\tilde{\phi}_h^{m-1}, v_h \right)_{\mathcal{T}}, \quad \forall v_h \in V_{\mathbf{p}}(\mathcal{T}), \quad (4.76)$$

and

$$\Delta t_r \left\langle \tilde{\phi}_h^m, \zeta \right\rangle_{\Gamma(\tilde{\phi}^0)} = 0, \quad \forall \zeta \in \mathcal{L}(\mathcal{T}_\Gamma). \quad (4.77)$$

Once again using the matrix notation introduced in Section Sections 4.4 and 4.5.1, the linear system to be solved at each time step can be stated as

$$\begin{bmatrix} M + K_{\text{PR}}(|\tilde{\phi}^{m-1}|)\Delta t_r & \Delta t_r \Lambda^\top \\ \Delta t_r \Lambda & 0 \end{bmatrix} \begin{Bmatrix} \tilde{\phi}^m \\ \lambda_D^m \end{Bmatrix} = \begin{Bmatrix} M\tilde{\phi}^{m-1} \\ 0 \end{Bmatrix}. \quad (4.78)$$

4.5.2.3 Fully implicit

Using the backwards Euler method to discretise the time derivative present in the system of nonlinear ODEs (4.65)-(4.66) allows one to state the corresponding variational formulation, SIPG-IE, as: find $\tilde{\phi}_h^m \in V_{\mathcal{P}}(\mathcal{T})$, and $\lambda_D^m \in \mathfrak{L}(\mathcal{T}_\Gamma)$, as $t_r^m \rightarrow \infty$ such that

$$\frac{1}{\Delta t_r} \left(\tilde{\phi}_h^m - \tilde{\phi}_h^{m-1}, v_h \right)_{\mathcal{T}} + B_{\text{PR}}(|\tilde{\phi}_h^m|; \tilde{\phi}_h^m, v_h) + \langle \lambda_D^m, v_h \rangle_{\Gamma(\tilde{\phi}^0)} = 0, \quad \forall v_h \in V_{\mathcal{P}}(\mathcal{T}), \quad (4.79)$$

and

$$\left\langle \tilde{\phi}_h^m, \zeta \right\rangle_{\Gamma(\tilde{\phi}^0)} = 0, \quad \forall \zeta \in \mathfrak{L}(\mathcal{T}_\Gamma). \quad (4.80)$$

The resulting system (4.79)-(4.80) is still nonlinear and therefore needs to be solved at each time step using an appropriate iterative solver. For simplicity, it is chosen here to use a quasi-Newton scheme which can be stated as follows

$$\begin{Bmatrix} \tilde{\phi}^{m,k+1} \\ \lambda_D^{m,k+1} \end{Bmatrix} = \begin{Bmatrix} \tilde{\phi}^{m,k} \\ \lambda_D^{m,k} \end{Bmatrix} - \begin{bmatrix} \left[\frac{\partial F_{\text{NR}}(\tilde{\phi}_h^{m,k}, \lambda_D^{m,k})}{\partial \tilde{\phi}} \right] \\ \left[\frac{\partial G_{\text{NR}}(\tilde{\phi}_h^{m,k}, \lambda_D^{m,k})}{\partial \tilde{\phi}} \right] \end{bmatrix} \begin{bmatrix} \left[\frac{\partial F_{\text{NR}}(\tilde{\phi}_h^{m,k}, \lambda_D^{m,k})}{\partial \lambda_D} \right] \\ 0 \end{bmatrix}^{-1} \begin{Bmatrix} F_{\text{NR}}(\tilde{\phi}_h^{m,k}, \lambda_D^{m,k}) \\ G_{\text{NR}}(\tilde{\phi}_h^{m,k}, \lambda_D^{m,k}) \end{Bmatrix}, \quad (4.81)$$

where $F_{\text{NR}} = (f_{\text{NR},i})$ and $G_{\text{NR}} = (g_{\text{NR},i})$ are the residual vectors associated with (4.79) and (4.80) respectively, with elements given by

$$f_{\text{NR},i}(\tilde{\phi}_h^{m,k}, \lambda_D^{m,k}) = \frac{1}{\Delta t_r} \left(\tilde{\phi}_h^{m,k} - \tilde{\phi}_h^{m,0}, v_i \right)_{\mathcal{T}} + B_{\text{PR}}(|\tilde{\phi}_h^{m,k}|; \tilde{\phi}_h^{m,k}, v_i) + \langle \lambda_D^{m,k}, v_i \rangle_{\Gamma(\tilde{\phi}^0,0)}, \quad (4.82)$$

and

$$g_{\text{NR},i}(\tilde{\phi}_h^{m,k}, \lambda_D^{m,k}) = \left\langle \tilde{\phi}_h^{m,k}, \zeta_i \right\rangle_{\Gamma(\tilde{\phi}^0,0)}. \quad (4.83)$$

The notation $\tilde{\phi}_h^{m,k}$ denotes the k^{th} Newton iteration, at the m^{th} time step, and thus $\tilde{\phi}^{m,0} = \tilde{\phi}^{m-1,\infty}$. The constituents of the Jacobian matrix, DF , that is $\frac{\partial F_{\text{NR}}}{\partial \tilde{\phi}}$, $\frac{\partial G_{\text{NR}}}{\partial \tilde{\phi}}$ and $\frac{\partial F_{\text{NR}}}{\partial \lambda_D}$, are computed using a first order finite difference method, i.e.

$$\left[\frac{\partial f_{\text{NR},i}(\tilde{\phi}_h^{m,k}, \lambda_D^{m,k})}{\partial \tilde{\phi}_j} \right]_{ij} = \frac{f_{\text{NR},i}(\tilde{\phi}_{h,j}^{m,k} + \delta \tilde{\phi}, \lambda_D^{m,k}) - f_{\text{NR},i}(\tilde{\phi}_{h,j}^{m,k}, \lambda_D^{m,k})}{\delta \tilde{\phi}}, \quad (4.84)$$

and

$$\begin{aligned} \left[\frac{\partial f_{\text{NR},i}(\tilde{\phi}_h^{m,k}, \lambda_D^{m,k})}{\partial \lambda_{D,j}} \right]_{ij} &= \left[\frac{\partial g_{\text{NR},i}(\tilde{\phi}_h^{m,k}, \lambda_D^{m,k})}{\partial \tilde{\phi}_j} \right]_{ij}^\top, \\ &= \frac{f_{\text{NR},i}(\tilde{\phi}_h^{m,k}, \lambda_D^{m,k} + \delta \lambda_D) - f_{\text{NR},i}(\tilde{\phi}_h^{m,k}, \lambda_D^{m,k})}{\delta \lambda_D}, \end{aligned} \quad (4.85)$$

where $\delta \tilde{\phi}$ and $\delta \lambda_D$ are two small constants, chosen for all examples in this thesis as $\delta \tilde{\phi} = \delta \lambda_D = 10^{-12}$. Furthermore, again for all examples presented in this thesis, the quasi-Newton loop is considered to have converged when $\|\{DF^{-1}[F_{\text{NR}} G_{\text{NR}}]^\top\}\|_2 < 10^{-10}$, that is the Euclidean norm of the relative change between Newton iterations is less than the specified tolerance.

4.5.3 Parabolic Reinitialisation: numerical examples

For all of the numerical experiments presented in this section again Müller's method, [144], is used to compute the integral along the interface, with the maximum order of the divergence free basis functions, β' , equal to 10, so to, as much as possible allow one to remove the error associated with the mesh/problem dependency of the integration method and thus better evaluate the reinitialisation method. The stopping criterion defining convergence is for all examples in this section defined as $|E_{SD}(\tilde{\phi}^m) - E_{SD}(\tilde{\phi}^{m-1})| < 10^{-8}$, that is the relative change in the signed distance error is smaller than a tolerance.

4.5.3.1 Investigation into critical time step

One of the first considerations to be made in regards to the three possible discretisations is the appropriate choice for the time step, Δt_r , in each case. This first experiment then, will form an investigation into the critical time step, Δt_{crit} , for the three time discretisations for a simple problem. To that end, the circular interface reinitialisation problem from Section 4.4.1.3 will be repeated here, by which the function (4.58), is L^2 projected onto the domain $\Omega = (-2, 2)^2 \setminus (-0.4, 0.4)^2$, which is then reinitialised using each of the three discretisations.

The circular interface reinitialisation problem will be computed on a sequence of Cartesian meshes of bilinear, that is $p = 1$, square elements of size, $h = 0.8, 0.4, 0.2, 0.1$. For each mesh, the three methods will attempt to reinitialise the level set function as a signed distance function to the interface, for a sequence of time steps of magnitude, $\Delta t_r^j = 2^{-(j-1)}$, for $j \in \mathbb{N}$. For each considered time step, Δt_r^j , the solver will run until either: the solution diverges, defined by the condition $E_{SD}(\tilde{\phi}^m) > 100$, the solution stagnates, defined by the condition, $E_{SD}(\tilde{\phi}^m) < E_{SD}(\tilde{\phi}^{m-1})$ AND $E_{SD}(\tilde{\phi}^{m-1}) > E_{SD}(\tilde{\phi}^{m-2})$, or the solution converges. The first time step with which the method converges will be considered the critical time step, Δt_{crit} .

The discussion begins by looking at the results for a fixed mesh. In this case the chosen mesh is that where $h = 0.2$; the results of which are shown in Figure 4.14 with details given in Table 4.1. The results presented in Figure 4.14 are exactly what one might expect from the three time discretisations. For the fully implicit discretisation, it can be seen in Figure 4.14 that the method is stable across the all of the time steps tested. For this experiment, when using the implicit discretisation the maximum time step considered $\Delta t_r^1 = 1$ was satisfactory and as such for the implicit discretisation larger values were also considered i.e. $\Delta t_r = 10, 100, 1000$. It was found that in all cases the implicit method converged to the same error values up to the number

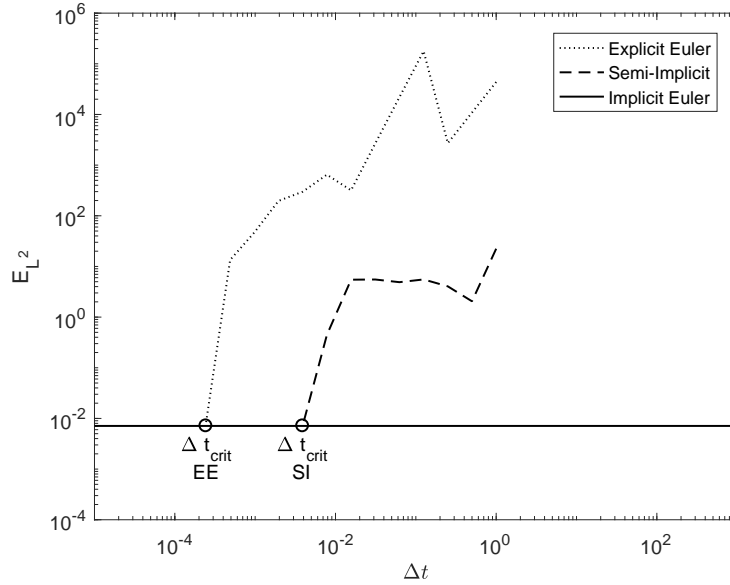


Figure 4.14: Variation of error in the L^2 norm of the converged solution to the circular interface reinitialisation problem with time step, for the three discretisations of the Parabolic Reinitialisation method on a fixed mesh with $h = 0.2$ and $p = 1$.

Method	Δt_{crit}	Number of Iterations	Time per Iteration (s), (Relative Time)	Total Time (s), (Relative Time)	E_{L^2}	E_{SD}
SIPG-EE	2.44e-4	40475	1.17e-1, (1.28e-2)	4713.40, (130.05)	8.05e-3	6.51e-2
SIPG-SI	3.91e-3	3101	2.60e-1, (2.87e-2)	806.45, (22.25)	7.25e-3	6.51e-2
SIPG-IE	-	4	9.06, (1)	36.24, (1)	7.10e-3	6.51e-2
Elliptic	-	295	1.21e-1, (1.34e-2)	35.55, (0.98)	7.40e-3	6.51e-2

Table 4.1: Critical time steps with associated errors and computational expense for the three discretisations of the Parabolic Reinitialisation method, as well as the Elliptic Reinitialisation method, when solving the circular interface reinitialisation problem. For the SIPG-IE formulation of the Parabolic method, Δt_r was set to 1000.

of significant figures given in Table 4.1, with the number of iterations required reducing from 25 at $\Delta t_r = 1$ to just 4 at $\Delta t_r = 1000$. As such the critical time step for the implicit discretisation was never reached in this experiment, and thus the results presented here assume $\Delta t_{\text{crit}} = 1000$ for the implicit discretisation. Figure 4.14 shows that when using the fully explicit discretisation, the method always diverges for values of the time step which do not strictly satisfy the CFL condition, and is stable for any value smaller. For the semi-implicit discretisation, it can be seen in Figure 4.14 that for time steps greater than some threshold value the method diverges, however there is also a region, in this case, $10^{-2} < \Delta t_r < 0.5$, whereby the solution stagnates and oscillates between multiple poor approximations. Once the critical time step is reached, the semi-implicit method is stable. Finally, it can be seen in Figure 4.14 that the critical time step for the semi-implicit discretisation is larger than for the explicit discretisation.

Table 4.1 gives more specific information pertaining to the critical time step for the problem on the fixed mesh, where $p = 1$ and $h = 0.2$. It can be seen in Table 4.1 that all three of the proposed discretisations converge to the same value of E_{SD} with slight variations in the E_{L^2} ;

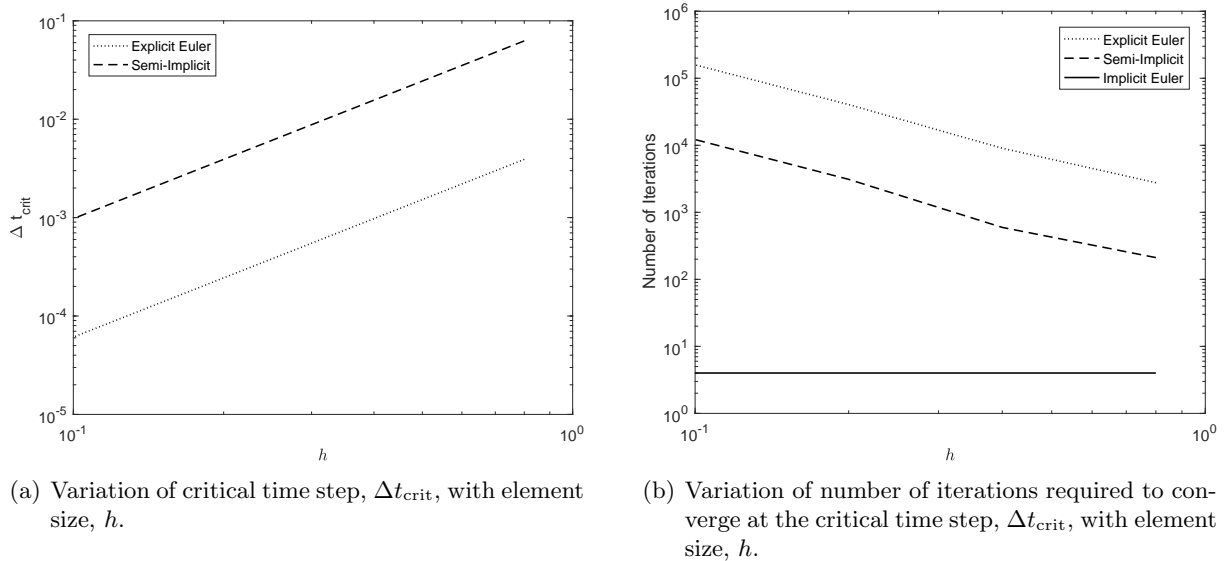


Figure 4.15: Variation of critical time step and number of iterations for the three discretisations of the Parabolic Reinitialisation method, for the circular interface reinitialisation problem.

this is because the convergence (and divergence) criteria are functions of the signed distance error (as this is what is being enforced), and as such all solutions which do converge, converge to the same value of E_{SD} . Table 4.1 also includes a result using the Elliptic Reinitialisation method. As will be seen throughout the numerical examples section, the computed error to which the proposed Parabolic and Elliptic Reinitialisation methods converge is approximately equal in all cases.

Also included in Table 4.1 is a comparison of the relative computational expense of the various discretisations and methods. It can be seen that the critical time step for both the fully explicit and semi-implicit time schemes results in extreme numbers of iterations required to achieve convergence. Such large numbers of iterations mean that in terms of computational time a semi-implicit discretisation for this problem was around 22 times more expensive than the implicit discretisation, and the explicit discretisation around 130 times more expensive. A result using the Elliptic Reinitialisation method is also presented and where it can be seen that the time taken in that case is roughly equivalent to the SIPG-IE parabolic formulation. It should be noted that all of the timing data presented in Table 4.1 was obtained using the `tic` function in MATLAB R2016b. All of the data was collected from a machine with an Intel i7-7700HQ CPU at 2.80GHz. It should also be noted that the relative time per iteration was calculated as the time taken to compute the time stepping loop to convergence divided by the number of iterations, normalised by the average time taken to compute one iteration of the SIPG-IE loop. It should also be noted that the resolution of the experiment is fairly coarse in terms of the time steps considered, however, this coarseness does little to affect the conclusion which can be drawn.

Figure 4.15 shows how the critical time step is related to h and also the effect of this on the number of iterations required to converge using the critical time step. As mentioned above, the critical time step was never reached in the case of the SIPG-IE discretisation and as such the

number of iterations presented in Figure 4.15(b) using SIPG-IE is for all h set to $\Delta t_r = 1000$. The ability to choose this value as a large constant as h varies, allows the number of iterations required to converge to also remain constant as h varies, for this problem. This fact further compounds the noted issues with expense of both the SIPG-EE and SIPG-SI discretisations stated earlier, whereby at least for the Explicit Euler discretisation the critical time step is known to be proportional to h^2 , [79].

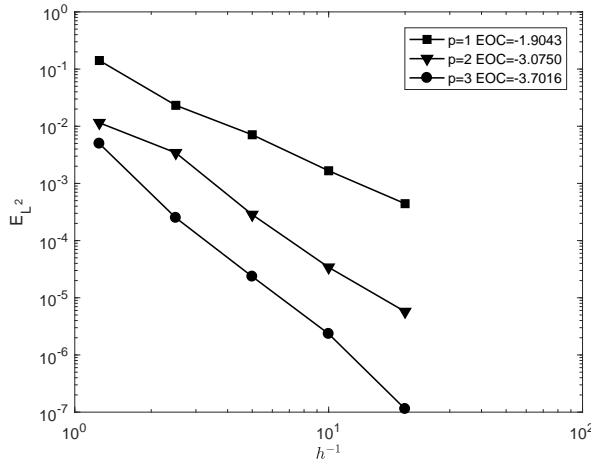
The results presented in this section make it clear that the comparative expense of the fully implicit discretisation is orders of magnitude less than either the explicit or semi-implicit discretisations. This is true despite the nonlinear solve required, due to severe restrictions on the time step required for stability and thus the large number of iterations required to reach a steady-state for both the semi-implicit and fully explicit discretisations. As there is no benefit accuracy-wise, as evidenced by Figure 4.14, it is thus recommended by the author that for problems of this kind, one should adopt an implicit time discretisation. For the remaining example problems to be presented in this section therefore, the focus shall be only on the SIPG-IE formulation of the Parabolic Reinitialisation method.

4.5.3.2 h -convergence study: circular interface

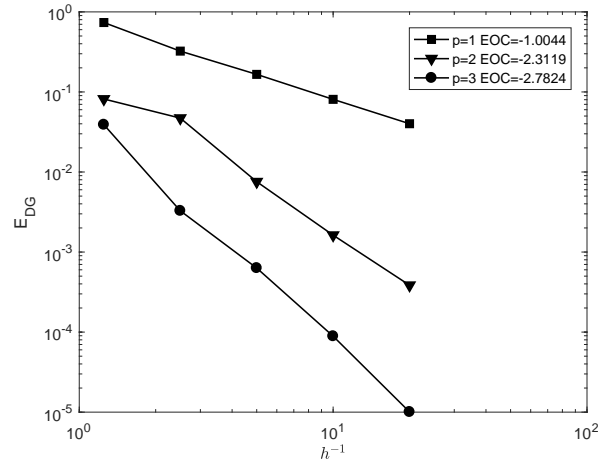
For the first numerical experiment testing the accuracy of the SIPG-IE formulation of the Parabolic Reinitialisation method, the same initial conditions as in Section 4.4.1.3 will be repeated i.e. a level set function with a circular interface defined by (4.58) is L^2 projected onto the domain $\Omega = (-2, 2)^2 \setminus (-0.4, 0.4)^2$. The initial level set function is then reinitialised such that the explicit function describing the solution everywhere in the domain is that stated in Equation (4.59). An h -convergence study will be computed on a sequence of Cartesian meshes with square elements of size, $h = 0.8, 0.4, 0.2, 0.1, 0.05$, for meshes of uniform polynomial order, $p = 1, 2, 3$. It should be noted for all of the partitions considered the time step is chosen as $\Delta t_r = 100$. Given that this is a relatively simple problem, which is smooth everywhere, it should again be expected that the proposed method will be capable of demonstrating experimental orders of convergence which align with the theoretically optimal for diffusion problems in all of the relevant norms as discussed in Section 4.4.1.2. The error and convergence data collected, is presented in Figure 4.16.

Figure 4.16 demonstrates that the method has experimental orders of convergence congruent with the expected optimal convergence rates in all of the relevant norms including a convergence rate of h^{p+1} in the L^2 norm and interface error norm, a convergence rate of h^p in the DG norm and signed distance error norm, and a convergence rate of $\ln(h^{-1})^{\bar{s}} h^{p+1}$ in the L^∞ norm. It can be noted that the quoted experimental order of convergence is computed in all cases using the difference between the error values for the meshes where $h = 0.4$ and $h = 0.05$. It can also be seen that the reported errors and quoted rates of convergence are essentially equivalent for both the Elliptic reinitialisation method, see Section 4.4.1.3, and the proposed Parabolic Reinitialisation method.

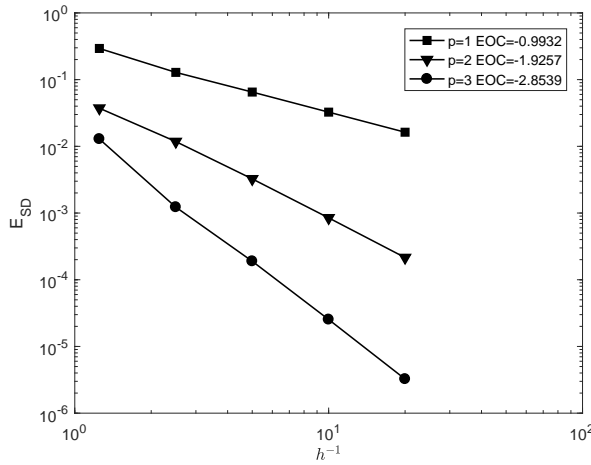
It can be noted that for fixed time step $\Delta t_r = 100$, the number of iterations required for almost all of the tested partitions is equal to 5. The exceptions to this was for the meshes of cubic elements where $h = 0.4$ and $h = 0.05$, where 4 and 6 iterations respectively were



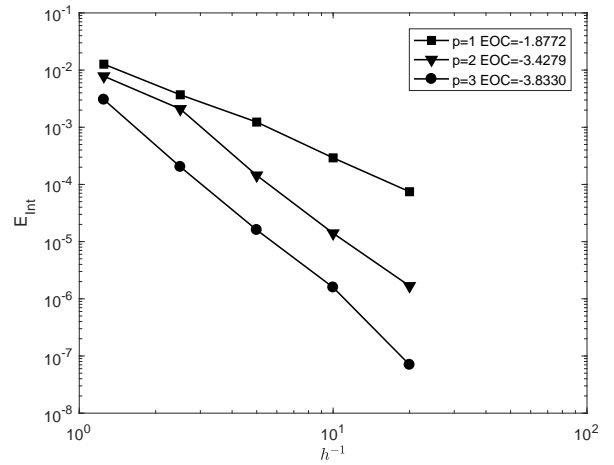
(a) Convergence in the L^2 norm.



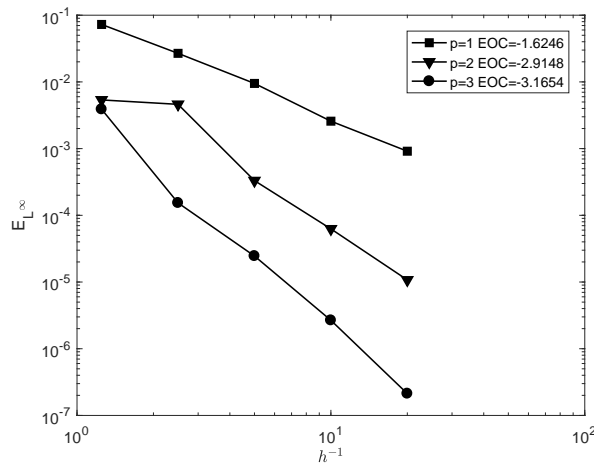
(b) Convergence in the DG norm.



(c) Convergence using the signed distance error measure.



(d) Convergence using the interface error measure.



(e) Convergence using the L^∞ norm.

Figure 4.16: Error data and convergence rates for the circular interface reinitialisation problem in the domain $\Omega = (-2, 2)^2 \setminus (-0.4, 0.4)^2$, solved using the Parabolic Reinitialisation method.

required to satisfy the convergence criterion. As shown in Table 4.1, 295 iterations are required to satisfy the convergence criterion using the Elliptic Reinitialisation method where $p = 1$ and $h = 0.2$, which lead to comparable run times for that given mesh, however, for the mesh where $p = 3$ and $h = 0.05$, the Elliptic method requires just 5 iterations to converge. It is difficult to approximate *a priori* the required number of iterations to satisfy the convergence criterion for either the Parabolic or Elliptic reinitialisation methods which makes it difficult to choose in which situation one would be preferable over the other in terms of computational expense.

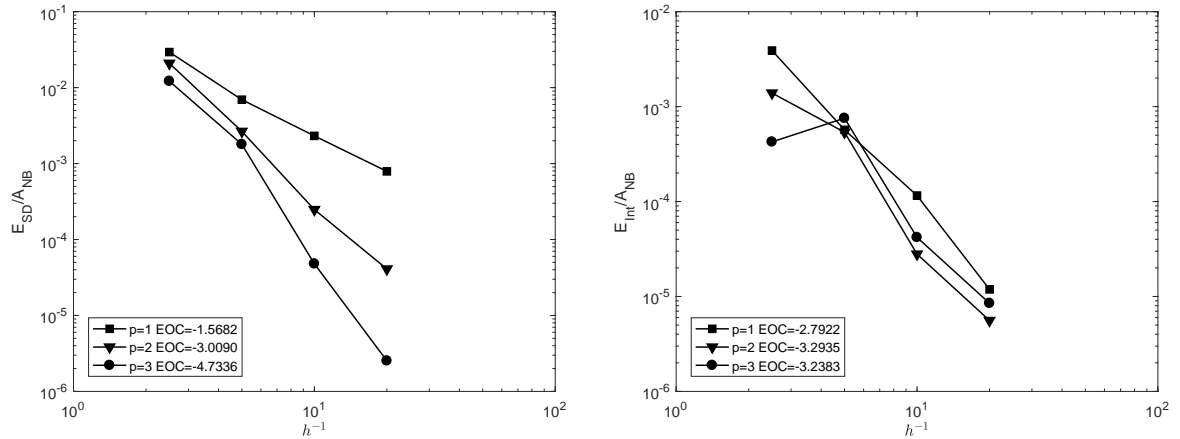
4.5.3.3 h-convergence study: smooth star interface

A second h -convergence study is presented for the smooth star interface reinitialisation problem, solved using the SIPG-IE discretisation of the Parabolic Reinitialisation method. The initial condition defined in Section 4.4.1.4, Equation (4.61), is L^2 projected onto the domain $\Omega = (-2, 2)^2$, where again a narrow band determines the actual domain size (which can vary for any given partition tested in the study). In this case, the narrow band is defined automatically for each mesh through the following condition; remove from the mesh any element which has a minimum absolute nodal value greater than two times the size of the smallest element, h_{min} . This initial level set function is then reinitialised on a series of Cartesian meshes of uniform polynomial order, $p = 1, 2, 3$, with square elements of sizes $h = 0.4, 0.2, 0.1, 0.05$. The error and convergence data to be presented is only given using the signed distance and interface error measures, and again, as the size of the domain over which these errors are calculated varies these error measures are normalised by the area of the domain comprising the narrow band. It should be noted for all meshes considered the time step is chosen as $\Delta t_r = 1$.

It is possible, especially for the coarser meshes, that a signed distance function to the smooth star interface may contain a singular region within the narrow band defined as it is. Given that this is the case, it should be expected that for any set of partitions which contain a solution which is singular, the rate of convergence will be limited to the equivalent linear rate of convergence in the relevant norms. For the set of partitions which are sufficiently fine such that the criterion for the narrow band becomes sufficiently narrow to remove all singular parts of the solution from the domain, then one should expect the method to once again demonstrate optimal convergence. The computed error and convergence data is presented in Figure 4.17.

Figure 4.17(a) shows that beyond the first two data for each polynomial order, an experimental order of convergence of h^p can be observed using the signed distance error measure which is equivalent to the previous example. It can be noted that the quoted experimental order of convergence is computed in all cases using the difference between the error values where $h = 0.2$ and $h = 0.05$. Figure 4.17(b) shows error and convergence data computed using the interface error measure. The behaviour of this error measure is again difficult to explain but does at least act consistently across the range of example problems beyond the circular interface. As such comments are limited to the following; it seems that one can expect the L^2 error along the interface to decrease proportionally to at least h^2 for all p , for any problem more complicated than the circular interface reinitialisation problem.

The same example problem was computed using the Elliptic Reinitialisation method in Section 4.4.1.4. The reported errors using the signed distance error measure are essentially equiv-



(a) Convergence using the signed distance error measure. (b) Convergence using the interface error measure.

Figure 4.17: Error data normalised by narrow band area, A_{NB} , and associated convergence rates for the smooth star interface reinitialisation problem in the domain $\Omega = (-2, 2)^2$, using a narrow band approach, solved using the Parabolic Reinitialisation method.

alent for both methods. This is less true for the interface error measure, however, it can be noted that the general trend is the same and for the reasons stated above otherwise difficult to analyse.

For this example problem, there is more variation in the number of iterations required to satisfy the convergence criterion. This number does not seem to scale with h although this is likely due to the variations in the domain size due to the narrow banding rule. Furthermore, it can be noted that the average number of iterations for all of the partitions tested is 14 with a maximum number of required iterations equal to 52, for the mesh where $p = 3$ and $h = 0.2$. The same problem computed using the Elliptic Reinitialisation method also has a somewhat random distribution of number of iterations required to converge for each partition tested, with an average of 563 iterations. As there is no pattern in the required number of iterations for a given level set function on a given mesh, it is again difficult *a priori* to determine whether or not it would be advantageous in terms of computational expense to choose one formulation over the other. This echoes the comparison made for the previous numerical experiment in Section 4.5.3.2.

4.5.3.4 h-convergence study: multiple interfaces

The final h -convergence study involves solving the multiple interface reinitialisation problem using the SIPG-IE formulation of the Parabolic Reinitialisation method. In this case, the initial level set function is defined at each point in the domain as in Section 4.4.1.5, Equations (4.62) and (4.63), which is sampled at the Gauss points and then L^2 projected onto the domain $\Omega = (-2, 2)^2$. This initial level set function is then reinitialised on a series of Cartesian meshes of uniform polynomial order, $p = 1, 2, 3$, with square elements of size $h = 0.2, 0.1, 0.05, 0.025, 0.0125$. Again, only the signed distance and interface errors, normalised

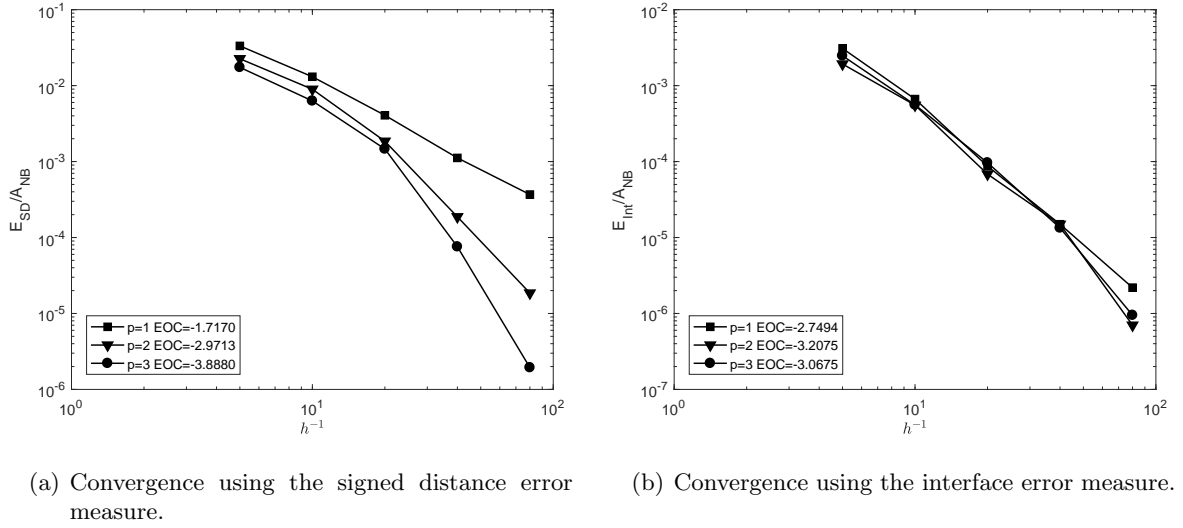


Figure 4.18: Error data normalised by the narrow band area, A_{NB} , and associated convergence rates for the multiple interfaces reinitialisation problem in the domain $\Omega = (-2, 2)^2$, using a narrow band approach, solved using the Parabolic Reinitialisation method.

by the narrow band area, will be reported for this problem. Furthermore, the narrow band will be applied to the problem restricting the domain size by the following rule: remove from the mesh any element which has a minimum absolute nodal value greater than four times the size of the smallest element, h_{\min} .

In computing the solution to the multiple interface reinitialisation problem, using the SIPG-IE discretisation, it was found that in some cases for stability reasons there was a restriction required on the time step, Δt_r . In particular, it was the coarsest meshes tested which were the least stable, with the required time step decreasing with increasing polynomial order. The likely reason for this is therefore that the singularities present in the solution (which are quite severe for this example as can be seen in Figure 4.12(b)) are poorly resolved by the mesh leading to overshoots and oscillations for larger time steps. This does provide evidence suggesting that the SIPG-IE discretisation, is only conditionally stable. As such for this numerical example, the time step is chosen such that $\Delta t_r = 0.01$, for all meshes. It should be noted however, that for the denser meshes, particularly those in the region where optimal rates of convergence are demonstrated, larger time steps do once again become viable.

The error and convergence data shown in Figure 4.18 echoes the previous example problem shown in Section 4.5.3.3. That is, for meshes where the narrow band is not sufficiently narrow to remove the singular parts of the solution the signed distance error measure converges with a rate equal to linear convergence for all polynomial orders. Beyond this point, the experimental order of convergence using the signed distance error measure aligns with what is expected to be optimal. Similarly, the interface error measure gives approximately equal values for all polynomial orders for a given element size and converges with a rate of at least h^2 for all polynomial orders. It should be noted that the quoted experimental order of convergence in all

cases is computed using the values for the meshes where $h = 0.05$ and $h = 0.0125$.

4.6 Summary

This chapter has presented research concerning level set reinitialisation in the DG paradigm. This research is concluded with the proposal of two novel reinitialisation methods, the Elliptic and Parabolic Reinitialisation methods, both of which have been shown to have significant advantages over those presented elsewhere in the literature, particularly with regards to accuracy both globally and along the level set interface. The proposed Elliptic and Parabolic Reinitialisation methods (Parabolic here refers to the SIPG-IE discretisation) are shown to be practically equivalent in terms of the solution computed, and to differ mainly in terms of (real-)time taken and stability. The stability of the Parabolic method, however, is parametrised by a time step, the choice of which is not obvious, and furthermore, which method will converge faster is difficult to predict. As the Elliptic Reinitialisation method is both unconditionally stable, and also parameter free in this respect, it is the preferred reinitialisation method of the author, and that which will be used elsewhere in this thesis where there is a requirement to reinitialise a level set function. With the development of the proposed level set reinitialisation method, one can maintain confidence that throughout an evolution the level set function will maintain its satisfaction of the signed distance property, ensuring stability and promoting accuracy, and further that inaccuracies introduced by the reinitialisation in the position of the level set interface have been minimised. The next chapter can thus proceed by presenting research concerning level set evolution in the DG paradigm.

Chapter 5

Level Set Evolution

With the development of a high-order accurate level set reinitialisation method, this chapter presents a DG approach applied to the level set equation to thus form a fully DG high-order accurate level set methodology. As such this chapter comprises the following sections. Section 5.1 and its subsections present a simplification which can be made to the level set equation and subsequently information pertaining to the DG spatial discretisation of the simplified level set equation including a novel flux term, and the proposed temporal discretisation of the level set equation using a high-order explicit Runge-Kutta method. Section 5.2 presents a more robust approach to narrow banding the level set function, including a novel approach to extending the value of the level set function beyond the narrow band as the level set interface evolves. Section 5.3 presents the formulation of an Anderson acceleration algorithm used in this thesis to increase the rate of convergence of the Picard iterative method used to linearise some of the equations present in the proposed level set methodology. Finally, Section 5.4, combines all of the information pertaining to the level set methodology presented up to this point in the thesis, including evolution, narrow banding, and reinitialisation, in the form of an algorithm. Section 5.4 is then concluded with a number of numerical examples demonstrating the efficacy of the proposed methodology.

5.1 The level set evolution equation

As the level set function will always be initialised as a signed distance function at the beginning of a simulation, see Section 3.2.1, and reinitialised as a signed distance function throughout the evolution, see Chapter 4, the following simplification can be made to the evolution equation. Given that the advection velocity can be written, $\mathbf{b} = b\mathbf{n}_\phi$ where, b , is the scalar magnitude of the advection velocity normal to the interface, and, $\mathbf{n}_\phi = \frac{\nabla\phi(\mathbf{x}, t_e)}{|\nabla\phi(\mathbf{x}, t_e)|}$, is the normal of the level set function, then, the evolution equation (3.2) can be rewritten as

$$\frac{\partial\phi(\mathbf{x}, t_e)}{\partial t_e} = -b \frac{\nabla\phi(\mathbf{x}, t_e)}{|\nabla\phi(\mathbf{x}, t_e)|} \cdot \nabla\phi(\mathbf{x}, t_e) = -b|\nabla\phi(\mathbf{x}, t_e)| = -b. \quad (5.1)$$

This leaves a pure source equation, however, in general the magnitude of the advection velocity vector is likely to be a function of the level set function and as such it can be useful to

conceptualise the simplified advection equation as a Hamilton-Jacobi equation as follows

$$\frac{\partial \phi(\mathbf{x}, t_e)}{\partial t_e} + b(\phi(\mathbf{x}, t_e)) = 0. \quad (5.2)$$

In this case, it is known, [154], that discretising (5.1) using a member of the class of monotone schemes results in a solution which will converge to the unique viscosity solution. The viscosity solution of (5.1) will be uniquely defined by the initial condition

$$\phi(\mathbf{x}, 0) = \phi, \quad \forall \mathbf{x} \in \Omega. \quad (5.3)$$

5.1.1 Level set evolution equation: spatial semi-discretisation

In order to discretise (5.1) spatially on a DG mesh, one can begin in the standard way by multiplying through by a test function, v_h , and integrating over the domain.

$$\left(\frac{\partial \phi_h(\mathbf{x}, t_e)}{\partial t_e}, v_h \right)_{\mathcal{T}} = (-b, v_h)_{\mathcal{T}}, \quad \forall v \in V_{\mathbf{p}}(\mathcal{T}). \quad (5.4)$$

As there is no longer a spatial derivative, there is no integration by parts through which a flux term would naturally occur. In the DG paradigm, with no flux term describing the relationship between sets of adjacent elements, each element will evolve separately from one another causing the development of sharp discontinuities throughout the computational domain. As is pointed out in [155], the numerical difference across an edge can be used to couple cells, which amounts to adding artificial dissipation to the flux. In this work, taking inspiration from interior penalty DG approaches, a simple method is used to couple adjacent cells by which the jump in the solution across each internal element edge, $\partial\tau \in S(\mathcal{T})$, is penalised, which weakly enforces continuity. This can be stated

$$\left(\frac{\partial \phi_h(\mathbf{x}, t_e)}{\partial t_e}, v_h \right)_{\mathcal{T}} + \mu \langle \llbracket \phi_h(\mathbf{x}, t_e) \rrbracket, \llbracket v_h \rrbracket \rangle_{S(\mathcal{T})} = (-b, v_h)_{\mathcal{T}}, \quad \forall v_h \in V_{\mathbf{p}}(\mathcal{T}), \quad (5.5)$$

where the penalty parameter, μ , is chosen as, $\mu = 10p^2/h$. Whilst this is an interior penalty method, and the resulting matrix on the LHS is symmetric, a distinction is made here that this discretisation is not equivalent to the standard SIPG discretisation as in [45], which one could apply to an elliptic problem. It is noted that enforcing continuity in this way adds an artificial viscosity to the system which could contribute to mass loss or the dissipation of physical discontinuities, however this will be mitigated by the high-order accuracy of the method, and the high resolution of such features when discretised with the full hp -adaptive method. Furthermore, the efficacy of this flux term will be demonstrated in the numerical results presented in Section 5.4.2.

5.1.2 Level set evolution equation: full discretisation

In general, the magnitude of the advection velocity vector, b , will be a function of the level set function, $b = b(\phi_h(\mathbf{x}, t_e))$. In such a case, the RHS of (5.5) will be dealt with explicitly. In order to maintain the high-order accuracy of which the spatial discretisation is capable, a FE space

0					
c_2	q_{21}				
c_3	q_{31}	q_{32}			
\vdots	\vdots		\ddots		
c_{N_s}	$q_{N_s 1}$	$q_{N_s 2}$	\dots	q_{N_s, N_s-1}	
	α_1	α_2	\dots	α_{N_s-1}	α_{N_s}

Table 5.1: General form of the Butcher tableau

with maximum polynomial order, p_{\max} , is paired with an explicit Runge-Kutta discretisation of order, $p_{\text{RK}} = p_{\max} + 1$, as discussed in [105]. The chosen penalty term on the LHS of (5.5) is linear and is thus dealt with implicitly, and as such we refer to the resulting temporal discretisation as semi-implicit. The full discretisation of (5.1) using a DG method in space, and a semi-implicit RK method in time can thus be stated as follows: find $\phi_h^n \in V_{\mathbf{p}}(\mathcal{T})$, for $t_e = (0, T)$ such that

$$(\phi_h^n, v_h)_{\mathcal{T}} + \Delta t_e \mu \langle \llbracket \phi_h^n \rrbracket, \llbracket v_h \rrbracket \rangle_{S(\mathcal{T})} = (\phi_h^{n-1}, v_h)_{\mathcal{T}} - \Delta t_e \left[\sum_{i=1}^{N_s} \alpha_i \left(b(\phi_h^{(i)}), v_h \right)_{\mathcal{T}} \right], \quad \forall v_h \in V_{\mathbf{p}}(\mathcal{T}), \quad (5.6)$$

where Δt_e denotes the time step, and $\phi^n(\cdot) = \phi(\cdot, t_e^n)$ denotes the value of ϕ at the n^{th} time step, i.e. $t_e^n = n\Delta t_e$. The terms $\phi^{(i)}$ for $i > 1$ denotes the solution at the intermediate RK stages which can be computed as follows

$$\left(\phi_h^{(i)}, v_h \right)_{\mathcal{T}} + c_i \Delta t_e \mu \langle \llbracket \phi_h^{(i)} \rrbracket, \llbracket v_h \rrbracket \rangle_{S(\mathcal{T})} = (\phi_h^{n-1}, v_h)_{\mathcal{T}} - c_i \Delta t_e \left[\sum_{j=0}^i q_{ij} \left(b(\phi_h^{(j)}), v_h \right)_{\mathcal{T}} \right], \quad \forall v_h \in V_{\mathbf{p}}(\mathcal{T}), \quad (5.7)$$

and thus $\phi^{(1)} = \phi^{n-1}$. The constant N_s denotes the number of stages associated with the Runge-Kutta method of order, p_{RK} , and the constants q_{ij} , α_i and c_i , come from the corresponding Butcher tableau of the form presented in Table 5.1. The specific coefficients for Runge-Kutta schemes of various orders used to compute the example problems in this thesis can be found in Appendix B. In this thesis, spatial discretisations up to a maximum order, $p = 8$, are considered, and as such Runge-Kutta schemes up to and including order $p_{\text{RK}} = 10$ are presented (a 10^{th} order RK scheme is used for meshes where $p_{\max} = 8$, with the usual $p_{\text{RK}} = p_{\max} + 1$, for $p < 8$). It should however be noted that most RK discretisations of a given order, $p_{\text{RK}} > 1$, have demonstrated stability on meshes of elements with polynomial order, p , in the case where, $p > p_{\text{RK}}$, but in such a case the maximum allowable time step which satisfies the CFL condition would have to be reduced, which explains why this choice is made.

Using the matrix notation introduced in Sections 4.4 and 4.5.1, the linear system to be solved at each time step of the evolution equation, can be stated as follows

$$(M + \Delta t_e K_{\text{P}}) \phi^n = M \phi^{n-1} - \Delta t_e F_E, \quad (5.8)$$

where $M = (m_{ij})$ denotes the mass matrix, $K_P = (k_{P,ij})$ denotes what will be referred to as the *penalty matrix*, with elements given by

$$k_{P,ij} = \langle \llbracket v_j \rrbracket, \llbracket v_i \rrbracket \rangle_{S(\mathcal{T})}, \quad (5.9)$$

and $F_E = (f_{E,i})$ is the force vector for the level set evolution problem, with elements given by

$$f_{E,i} = \sum_{j=1}^{N_s} \alpha_j \left(b(\phi_h^{(j)}), v_i \right)_{\mathcal{T}}. \quad (5.10)$$

For a DG spatial discretisation with polynomial order vector, \mathbf{p} , and a Runge-Kutta temporal discretisation of order $p_{\text{RK}} = p_{\text{max}} + 1$, a well known, [105], appropriate time step can be stated

$$\Delta t_e = \min_{\tau \in \mathcal{T}} \left(\frac{h_\tau}{\|b_\tau\|_\infty (2p_\tau + 1)} \right), \quad (5.11)$$

where $\|b_\tau\|_\infty$ is the infinity norm (see Equation (4.54)) of the advection velocity on the element τ , sampled at the integration points.

5.2 Narrow band level set method

The idea of a narrow banded level set method was introduced in Section 3.2.2. The numerical experiments in Sections 4.4.1 and 4.5.3, then provided evidence for the necessity of a narrow band approach. This is because it can be seen that in the case that the level set function is always a signed distance function, and if that signed distance function describes an interface which is a closed loop on a simply connected subdomain, that the level set function is necessarily singular somewhere inside that loop. Problems posed on Cartesian meshes of uniform polynomial order, which have a solution which is known to be singular are unable to demonstrate high-order accuracy. These singular regions in the solution however, will often be far enough away from the level set interface, which is the only important part of the domain, and therefore can ultimately be removed from the domain by only solving the reinitialisation and evolution problems on a narrow band of elements near to the interface. This was demonstrated in the numerical examples presented in Sections 4.4.1 and 4.5.3. This is true unless the interface itself is singular, in which case accuracy will be limited anyway, or the shape described by the level set interface is about to collapse and therefore remove itself from the domain in the near future. In the previous section where a narrow band was required, simple *ad hoc* rules for the narrow band were adopted based on the problem to be solved; this section then describes a more robust approach for the general case.

The best case scenario for a narrow banded level set method, is that the narrow band contain the set of elements intersected by the interface, \mathcal{T}_Γ , as defined in Section 4.3.3, Equation (4.32), and one layer of elements either side of \mathcal{T}_Γ within which the interface could evolve into. For the methodology specific to this thesis, it is useful to extend this definition to include two layers of elements either side of \mathcal{T}_Γ , this will be discussed further in Section 5.2.1. That is the set of elements forming the narrow band, \mathcal{T}_{NB} , is defined as the union of the set of elements cut by

the interface, \mathcal{T}_Γ , the set of elements which aren't cut by the interface but share a node with any element cut by the interface, and the set of elements which share a node with any element which shares a node with any element which is cut by the interface, that is

$$\mathcal{T}_{\text{NB}} = \mathcal{T}_\Gamma \cup \{\tau \in \mathcal{T} : \partial\tau \in \mathcal{E}_{\text{ext}}(\mathcal{T}_\Gamma)\} \cup \{\tau \in \mathcal{T} : \partial\tau \in \mathcal{E}_{\text{ext}}(\{\tau \in \mathcal{T} : \partial\tau \in \mathcal{E}_{\text{ext}}(\mathcal{T}_\Gamma)\})\}. \quad (5.12)$$

It is also useful to define a set of elements constituting the layer of elements just outside of the narrow band, \mathcal{T}_{oNB} . This set of elements is defined as the set of elements which aren't inside the narrow band but which share at least one node with an element inside the narrow band, that is

$$\mathcal{T}_{\text{oNB}} = \{\tau \in \mathcal{T} : (\tau \notin \mathcal{T}_{\text{NB}}) \cup (\partial\tau \in \mathcal{E}_{\text{ext}}(\mathcal{T}_{\text{NB}}))\}. \quad (5.13)$$

For both the level set reinitialisation and the level set evolution problem, the only boundary condition on the domain boundary is a homogeneous Neumann boundary condition. When solving either of these problems on just the narrow band, the Neumann condition extends naturally to the boundary of the narrow band and thus no change needs to be made. The homogeneous Dirichlet boundary condition for the reinitialisation problem is enforced over the set of elements intersected by the level set interface which will always be inside both the full partition and the narrow band, and therefore no change needs to be made to the methods detailed in Sections 4.3.2 and 4.3.3, when solving the reinitialisation problem on the narrow band. The level set evolution equation does not have a Dirichlet boundary condition. In all other cases in this thesis, the boundary conditions where the narrow band is concerned will be stated explicitly.

On an implementation level, to determine if an element is intersected by the level set interface, the value of the level set function is sampled along element edges at the nodes and Gauss points and the sign computed. If any two of the signs are not equal, then the element is determined to be intersected by the interface. Vectors containing element neighbours are generated during the initialisation of the mesh and updated after a given refinement of the mesh, and therefore it is trivial to subsequently decide which elements form the narrow band.

5.2.1 Narrow band extrapolation

The cost of using a narrow band, is that at each iteration of the evolution of the level set function, one has to forfeit the information pertaining to the level set function outside of the narrow band. Thus when necessary, that is when an element which was outside of the narrow band before an iteration of the evolution equation, has now moved inside the narrow band; after that iteration, new information pertaining to the value of the level set function on those elements must be generated. For ease of communication the set of elements which move from outside to inside the narrow band will be denoted as \mathcal{T}_T . As the level set function should always be a signed distance function to the level set interface at any given time, the value of the level set function on the elements in \mathcal{T}_T can be generated by solving the elliptic reinitialisation problem, (4.45), on those elements with the following modifications.

First of all, the level set function on \mathcal{T}_{NB} is already approximately a signed distance function, and as such, so as to avoid the relatively expensive techniques for imposing a Dirichlet boundary

condition on the interface, one can simply use the known value of the level set function on the edge of the narrow band,

$$\phi_D = \phi(\mathbf{x}), \quad \forall \mathbf{x} \in \mathcal{E}_{\text{ext}}(\mathcal{T}_{\text{NB}}), \quad (5.14)$$

as a Dirichlet boundary condition using Nitsche's method. Thus the strong form of what will be referred to henceforth as the *level set extrapolation problem* can be stated

$$\begin{aligned} \nabla \cdot \left(d_3(|\nabla \tilde{\phi}(\mathbf{x})|) \nabla \tilde{\phi}(\mathbf{x}) \right) &= 0, & \mathbf{x} \in \mathcal{T}_T, \\ \tilde{\phi}(\mathbf{x}) &= \phi_D, & \mathbf{x} \in \mathcal{E}_{\text{ext}}(\mathcal{T}_{\text{NB}}), \\ d_3(|\nabla \tilde{\phi}(\mathbf{x})|) \nabla \tilde{\phi}(\mathbf{x}) \cdot \hat{\mathbf{n}} &= 0, & \mathbf{x} \in \partial \mathcal{T}_T. \end{aligned} \quad (5.15)$$

Equation (5.15) can, similar to the Elliptic Reinitialisation method (see Section 4.4), be linearised using Picard's method, and then discretised spatially using the SIPG method leading to a variational formulation which can be stated: find $\tilde{\phi}_h^m \in V_{\mathbf{p}}(\mathcal{T}_T)$ as $m \rightarrow \infty$ such that the following weak form statement of equilibrium is satisfied

$$\begin{aligned} \left(\nabla \tilde{\phi}_h^m, \nabla v \right)_{\mathcal{T}_T} - \left\langle \{ \{ \nabla \tilde{\phi}_h^m \} \}, [v] \right\rangle_{S(\mathcal{T}_T) \cup \mathcal{E}_{\text{ext}}(\mathcal{T}_{\text{NB}})} - \left\langle \{ \{ \nabla v \} \}, [\tilde{\phi}_h^m] \right\rangle_{S(\mathcal{T}_T) \cup \mathcal{E}_{\text{ext}}(\mathcal{T}_{\text{NB}})} \\ + \mu \left\langle [\tilde{\phi}_h^m], [v] \right\rangle_{S(\mathcal{T}_T) \cup \mathcal{E}_{\text{ext}}(\mathcal{T}_{\text{NB}})} = \left((1 - d_3(|\nabla \tilde{\phi}_h^{m-1}|)) \nabla \tilde{\phi}_h^{m-1}, \nabla v \right)_{\mathcal{T}_T} \\ - \left\langle \{ \{ (1 - d_3(|\nabla \tilde{\phi}_h^{m-1}|)) \nabla \tilde{\phi}_h^{m-1} \} \}, [v] \right\rangle_{S(\mathcal{T}_T) \cup \mathcal{E}_{\text{ext}}(\mathcal{T}_{\text{NB}})} - \left\langle \nabla v \cdot \hat{\mathbf{n}}, \tilde{\phi}_D \right\rangle_{\mathcal{E}_{\text{ext}}(\mathcal{T}_{\text{NB}})} \\ + \mu \left\langle \tilde{\phi}_D, v \right\rangle_{\mathcal{E}_{\text{ext}}(\mathcal{T}_{\text{NB}})}, \quad \forall v \in V_{\mathbf{p}}(\mathcal{T}_T). \end{aligned} \quad (5.16)$$

The second modification concerns the fact that the extrapolation equation, (5.16), requires an initial condition which at the time the extrapolation routine is called would not exist. The unique viscosity solution to the reinitialisation problem, (5.15), requires only that the sign of the gradient of the initial level set function at each point is correct, i.e. that the level set function is correctly oriented. For a given element $\tau^+ \in \mathcal{T}_T$, it can be assumed that an approximation of the correct orientation of the level set function on τ^+ is equal to the average of the gradient of the level set function on its neighbour(s) inside the narrow band. An initial condition then, which will allow (5.16) to converge to the desired signed distance function can be generated by solving the following: find $\tilde{\phi}_h^0 \in V_{\mathbf{p}}(\mathcal{T}_T)$ such that the following weak form statement of equilibrium is satisfied

$$\begin{aligned} \left(\nabla \tilde{\phi}_h^0, \nabla v \right)_{\mathcal{T}_T} - \left\langle \{ \{ \nabla \tilde{\phi}_h^0 \} \}, [v] \right\rangle_{\mathcal{E}_{\text{ext}}(\mathcal{T}_{\text{NB}})} - \left\langle \{ \{ \nabla v \} \}, [\tilde{\phi}_h^0] \right\rangle_{\mathcal{E}_{\text{ext}}(\mathcal{T}_{\text{NB}})} + \mu \left\langle [\tilde{\phi}_h^0], [v] \right\rangle_{\mathcal{E}_{\text{ext}}(\mathcal{T}_{\text{NB}})} \\ = \left((1 - d_3(|\overline{\nabla \phi_\tau}|)) \overline{\nabla \phi_\tau}, \nabla v \right)_{\mathcal{T}_T} - \left\langle (1 - d_3(|\nabla \phi_D|)) \nabla \phi_D \cdot \hat{\mathbf{n}}, v \right\rangle_{\mathcal{E}_{\text{ext}}(\mathcal{T}_{\text{NB}})} \\ - \left\langle \nabla v \cdot \hat{\mathbf{n}}, \phi_D \right\rangle_{\mathcal{E}_{\text{ext}}(\mathcal{T}_{\text{NB}})} + \mu \left\langle \phi_D, v \right\rangle_{\mathcal{E}_{\text{ext}}(\mathcal{T}_{\text{NB}})}, \quad \forall v \in V_{\mathbf{p}}(\mathcal{T}_T), \end{aligned} \quad (5.17)$$

where $\overline{\nabla \phi_\tau}$ is the average of the gradient of the level set function at all of the Gauss points inside all of the elements which are both currently inside the narrow band and share a node with a given element $\tau \in \mathcal{T}_T$; which has been substituted for the gradient at the $m - 1^{\text{th}}$ iteration on

the RHS.

One possible issue with generating the initial condition in this way is that near to a singular region, the initial level set function might not be correctly oriented (which might be the case where an interface is about to collapse for example). If the narrow band is defined (as mentioned in Section 5.2) to be just one element wide either side of \mathcal{T}_Γ , then the solution on the element which isn't correctly oriented could now potentially cross the zero-plane (especially on a hp -adaptive mesh where the elements in \mathcal{T}_T could be physically larger than those inside the narrow band). In such a case there would therefore be a new interface erroneously generated during the extrapolation. This is the reason for defining the narrow band as in (5.12), that is, two layers of elements wide either side of \mathcal{T}_Γ as opposed to just one, as such a choice ensures any element to be extrapolated on will always be far enough away from the level set interface such that even if the level set function is incorrectly oriented it will not interfere with the level set interface, given the local bound on variation in element size.

As has hopefully been made apparent, for the proposed level set methodology in this thesis, it is imperative that the level set function satisfies the signed distance property. The extrapolation problem is in general a much cheaper problem to solve than the full reinitialisation problem, as the number of elements in \mathcal{T}_T is often significantly fewer than in the narrow band, \mathcal{T}_{NB} , as well as the fact that it is coupled with the simpler method for enforcing the Dirichlet boundary condition. It is generally cheaper therefore to ensure that the level set function on these new elements can, as strictly as the spatial discretisation allows, satisfies the Eikonal equation, (3.4), rather than attempting to save time here by limiting the number of iterations on the extrapolation and then trying to correct the signed distance-ness of the level set function later by reinitialising everywhere in the narrow band. As such the stopping criteria for the extrapolation problem are chosen to be strict in order to ensure that the level set function generated through the extrapolation is a strict signed distance function. Specifically, there are two stopping criteria used for the extrapolation problem: the Euclidean norm of the residual between the two most recent consecutive solutions over the computational domain to the extrapolation problem is smaller than a tolerance, $\|\tilde{\phi}_h^m - \tilde{\phi}_h^{m-1}\|_2 < 10^{-13}$; or a maximum number of iterations have been computed, where that maximum is set to 10000. It should be noted that despite the relatively slow convergence of the Picard scheme, it is unlikely that the required number of iterations would approach such a maximum and thus this limit is in place as a fail-safe.

5.3 Anderson acceleration

One bottleneck in the simulations computed using the level set methodology presented herein is the slow convergence rate of the Picard iterative method used to linearise both the reinitialisation and extrapolation equations. This is especially true given the requirement to return a solution with the desired high level of accuracy. To help combat this issue, an Anderson acceleration algorithm has been applied wherever a fixed point iterative method is required. The idea of Anderson acceleration is that after solving the appropriate linear system at each iteration of the fixed point method, the computed solution is modified to include information from a number of the most recent iterations of the computed solution by performing a line search to find a linear

combination of these solutions which minimises the residual between them. By decreasing this residual, which ultimately defines the convergence criterion, convergence can be achieved more quickly. Such an approach which can be used in the solution of the Elliptic Reinitialisation equations (4.20) or level set extrapolation equations, (5.16), is presented in Algorithm 1. The form of the Anderson acceleration algorithm used in this thesis is equivalent to that presented in [156, 157], and the information specific to this thesis is shown in Algorithm 1 and explained in the remainder of this section.

The first step of Algorithm 1 is to initialise some parameters; `solutionOld` which is a vector containing the solution at the previous iteration; `solChangeHistory` which is a matrix, each column of which contains a vector of residuals between previous solutions and is thus initialised as an empty matrix; `resChangeHistory` which is a matrix each column of which contains a residual vector between previous residual vectors to be explained presently; and `stepLength` which denotes the solution to the least squares problem forming the line search, and which therefore will be a vector containing ‘weights’ associated with each of the previous residuals contained within `solChangeHistory`. After this, the appropriate LHS system matrix can be computed; this depends on the problem being solved, reinitialisation or extrapolation, which is defined using the `problemHandle` input parameter.

After this the Anderson loop can begin. The first step of the loop at each iteration is to compute a new solution to the problem being solved, `solutionNew`. Again this requires the `problemHandle` input parameter to point towards the correct construction of a RHS vector and the linear system to be solved. Once the solution at the current iteration has been computed, `solutionNew`, the residual, `residualNew`, between the current and previous solutions can be computed. In this way there will always be one fewer previous residuals than previous solutions at a given iteration. As is explained in [157], a more efficient implementation of the least squares problem forming the line search can be achieved if the minimisation is in the change in the residual, as opposed to the direct minimisation of the residuals themselves. For this reason at least two residuals are required to perform the line search, and thus during the first iteration of the Anderson loop, that is when $m = 0$, no line search is computed. As such the remainder of the first iteration consists of checking the stopping criteria, and should they not be satisfied updating the variable `solutionOld` and initialising the variable `residualOld` for the next iteration.

Beyond the first iteration, when $m > 0$, the loop begins again by computing a new solution and the residual between it and the previous solution and checking the convergence criterion. Should the convergence criterion not be satisfied on this and any future iterations, then the line search will be computed. The number of previous iterated solutions to be included in the line search is defined by the parameter, `lineSearchSize`. Initially the value of the `lineSearchSize` parameter will be determined by the current number of iterations, and this can, with future iterations, grow up to a user decided maximum, defined by the input parameter `maxLineSearchSize`. In deciding the value of the parameter `maxLineSearchSize` one needs to consider that increasing this value should reduce the number of required iterations, but will also increase the computational expense of each iteration (as there will a

Input: $\tilde{\phi}^0$, maxLineSearchSize, maxIt, problemHandle, errorHandle, resTol, maxErr
Output: $\tilde{\phi}^\infty$, recomputeFlag

```

solutionOld =  $\tilde{\phi}^0$ ;
solChangeHistory = [];
resChangeHistory = [];
stepLength = 0;
Compute LHS matrix,  $K$ ;
for  $m \leftarrow 0$  to  $maxIt-1$  do
    Compute RHS force vector,  $F$ ;
    solutionNew =  $K^{-1}F$ ;
    residualNew = solutionNew -  $\tilde{\phi}^m$ ;
    if  $\|residualNew\|_2 < resTol$  then
        | return solutionNew, 0;
    end
    if  $m > 0$  then
        | lineSearchSize = min{maxLineSearchSize,  $m$ };
        | residualChange = residualNew - residualOld;
        | resChangeHistory = [resChangeHistory, residualChange];
        | while  $condest(resChangeHistory(:, m-lineSearchSize-1 : m)) > 10^{10}$  do
            | | lineSearchSize = lineSearchSize - 1;
        | end
        | Find stepLength = (stepLength[0], ..., stepLength[lineSearchSize-1])T
        | which solves min  $\|residualNew -$ 
        |  $resChangeHistory(:, m-lineSearchSize-1 : m) * stepLength\|_2$ ;
        | solutionChange = solutionNew - solutionOld;
        | solChangeHistory = [solChangeHistory, solutionChange];
    end
     $\tilde{\phi}^{m+1} =$  solutionNew -
    solChangeHistory(:,  $m-lineSearchSize-1 : m$ ) * stepLength;
    Compute the signed distance error,  $E_{SD}$ ;
    if  $E_{SD} > maxErr$  then
        | return  $\tilde{\phi}^0$ , 1;
    end
    solutionOld = solutionNew;
    residualOld = residualNew;
end
return  $\tilde{\phi}^{maxIt-1}$ , 0 ;

```

Algorithm 1: Anderson acceleration algorithm for level set reinitialisation or extrapolation.

larger system to solve) and as such an appropriate balance may need to be sought between the two. In this work, the parameter is chosen such that `lineSearchSize=maxIt`, i.e. at each iteration `lineSearchSize = m`, as it has been found that the problem generally satisfies a stopping criterion before the size of the linear system becomes problematic in terms of computational efficiency. After this the change in the residuals, `residualChange`, can be computed, and appended to the `resChangeHistory` matrix. At this point a check is made that the matrix, `resChangeHistory(:,m-lineSearchSize-1:m)`, that is the matrix containing only the ‘`lineSearchSize`’ most recent `residualChange`’s, is not ill-conditioned, which can happen as `lineSearchSize` grows with the number of iterations. This check is done using the MATLAB built-in function `condest`, and should the conditioning be poor, that is, `condest(resChangeHistory(:,m-lineSearchSize-1:m)) > 1010`, then `lineSearchSize` can be reduced until that condition can be satisfied. This is another reason why choosing the parameter `maxLineSearchSize` to be arbitrarily large is unlikely to cause problems. At this point the line search can be performed which will return the vector `stepLength`, which is a set of weights which correspond to each entry in the `solChangeHistory(:,m-lineSearchSize-1:m)` matrix. Here the least squares problem is solved using a QR decomposition approach described in [157], which is efficient as the factors of the decomposition can be updated at each iteration to take into account the new residual vector computed at each iteration. It is noted here that in Algorithm 1, the symbol, `*`, denotes matrix-vector multiplication. The computed solution can then be updated to take into account the residual between previous consecutive iterations weighted appropriately to minimise those residuals. The loop then continues until a stopping criterion is satisfied.

There are two stopping criteria used in Algorithm 1. Firstly, a convergence criterion is used, defined by the Euclidean norm of the residual between two iterations satisfying a given tolerance, `residualNew<resTol`, which if satisfied triggers the return of the the solution at the current iteration as the converged solution to the problem. Secondly a stopping criterion is provided to prevent excessive computation in the event that the solver stagnates, in such a case computation can be ceased once a user defined maximum number of iterations having been computed, `m >maxIt`, which triggers the return of most recent solution as the converged solution. The values `resTol`, and `maxIt` will in this thesis be chosen differently for the reinitialisation and extrapolation problems, and thus will be chosen as explained in Sections 4.4 and 5.2.1 respectively, unless otherwise stated explicitly for a given example problem.

Whilst Anderson acceleration seems to be effective in reducing the number of iterations for problems with smooth solutions, by attempting to increase the rate of convergence of the Picard iteration, the Anderson acceleration algorithm can introduce overshoots and therefore instability into the underlying method. This is particularly problematic when the level set function contains a singularity. As such if the error is deemed to have grown to an unacceptable level, defined as $E_{SD} > \text{maxErr}$, the loop will break and return the initial input level set function $\tilde{\phi}^0$ as well as a flag, `recomputeFlag = 1`, indicating that the reinitialisation should be recomputed using a standard Picard iteration, a method which has demonstrated nothing but stability throughout this work.

5.4 A narrow banded discontinuous Galerkin level set method

This section will present how these components of the level set methodology detailed in Sections 4.4, 5.1.2 and 5.2 work together to form a high-order accurate methodology for solving level set problems, as well as a number of numerical examples to demonstrate the efficacy of the proposed methodology.

5.4.1 Algorithm

A problem to be solved using the level set methodology proposed, on a Cartesian mesh of elements of size h , with uniform polynomial order, p , will take the form presented in Algorithm 2. The first step of the algorithm is to partition the domain, Ω , into a mesh, \mathcal{T} . The initial level set function, $\phi^0(\mathbf{x})$, can then be L^2 projected onto \mathcal{T} , to get a computational initial condition, ϕ_h^0 . After this, the narrow band can then be generated using the procedure described in Section 5.2. If at this stage it is known that the initial level set function, $\phi^0(\mathbf{x})$, was not a signed distance function, then the computed level set function, $\phi_h^0(\mathbf{x})$, can be reinitialised.

```

Input:  $h, p, \Omega, \phi^0, \text{advectionType}, T$ 
Output:  $\phi_h(\mathbf{x}, t_e)$ 
Initialise problem parameters:  $\phi_h^0, \mathcal{T}$ ;
Compute initial Narrow Band,  $\mathcal{T}_{\text{NB}}$ ;
if  $\phi_h^0$  is not a signed distance function then
  | Reinitialise Level set function
end
while  $t_e^m < T$  do
  | Compute Advection Velocity Vector,  $\mathbf{b}$ ;
  | Compute time step,  $\Delta t_e$ ;
  | Evolve Level Set Interface by solving (5.6),  $\phi_h^m(\mathbf{x})$ ;
  | Reinitialise level set function,  $\phi_h^m(\mathbf{x})$ ;
  | Update narrow band,  $\mathcal{T}_{\text{NB}}$ ;
  | if elements move from outside to inside narrow band then
  |   | Initialise the level set function on those elements,  $\phi_h^m(\mathbf{x})$ ;
  | end
  |  $t_e^{m+1} = t_e^m + \Delta t_e$ ;
end

```

Algorithm 2: Narrow band level set evolution algorithm for problems on Cartesian meshes of elements of uniform polynomial order.

The first step in the evolution loop is to compute the advection velocity for the current iteration. If this was to be a problem driven by physics, then the physical problem would be computed at this stage. Similarly, this could be a prescribed value, or be a function of the level set function itself. This information would need to be decided by the user prior to computation and passed into the algorithm, through the `advectionType` parameter. The time step for the current iteration, Δt_e , can then be computed using (5.11). The evolution equation (5.6) can then be solved to determine the position of the level set interface at time, t_e^m . The next step is to reinitialise the level set function, by solving the Elliptic Reinitialisation problem (4.46). Whilst it should be the case that the level set interface is fixed beyond the evolution step, it is conceivable

that there are small erroneous movements of the level set interface during reinitialisation, which could have an affect on which elements comprise the narrow band. Furthermore, it is beneficial that the gradient as much as possible satisfies the signed distance constraint prior to computing the extrapolation of the level set function, as both the initial condition, and also the terms forming the RHS of the extrapolation equation require ‘correct’ information concerning the value of the level set function and its gradient on the boundary elements inside the narrow band. Therefore, whilst it might seem more logical to update the narrow band immediately after the evolution such that reinitialisation is only computed on those elements, it is beneficial in terms of both accuracy and efficiency to order the reinitialisation immediately after the evolution step. After the reinitialisation, the narrow band can then be updated using the same procedure described in Section 5.2. If elements move from outside to inside the narrow band as a result of the evolution, then the extrapolation procedure described in Section 5.2.1 can be used to initialise the level set function on these elements. The loop then continues until $t_e^m = T$, at which point the level set function at each time step is outputted to the user.

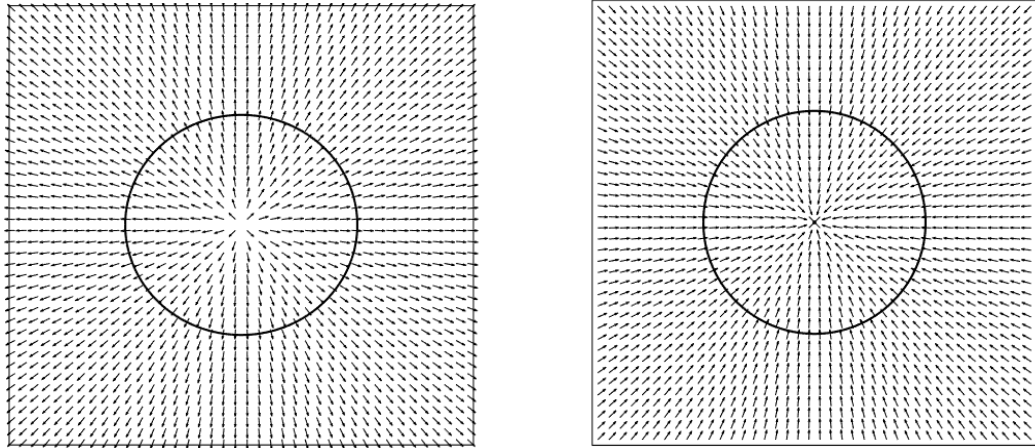
5.4.2 Narrow banded discontinuous Galerkin level set method: numerical examples

In order to demonstrate the efficacy of the proposed method a number of numerical examples have been computed. In all cases the examples will be computed using a mesh of square elements of uniform size and uniform polynomial order. The examples will be repeated for a series of meshes of different h , for each p until a given point in pseudotime, $t_e = T$, at which various error measures will be presented, see Section 4.4.1.1.

The first example is that of a circle growing at a constant rate until a given time, shrinking with the same constant rate until a second given time, and then growing again until it reaches its original position. Images demonstrating the velocity fields for this example can be seen in Figure 5.1. This is a very simple example, by which the constant velocity field should allow the level set function to evolve as a rigid body in space maintaining its shape. As such any variation in the shape of the interface between the initial and final iteration can only be as a result of reinitialisation. Nevertheless, given the simplicity of the example, it is a good starting point for the demonstration of optimal orders of convergence using the method.

A second example problem consists of a circle undergoing a translation across the domain from left to right. In this example, the advection velocity is equal to the magnitude of the x-component of the gradient of the level set function, as can be seen in Figure 5.2. This means that the advection velocity vector is to be dealt with explicitly in this example, and furthermore that it is imperative that the level set interface maintain its shape and its satisfaction of the signed distance property during the evolution such that the advection velocity can be accurately computed at each step. However, such a velocity field will cause the gradient of the level set function to change during each evolution step, particularly in the area where the interface velocity is close to zero, see Figure 5.2, and thus the combined efficacy of the evolution and the reinitialisation is tested in this example.

A third example, taken from [158], is computed by which a circular interface is evolved subject to a flow field produced by homogeneous strain. More specifically the advection velocity vector



(a) Advection field driving growth at a constant rate. (b) Advection field driving shrinking at a constant rate.

Figure 5.1: Advection velocity field for the growing and shrinking circle on a square domain.

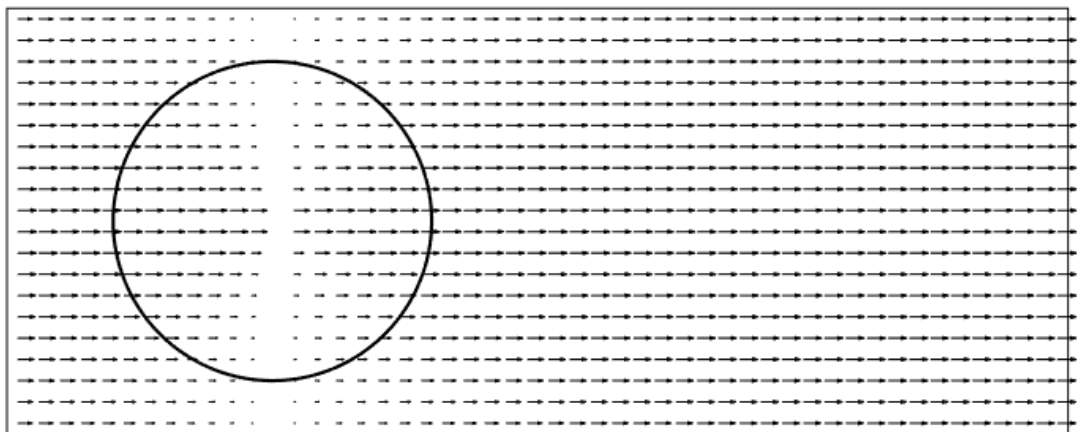


Figure 5.2: Advection velocity field for the translating circle problem.

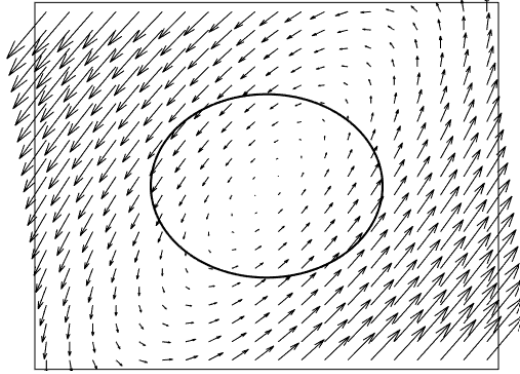
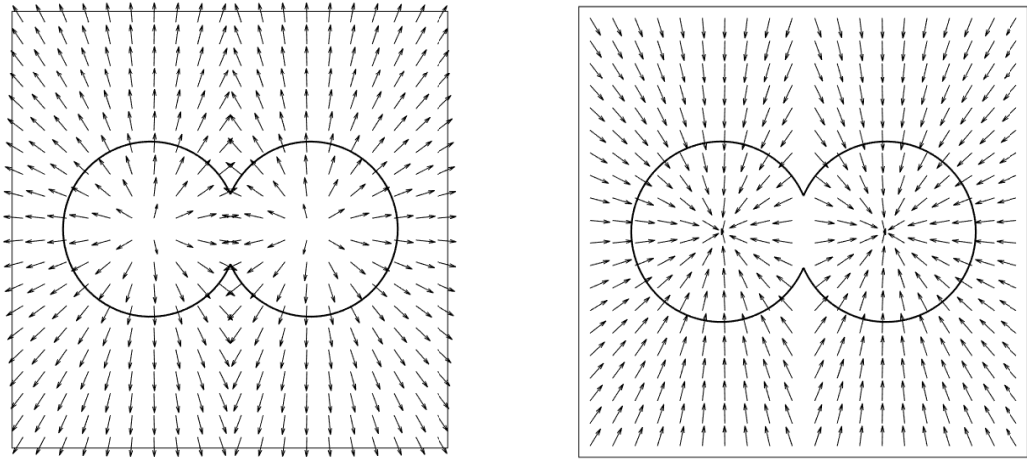


Figure 5.3: Advection velocity field for the circle under homogeneous strain.



(a) Advection field driving growth at a constant rate. (b) Advection field driving shrinking at a constant rate.

Figure 5.4: Advection velocity field for the growing and shrinking of the two merging circles problem.

takes the following form

$$\mathbf{b} = \begin{Bmatrix} (x - y)\nabla_x\phi \\ (2x - y)\nabla_y\phi \end{Bmatrix}, \quad (5.18)$$

which can be conceptualised as a circle undergoing a shear, a stretch and a rotation. The corresponding advection field can be seen in Figure 5.3. This example contains the added difficulty that the curvature of the level set function changes over time as the circle shears and becomes more elliptical. Similarly as the interface begins to approach itself as the minor axis shrinks over time, the singular part of the level set function inside the shape, is likely to move inside the narrow band further increasing the difficulty of the problem. Another point of note here is that the velocity field in this case is divergence free and as such the area of the shape should be constant, which is useful as a measure of error.

The final example begins with two circles of equal size which grow at a constant rate until a given time, t_1 . At some point prior to t_1 , the two circles will meet and then will merge becoming

one shape. At time t_1 , the shape will then begin to shrink. The corresponding advection fields can be seen in Figure 5.4. As the shape shrinks one should expect the diffusion of the method to smooth out the singularity at the intersection of the original two circles creating a dumbbell shape that should maintain until $t_e = T$. Any splitting of this shape during that time between t_1 and T would therefore be merely a numerical artefact.

5.4.2.1 h-convergence study: growing-shrinking-growing circular interface

A level set function which is the signed distance function describing a circular interface with radius, $r = 1$, that is

$$\phi^0 = \sqrt{x^2 + y^2} - r, \quad (5.19)$$

is L^2 projected onto the domain, $\Omega = (-2, 2)^2$. The level set function is evolved, from time $t_e = (0, 0.6)$, subject to a prescribed advection velocity vector the magnitude of which is defined as follows

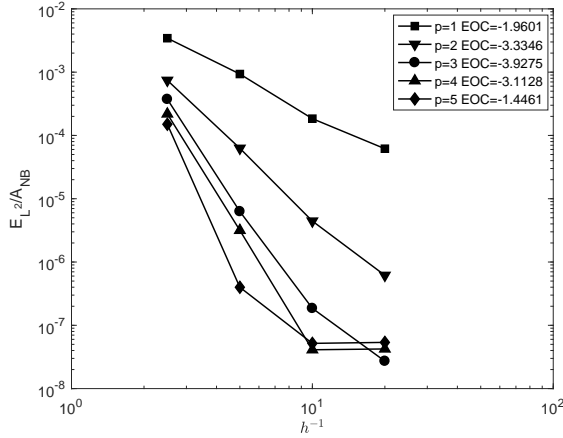
$$b = \begin{cases} -1, & t_e < 0.15, \\ 1, & 0.15 \leq t_e < 0.45, \\ -1, & t_e \geq 0.45. \end{cases} \quad (5.20)$$

That is, a circular interface which grows at a constant rate from time, $t_e = (0, 0.15]$, then shrinks at the same constant rate from time, $t_e = (0.15, 0.45]$, before growing again at the same constant rate from time $t_e = (0.45, 0.6]$. At $t_e = T = 0.6$, the interface should be back to the position of the initial level set interface projected on the mesh at time, $t_e = 0$. The analytical solution for all time t_e during the evolution can thus be computed as

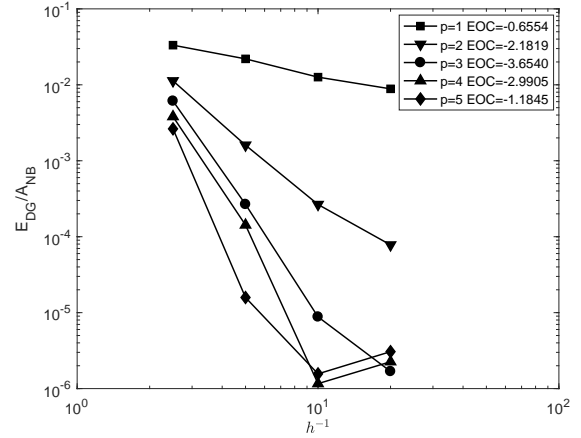
$$\phi(\mathbf{x}, t_e) = \begin{cases} \sqrt{x^2 + y^2} - (r + t_e), & t_e < 0.15, \\ \sqrt{x^2 + y^2} - (r + 0.3 - t_e), & 0.15 \leq t_e < 0.45, \\ \sqrt{x^2 + y^2} - (r - 0.6 + t_e), & t_e \geq 0.45. \end{cases} \quad (5.21)$$

This problem will be computed on a series of uniform Cartesian meshes with square elements of size, $h = 0.4, 0.2, 0.1, 0.05$, and of uniform polynomial order, $p = 1, 2, 3, 4, 5$, and the error at $t_e = 0.6$ will be reported. For the purposes of accurately computing errors, at the time step where $t_e + \Delta t > 0.15, 0.45, 0.6$, the time step is reduced such that the level set function at these instances (i.e. $t_e = 0.15, 0.45, 0.6$ (to machine precision)) are captured.

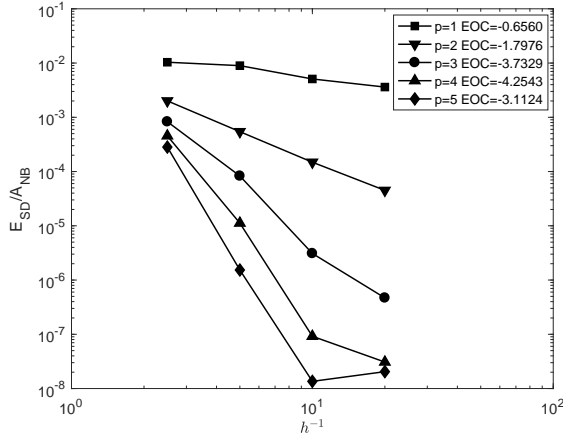
Figure 5.5 shows error values for the growing-shrinking-growing (GSG) problem on all of the meshes considered at time, $t_e = T = 0.6$ and convergence rates for each of the polynomial orders, using the L^2 , DG and signed distance and interface error measures, defined in Section 4.4.1.1. As the size of the computational domain will be different for each partition in the convergence study by virtue of the narrow band, all of the reported errors will be normalised by the area of the narrow band. The convergence rates for all polynomial orders and all error measures are computed using the difference in the error between the meshes where $h = 0.2$ and $h = 0.05$. As this example problem is so simple it is a good benchmark to see what the best case scenario might be when it comes to convergence rates in the various norms. From Figure 5.5 it can be observed that the error in the L^2 norm over the domain, decreases roughly on the order of



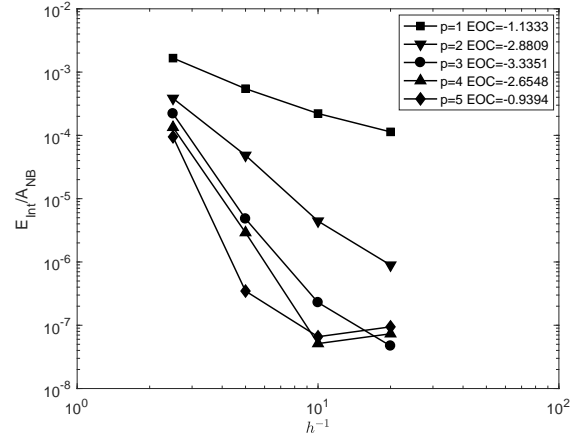
(a) Convergence in the L^2 norm.



(b) Convergence in the DG norm.

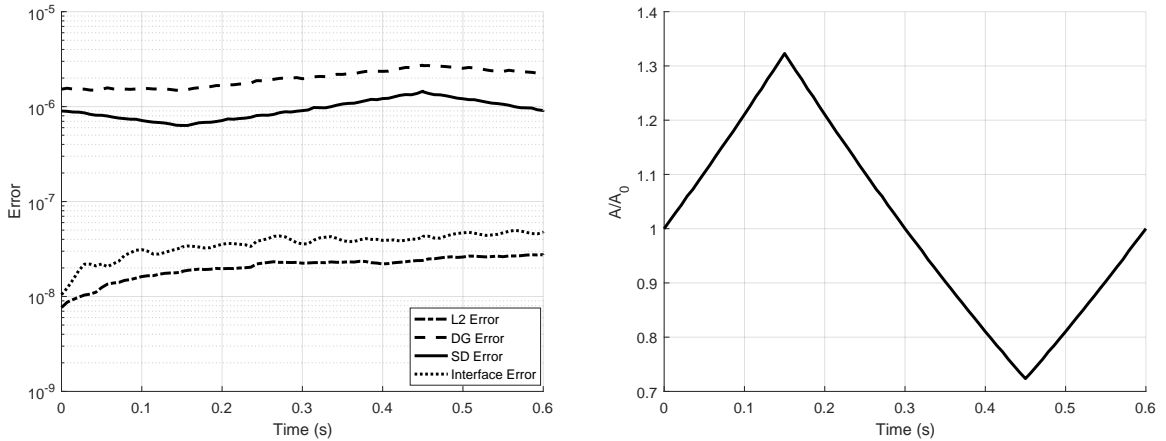


(c) Convergence using the signed distance error measure.



(d) Convergence using the interface error measure.

Figure 5.5: Error data normalised by narrow band area, A_{NB} , and associated convergence rates for the GSG circular interface evolution problem on narrow banded Cartesian meshes at time, $t_e = 0.6$.



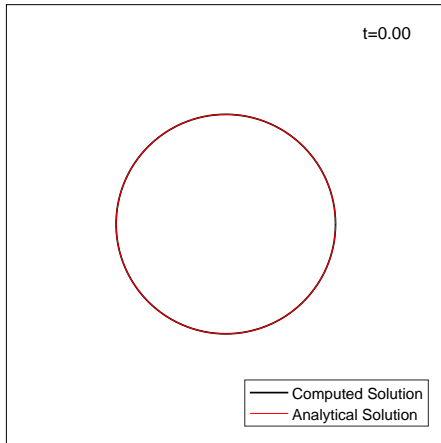
(a) Error against pseudotime in the relevant norms.

(b) Area ratio against pseudotime.

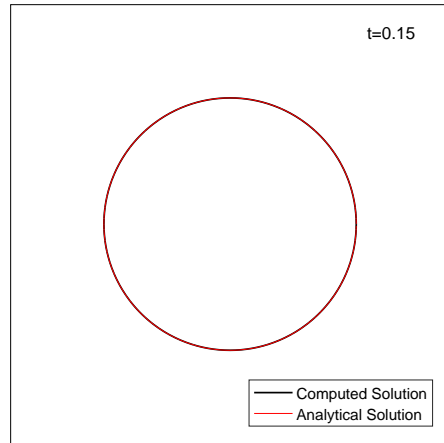
Figure 5.6: Error and area ratio over time for the GSG circular interface evolution problem at time, $t_e = 0.6$, on a narrow banded Cartesian mesh where $h = 0.05$ and $p = 3$. The area ratio compares the area of the shape enclosed by the interface, A , with the area contained inside the initial interface, A_0 .

h^{p+1} , for $p = 1, 2, 3$. For higher order polynomial bases, the accuracy of the solution approaches machine precision which ultimately limits the rate of convergence beyond this point. For both the DG norm and the signed distance error measure the rate of convergence is approximately on the order of $h^{\frac{3p}{2}-1}$; as both are measures related to the H^1 seminorm, it is unsurprising that the error in both of these decays at a similar rate. Again this is true in the region before machine precision is reached. The L^2 error at the interface converges at a slightly slower rate than the error in the entire domain, and can be stated as approximately of the order, h^p .

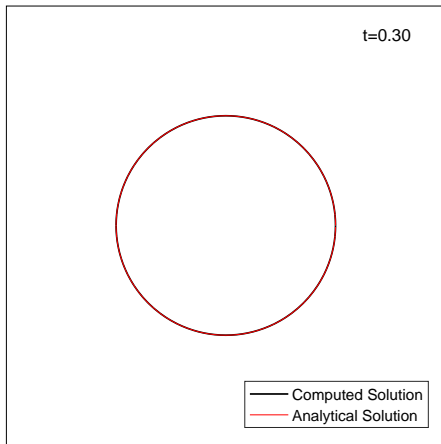
Figure 5.6(a) shows the variation in the L^2 , DG and signed distance and interface error measures, over time for the GSG problem computed on the mesh where $p = 3$, and $h = 0.05$. The L^2 error over the domain, and the L^2 error at the interface are both similar in magnitude and grow with time in a similar manner. The relative position of the interface to the grid has an effect on the computation of the error at the interface, and this can be seen in the slight oscillations in this error measure over time. Specifically this is to do with points where the interface passes close to mesh nodes or aligns with element edges and thus the portion of the interface inside some elements can be much smaller than the size of the element itself (these elements are sometimes referred to in the literature as small cut elements). This can also have an effect on the degree to which the Dirichlet boundary condition is enforced, although this effect does not seem to be significant in this example problem. One interesting note is that as the circle grows over time, the curvature of the level set function inside the narrow band shrinks, (as $\kappa \propto r^{-2}$ for a circular interface, where κ denotes curvature), this is reflected in the signed distance error measure which improves as the circle grows and worsens as the circle shrinks and thus the curvature increases. This is due to the fact that the solution to which the level set reinitialisation method converges, upon which the signed distance error depends, will be better resolved for a fixed mesh size when the curvature of the interface is smaller. This is also reflected



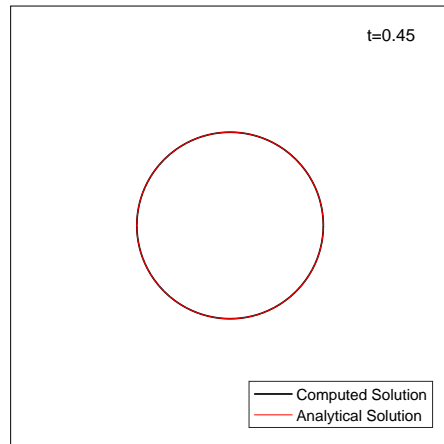
(a) Relative position of domain and interface at $t_e = 0.00$.



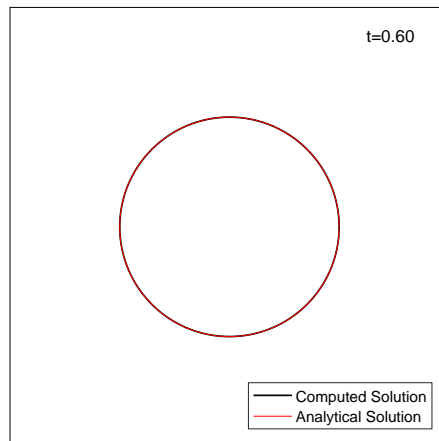
(b) Relative position of domain and interface at $t_e = 0.15$.



(c) Relative position of domain and interface at $t_e = 0.30$.



(d) Relative position of domain and interface at $t_e = 0.45$.



(e) Relative position of domain and interface at $t_e = 0.60$.

Figure 5.7: Computed and analytical interface position over time for the GSG circular interface evolution problem, on the narrow banded Cartesian mesh where $h = 0.05$ and $p = 3$.

somewhat in the DG norm, although to a lesser degree.

Finally, in Figure 5.7, the computed and analytical interface for the GSG problem is plotted on the domain, at multiple instants in time from $t_e = 0$ to $t_e = 0.6$, for the mesh where $p = 3$ and $h = 0.05$. It can be seen that there is no noticeable deviation between the computed solution and analytical solution, as would be expected for such small reported errors.

5.4.2.2 h-convergence study: translating circular interface

A level set function which is the signed distance function describing a circular interface with radius, $r = 0.15$, that is

$$\phi^0 = \sqrt{(x + 0.25)^2 + y^2} - r, \quad (5.22)$$

is L^2 projected onto the domain, $\Omega = (-0.5, 0.5) \times (-0.2, 0.2)$. The level set function is evolved, from time $t_e = (0, 0.5)$, subject to an advection velocity vector the magnitude of which is defined as follows

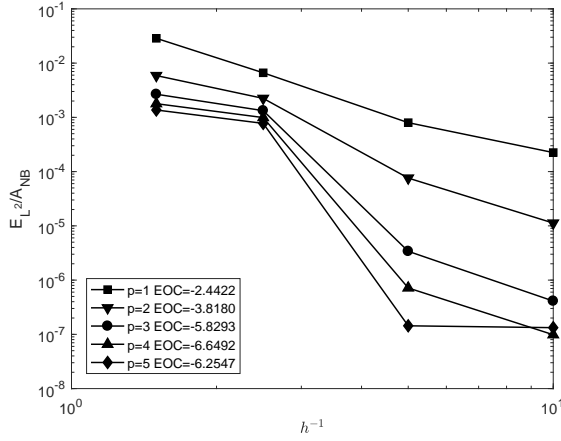
$$b = -\frac{\nabla_x \phi_h}{|\nabla \phi_h|}. \quad (5.23)$$

That is, a circular interface centred at $\mathbf{x} = (-0.25, 0)$ which translates in the positive x -direction to $\mathbf{x} = (0.25, 0)$, at $t_e = 0.5$, whilst maintaining its shape. The analytical solution for all time t_e during the evolution can thus be computed as

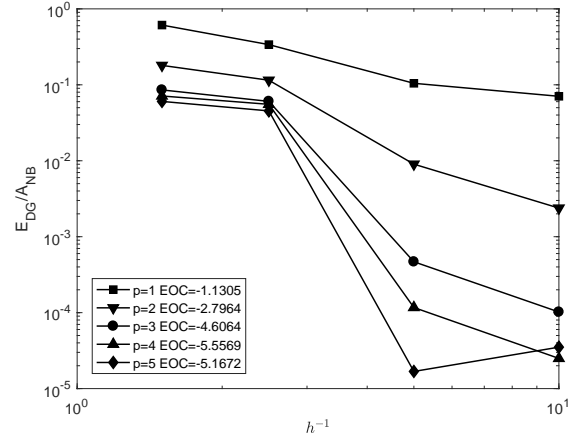
$$\phi(\mathbf{x}, t_e) = \sqrt{(x - 0.25 + t_e)^2 + y^2} - r. \quad (5.24)$$

This example problem will be computed on a series of uniform Cartesian meshes with square elements of size, $h = 0.0667, 0.04, 0.02, 0.01$, and of uniform polynomial order, $p = 1, 2, 3, 4, 5$, and the error at $t_e = 0.5$ will be reported. For the purposes of accurately computing errors, at the time step where $t_e + \Delta t > 0.5$, the time step is reduced such that the level set function at this instance (i.e. $t_e = 0.5$ (to machine precision)) is captured.

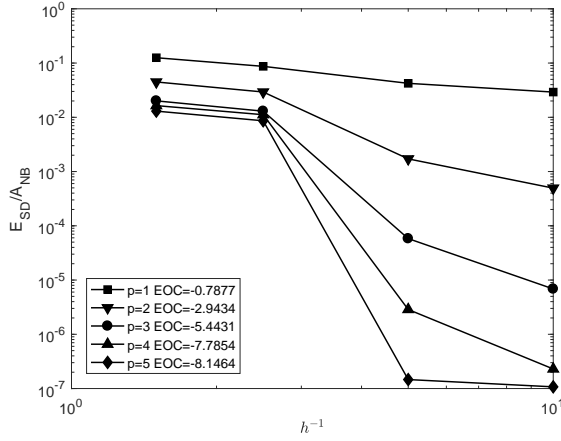
Figure 5.8 shows normalised errors for the translating circle problem on all of the meshes at time, $t_e = T = 0.5$ and convergence rates for each of the polynomial orders, using the L^2 , DG and signed distance and interface error measures, defined in Section 4.4.1.1. One immediate note of difference between the error data for this example in comparison to the previous example presented in Section 5.4.2.1, is that for all polynomial orders, the rate of convergence between the two coarsest meshes (that is the meshes where $h = 0.0667$ and $h = 0.04$), for all of the error measures tested is constant, and equal to the rate of convergence in the linear case. The reason for this is that for the criterion defining the narrow band, (5.12), on a mesh where $h = 0.0667$ describing a circular interface of radius, $r = 0.15$ isn't sufficiently narrow to remove from the domain the singularity present in the signed distance function describing the interface. As such, the solution does not satisfy the smoothness condition discussed in Section 4.4.1.2 required to demonstrate high-order convergence between these two meshes. Beyond this point it can once again be seen that the order of convergence increases with p for all of the error measures until machine precision is reached which limits further increases. For this example problem, the experimental orders of convergence are generally slightly better than those quoted from the previous example. Again for all polynomial orders and for all error measures, the



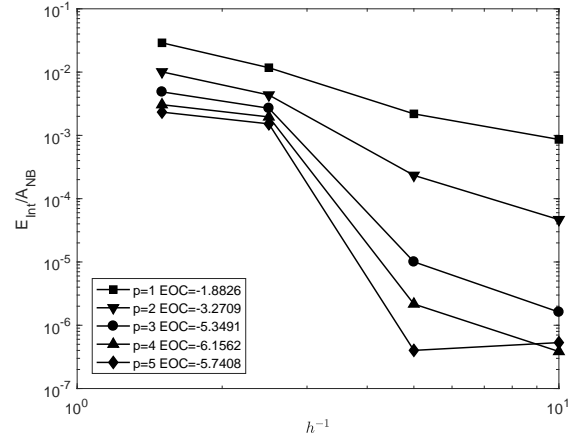
(a) Convergence in the L^2 norm.



(b) Convergence in the DG norm.

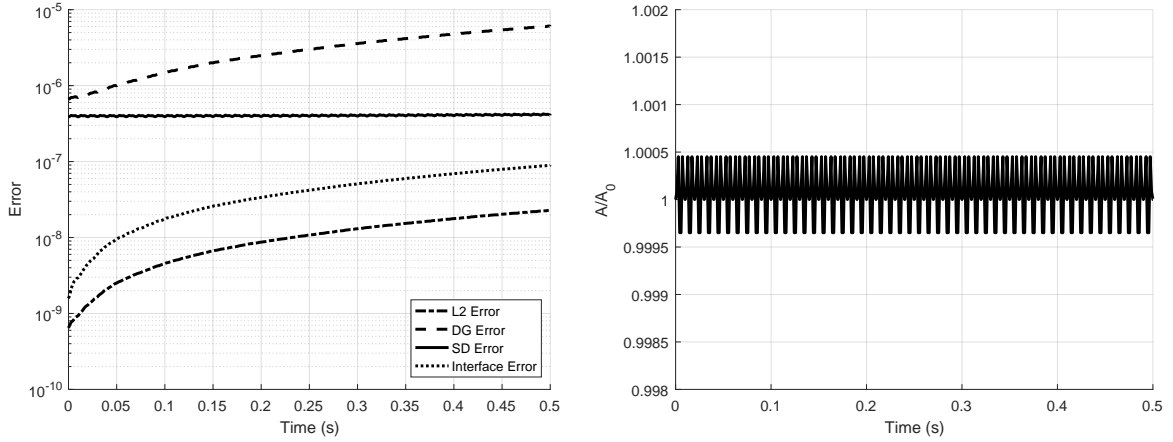


(c) Convergence using the signed distance error measure.



(d) Convergence using the interface error measure.

Figure 5.8: Error data normalised by narrow band area, A_{NB} , and associated convergence rates for the translating circular interface problem on narrow banded Cartesian meshes at time, $t_e = 0.5$.



(a) Error against pseudotime in the relevant norms.

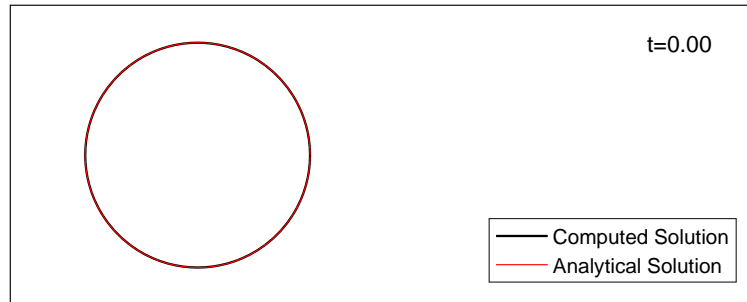
(b) Area ratio against pseudotime.

Figure 5.9: Error and area ratio over time for the translating circular interface problem on the narrow banded Cartesian mesh where $h = 0.01$ and $p = 3$. The area ratio compares the area of the shape enclosed by the interface, A , with the area contained inside the initial interface, A_0 .

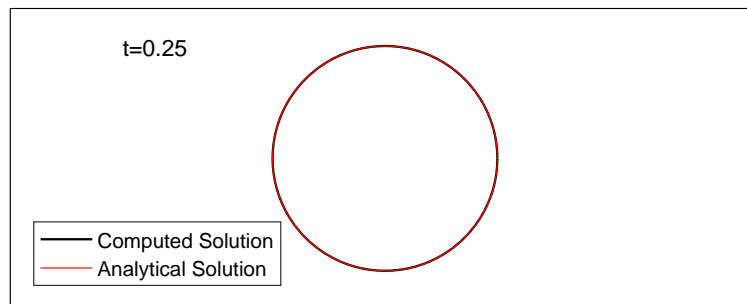
quoted experimental order of convergence is computed between the meshes where $h = 0.04$ and $h = 0.01$. The error in the L^2 norm decreases approximately proportional to $h^{\frac{3p}{2}+1}$. The error in the DG and signed distance norms decreases approximately proportional to $h^{\frac{3p}{2}}$. The interface error decreases approximately proportional to $h^{\frac{3p+1}{2}}$.

Figure 5.9(a) shows the error over time for the translating circle problem on the mesh where $p = 3$ and $h = 0.01$. Many of the patterns noted for the previous example are echoed here. The error in the L^2 norm over the domain, has the smallest magnitude throughout the simulation, increasing over time at the same rate as the L^2 error at the interface. The circle in this problem has constant size, and since the signed distance error is that which is enforced using the reinitialisation and extrapolation methods, the error in this measure remains roughly constant throughout the simulation. The relative position between the interface and the background grid is again responsible for the oscillations here. As can be seen in Figure 5.9(b), there is a pattern in the how the area varies over time, this is because there is a repeating pattern in the relative position between the interface and the grid due to the constant time step present in this example. Again, the variation in the area (and thus the radius and curvature of the circular interface) is reflected in the signed distance error. The magnitude of the error computed using the DG norm is again the largest. The signed distance error tends to bound the error in the L^2 norm from above and the error in the DG norm from below.

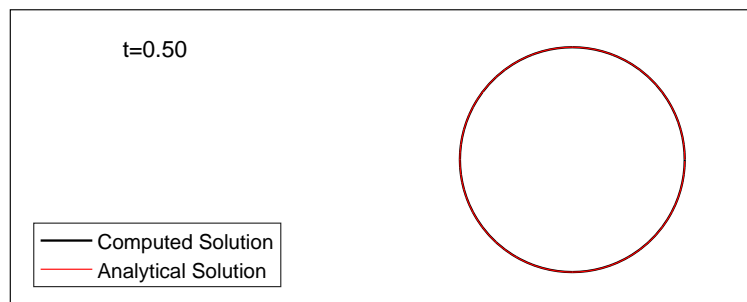
In Figure 5.10, the computed and analytical interface for the translating circle problem is plotted on the domain, at multiple instants in time from $t_e = 0$ to $t_e = 0.5$, for the mesh where $p = 3$ and $h = 0.01$. Again for such small errors the computed and analytical solutions align almost exactly.



(a) Relative position of domain and interface at $t_e = 0.00$.



(b) Relative position of domain and interface at $t_e = 0.25$.



(c) Relative position of domain and interface at $t_e = 0.50$.

Figure 5.10: Computed and analytical interface position over time for the translating circle problem, on the narrow banded Cartesian mesh where $h = 0.01$ and $p = 3$.

5.4.2.3 h-convergence study: shearing circular interface

A level set function which is the signed distance function describing a circular interface with radius, $r = 0.5$, that is

$$\phi^0 = \sqrt{x^2 + y^2} - r, \quad (5.25)$$

is L^2 projected onto the domain, $\Omega = (-1, 1)^2$. The level set function is evolved, from time $t_e = (0, 0.5)$, subject to an advection velocity vector, (5.18), the magnitude of which is defined as follows

$$\mathbf{b} = - \left((x - y) \frac{\nabla_x \phi_h}{|\nabla \phi_h|} + (2x - y) \frac{\nabla_x \phi_h}{|\nabla \phi_h|} \right), \quad (5.26)$$

such that the circle undergoes a shear, a stretch and a rotation. An analytical solution for the position of the level set interface evolved in such a manner is derived in [158]. A set of points on the interface can be computed as follows, for $\vartheta = (0, 2\pi)$,

$$\mathbf{x} = \begin{bmatrix} A & B \\ -2B & C \end{bmatrix} \begin{Bmatrix} r \cos(\vartheta) \\ r \sin(\vartheta) \end{Bmatrix}. \quad (5.27)$$

where $A = \cos(-t_e) - \sin(-t_e)$, $B = \sin(-t_e)$ and $C = \cos(-t_e) + \sin(-t_e)$. The distance function to the interface (5.27) can then be computed by

$$\phi(\mathbf{x}, t_e) = \min(\sqrt{(x - Ar \cos(\vartheta_{\mathbf{x}}) - Br \sin(\vartheta_{\mathbf{x}}))^2 + (y + 2Br \cos(\vartheta_{\mathbf{x}}) - Cr \sin(\vartheta_{\mathbf{x}}))^2}), \quad (5.28)$$

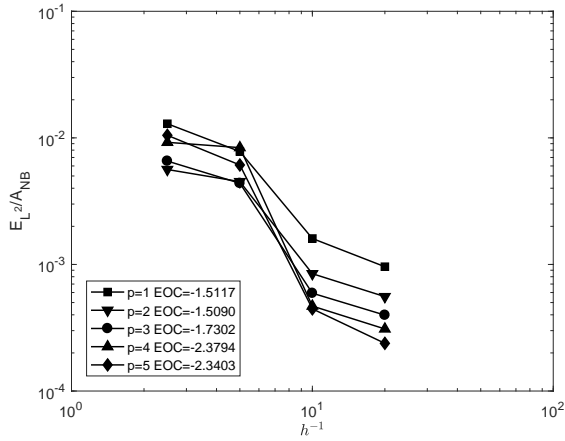
where $\vartheta_{\mathbf{x}}$ are the roots to the equation,

$$\begin{aligned} & (4BCr^2 - 2ABr^2) \sin^2(\vartheta_{\mathbf{x}}) + (2ABr^2 - 4BCr^2) \cos^2(\vartheta_{\mathbf{x}}) \\ & + (2C^2r^2 - 2A^2r^2 - 6B^2r^2) \sin(\vartheta_{\mathbf{x}}) \cos(\vartheta_{\mathbf{x}}) + (2Arx - 4Bry) \sin(\vartheta_{\mathbf{x}}) \\ & + (-2Brx - 2Cry) \cos(\vartheta_{\mathbf{x}}) = 0, \end{aligned} \quad (5.29)$$

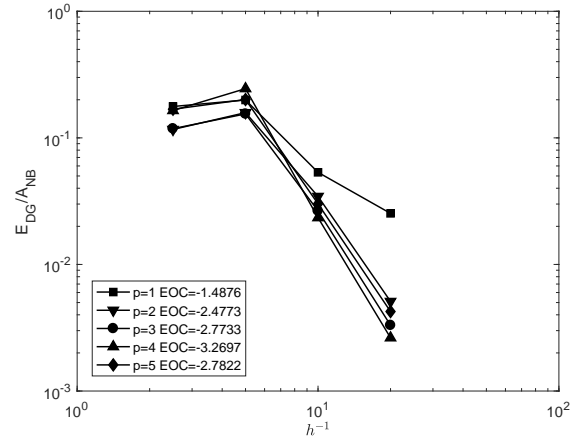
which for the purposes of this thesis are computed numerically using a bisection method, to a tolerance of 10^{-15} . Similarly the known value of the gradient is computed numerically using a linear finite difference method where the solution at a point is computed from (5.28) with (spatial) step size, $\Delta x = 10^{-12}$.

This problem will be computed on a series of uniform Cartesian meshes with square elements of size, $h = 0.2, 0.1, 0.05, 0.025$, and of uniform polynomial order, $p = 1, 2, 3, 4, 5$, and the error at $t_e = 0.3$ will be reported. For the purposes of accurately computing errors, at the time step where $t_e + \Delta t > 0.5$, the time step is reduced such that the level set function at this instance (i.e. $t_e = 0.5$ (to machine precision)) is captured.

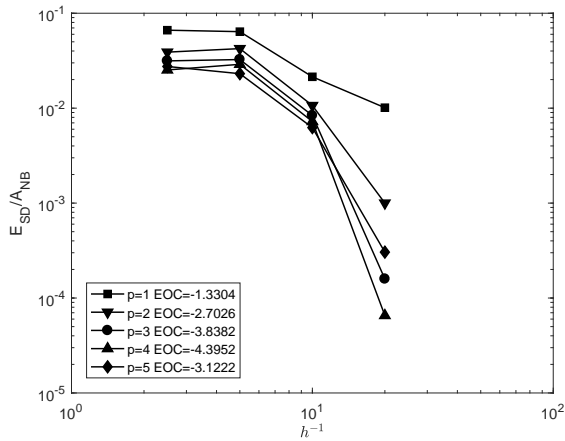
Figure 5.11 shows normalised errors for the shearing circle problem on all of the meshes at time, $t_e = T = 0.5$ and convergence rates for each of the polynomial orders, using the L^2 , DG and signed distance and interface error measures, defined in Section 4.4.1.1. For this example problem, at the moment the error is computed, $t_e = 0.5$, for the all of the meshes tested, a singularity is contained within the narrow band. As an example of this, Figure 5.12 shows the distribution of the error over the mesh where $p = 3$, and $h = 0.025$, where it can be seen



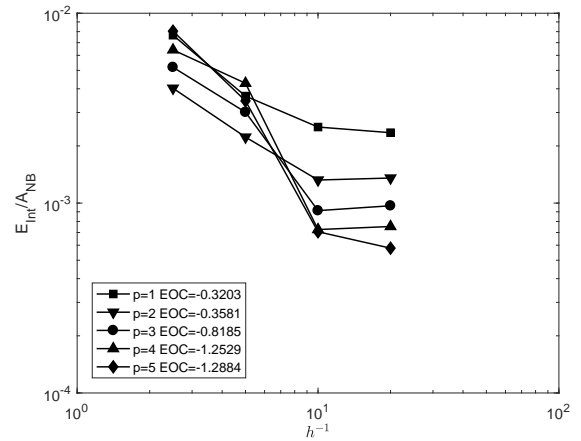
(a) Convergence in the L^2 norm.



(b) Convergence in the DG norm.



(c) Convergence using the signed distance error measure.



(d) Convergence using the interface error measure.

Figure 5.11: Error data normalised by narrow band area, A_{NB} , and associated convergence rates for shearing circular interface problem on narrow banded Cartesian meshes at time, $t_e = 0.5$.

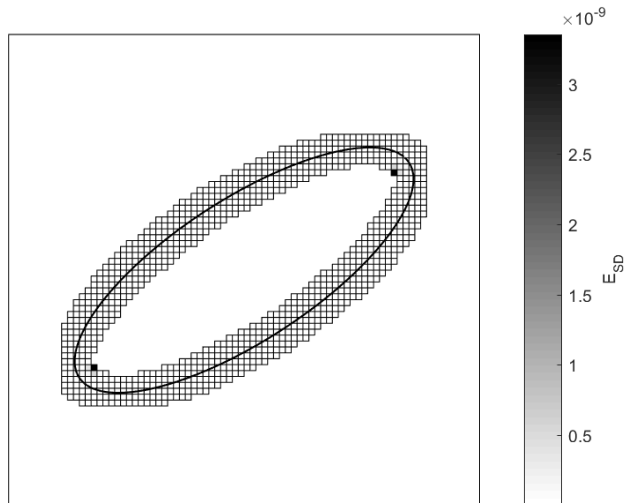
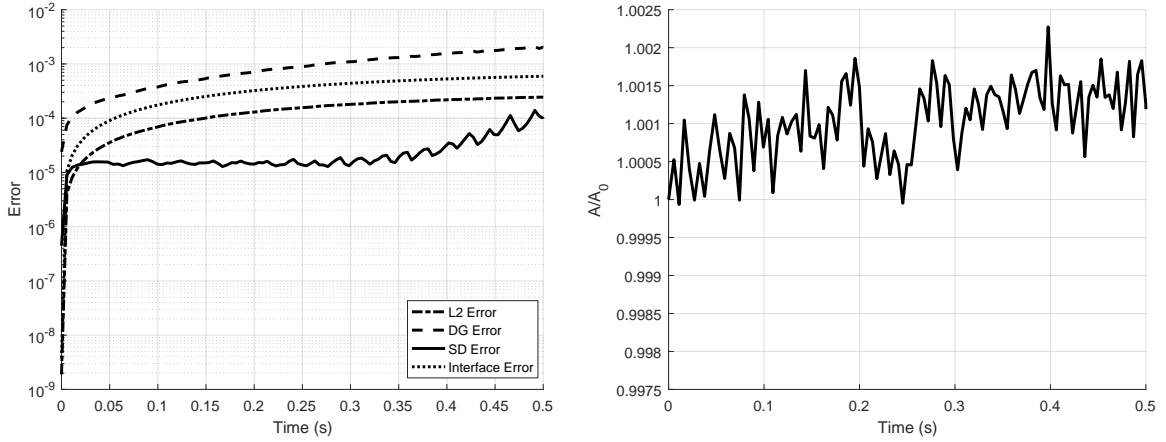


Figure 5.12: Distribution of the signed distance error over the narrow banded Cartesian mesh where $p = 3$ and $h = 0.025$, at time $t_e = 0.5$, for the shearing circle problem. The thick black line denotes implied position of the interface.

that almost all of the error is contained within two elements where the curvature of the level set function is greatest. For the coarser meshes more of the singular region will be included further affecting the accuracy of the solution. The experimental orders of convergence therefore should be expected to be equivalent to the linear case for all p in each of the error norms. This is exactly what is found in the experiment, with the error in the L^2 norm decreasing at a rate approximately proportional to h^2 for all p . This aligns with the convergence rate found in the first example problem. The error in the L^2 norm at the interface also aligns with that presented in the first example, decreasing at a rate approximately proportional to h^1 for all p . The error in the DG norm, does seem to show a significant improvement in the rate of convergence between meshes of linear and quadratic elements, however the rate is then limited for partitions of elements of higher polynomial order, to approximately h^3 . Likewise, the signed distance error appears to show improvement in the rates of convergence between meshes of linear and quadratic elements, and meshes of quadratic and cubic elements, before being limited to an order of approximation of around h^4 . It is difficult to explain the behaviour of both of the gradient based error measures. Also it should be noted that the quoted experimental orders of convergence have been computed between the meshes where $h = 0.1$ and $h = 0.025$.

The shearing circle problem ends up being much more difficult to solve than either of the preceding example problems, due to the larger singular region first and foremost, but also because the curvature of the level set function is no longer constant as it would be in the case of a circular interface. In the regions of increasing curvature the problem becomes more difficult to solve. In the previous examples this was evidenced by cases where the circular interface shrank. In this example however, as can be seen in Figure 5.13, almost immediately as the circle begins to shear, the ability of the mesh to resolve the shape diminishes by orders of magnitude in all error measures, and this occurs prior to the inclusion of any singularity the domain. For similar reasons, the solution to which the reinitialisation and extrapolation methods



(a) Error against pseudotime in the relevant norms.

(b) Area ratio against pseudotime.

Figure 5.13: Error and area ratio over time for the shearing circular interface problem, on the narrow banded Cartesian mesh where $h = 0.025$ and $p = 3$. The area ratio compares the area of the shape enclosed by the interface, A , with the area contained inside the initial interface, A_0 .

can converge worsens significantly over time. As the velocity field driving the advection for the shearing circle problem is divergence free, once again the area should remain constant throughout the evolution. Similar oscillations, as a result of the relative position of the interface to the grid, as in the previous examples can be seen, which is again reflected in the signed distance error. However, there is also a slight growth in the area contained within the shape, for this example of around 0.15%, which also coincides with the diminishing accuracy of the reinitialisation method as the curvature increases. Despite that which is stated above, there is still no visible difference between the computed and analytical solutions over the duration of the simulation, as can be evidenced by Figure 5.14.

5.4.2.4 h-convergence study: two merging circular interfaces

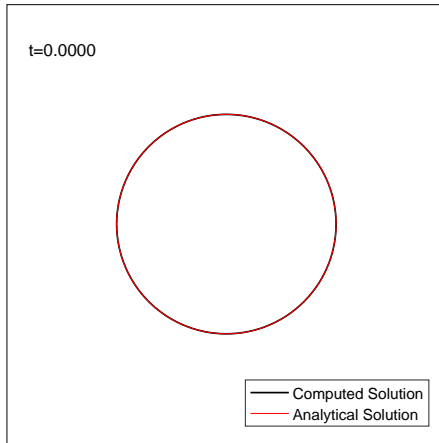
A level set function which is the signed distance function describing two circular interfaces with radii, $r = 0.5$, that is

$$\phi^0 = \min \left(\sqrt{(x - 0.55)^2 + y^2} - r, \sqrt{(x + 0.55)^2 + y^2} - r \right) \quad (5.30)$$

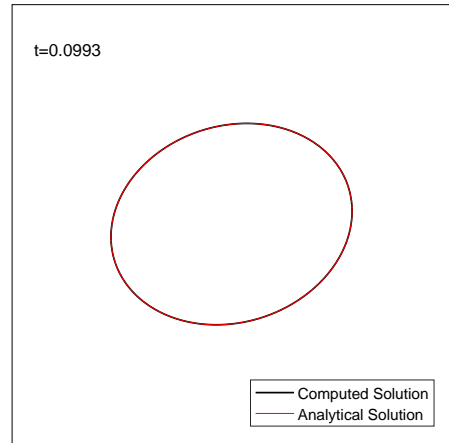
is L^2 projected onto the domain, $\Omega = (-2, 2)^2$. The level set function is evolved, from time $t_e = (0, 0.3)$, subject to a prescribed advection velocity vector the magnitude of which is defined as follows

$$b = \begin{cases} -1, & t_e < 0.15, \\ 1, & t_e \geq 0.15. \end{cases} \quad (5.31)$$

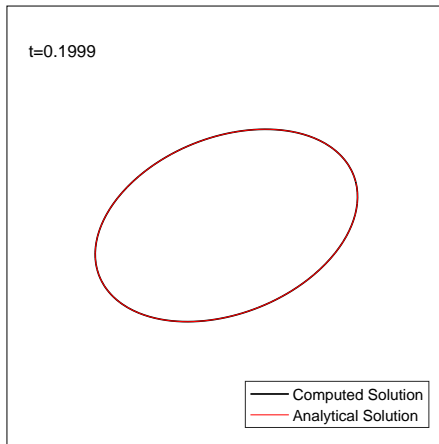
That is, two circular interfaces centred at $\mathbf{x} = (-0.55, 0)$ and $\mathbf{x} = (0.55, 0)$ which both grow at an equal constant rate. At $t_e = 0.05$ the two circles will meet at the origin, and then merge whilst continuing to grow at a constant rate. At $t_e = 0.15$, the shape of the merged circles begins to shrink, until $t_e = 0.3$. An analytical solution to this problem does not exist and as



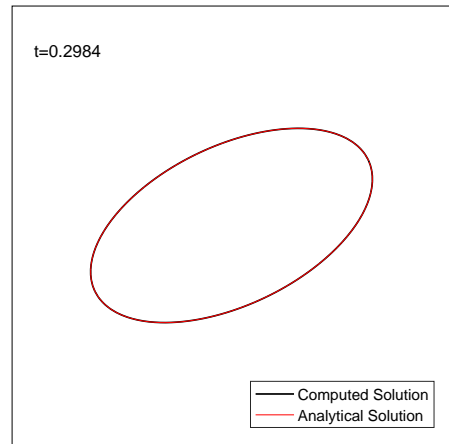
(a) Relative position of domain and interface at $t_e = 0.00$.



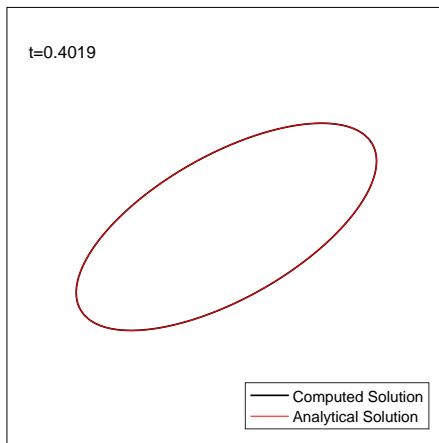
(b) Relative position of domain and interface at $t_e = 0.10$.



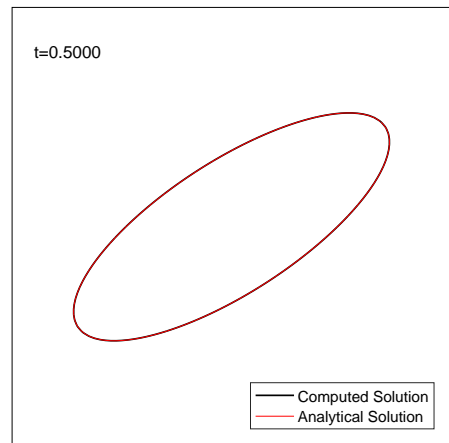
(c) Relative position of domain and interface at $t_e = 0.20$.



(d) Relative position of domain and interface at $t_e = 0.30$.



(e) Relative position of domain and interface at $t_e = 0.40$.



(f) Relative position of domain and interface at $t_e = 0.50$.

Figure 5.14: Computed and analytical interface position over time for the shearing circle problem, on the narrow banded Cartesian mesh where $h = 0.025$ and $p = 3$.

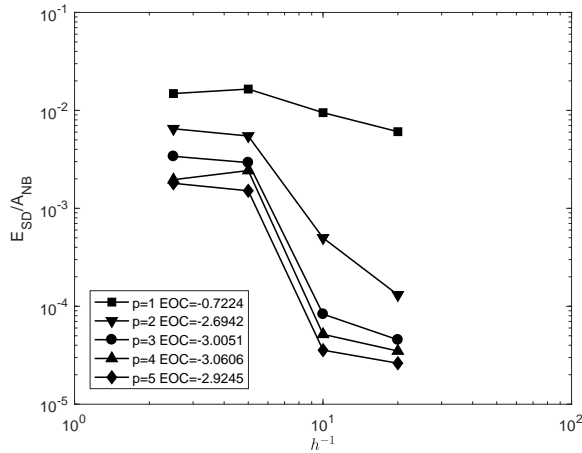


Figure 5.15: Error data normalised by narrow band area, A_{NB} , and associated convergence rates for the signed distance error measure, for the merging circles problem on narrow banded Cartesian meshes at time, $t_e = 0.5$.

such errors can only be reported in the signed distance error measure.

This problem will be computed on a series of uniform Cartesian meshes with square elements of size, $h = 0.3, 0.15, 0.075, 0.0375$, and of uniform polynomial order, $p = 1, 2, 3, 4, 5$, and the error at $t_e = 0.3$ will be reported. For the purposes of accurately computing errors, at the time step where $t_e + \Delta t > 0.3$, the time step is reduced such that the level set function at this instance (i.e. $t_e = 0.3$ (to machine precision)) is captured.

Figure 5.15 shows normalised errors for the merging circles problem on all of the meshes at time, $t_e = T = 0.3$ and convergence rates for each of the polynomial orders, using the signed distance error measure presented in Section 4.4.1.1. The quoted experimental orders of convergence have been computed between the meshes where $h = 0.15$ and $h = 0.0375$. Again it is known that at some point during the simulation a singularity will develop inside the narrow band as the two circles approach each other, which will be maintained throughout the remainder of the simulation. Therefore again it should be expected that the experimental orders of convergence presented will be limited by the lack of smoothness of the solution. As was seen in the previous example, in spite of the presence of a singularity there is a significant improvement in the rate of convergence between meshes of linear and quadratic elements, as well as a slight improvement in the rate of convergence between meshes of quadratic and cubic elements, at which point a limit is reached.

For the merging circles problem, the intersection and thus the singular region aligns exactly with the element edges for the first half of the simulation, this can be seen in Figure 5.17(a). As such the curvature of the level set function inside all of the elements in the narrow band decreases for the first half of the simulation, and the error over time in Figure 5.16 can be seen to appropriately decrease during this time. As soon as the merged circle begins to shrink the extrapolation inside the narrow band, introduces the singularity into the mesh as the average gradient forming the initial guess to the extrapolation takes gradients from either side of the singularity. This explains the spike in error seen between $t_e = 0.15$ and $t_e = 0.1554$ and is

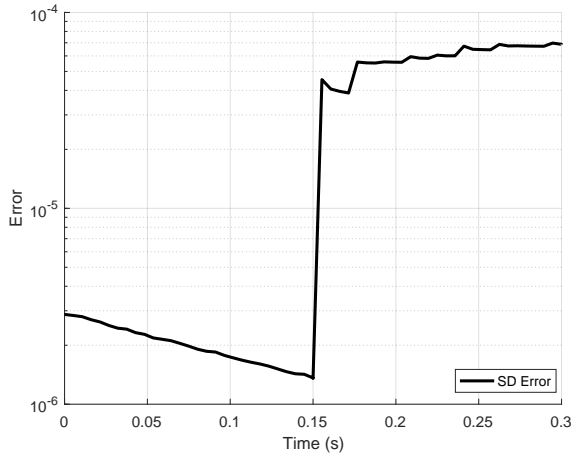
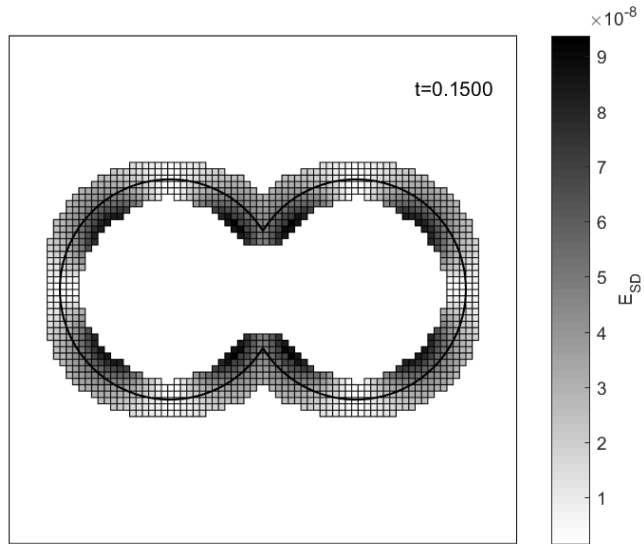


Figure 5.16: Signed distance error against pseudotime for the merging circles problem on the narrow banded Cartesian mesh where $p = 3$ and $h = 0.0375$.

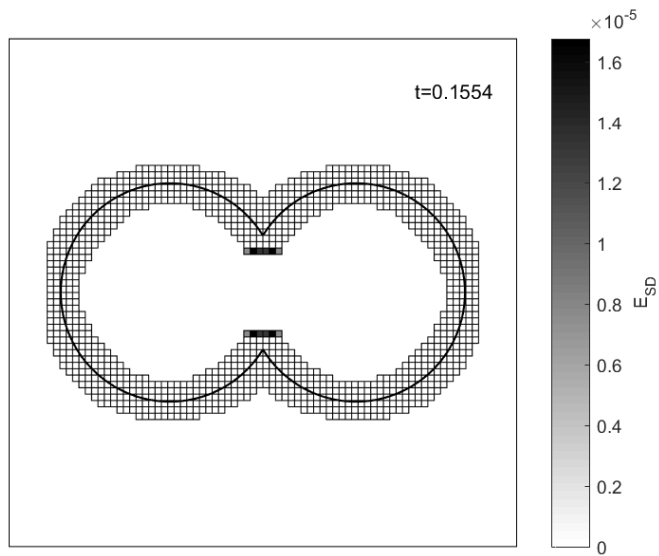
demonstrated in Figure 5.17(b) where almost all of the error is focussed on those elements near to the singular region upon which the level set function is extrapolated. Figure 5.18 shows the position of the computed interface over time. Beyond $t_e = 0.15$, a singularity is present inside these elements, and the sharp interface begins to dissipate over time which is reflected in the signed distance error and as can be seen in Figure 5.18.

5.5 Summary

With the inclusion of the Elliptic Reinitialisation method presented in Chapter 4, Chapter 5 proposes a narrow banded RKDG level set evolution methodology based on a simplified level set evolution equation discretised using a novel flux. The presented methodology also includes a novel extrapolation technique to extend the solution outside of the narrow band when necessary, and an Anderson acceleration algorithm which can be used to increase the convergence rate of the fixed point iterative method which is used when solving the reinitialisation and extrapolation problems. Through numerical experiments the proposed methodology is shown to be high-order accurate for sufficiently smooth problems. The example problems also demonstrate however, that the shape described by the level set interface does not have to become very complex before singular regions can move inside the narrow band and the desired high-order accuracy is surrendered. As the width of the narrow band is a function of the size of the elements comprising the narrow band, a potential solution to this is to adaptively refine the mesh near to these singular regions in order to narrow the narrow band where necessary to remove these singularities from the computational domain. If these singular regions are always too close to the interface, or in the worst case, where the interface itself is singular, likewise, adaptive mesh refinement will be necessary to minimise the error associated with capturing these regions. Furthermore, adaptive mesh refinement is likely to have other benefits including efficiency with regards to the number of degrees of freedom required to satisfy a given level of accuracy. For these reasons the next chapter proceeds by presenting research concerning adaptive mesh refinement for the proposed DG discretised level set methodology.

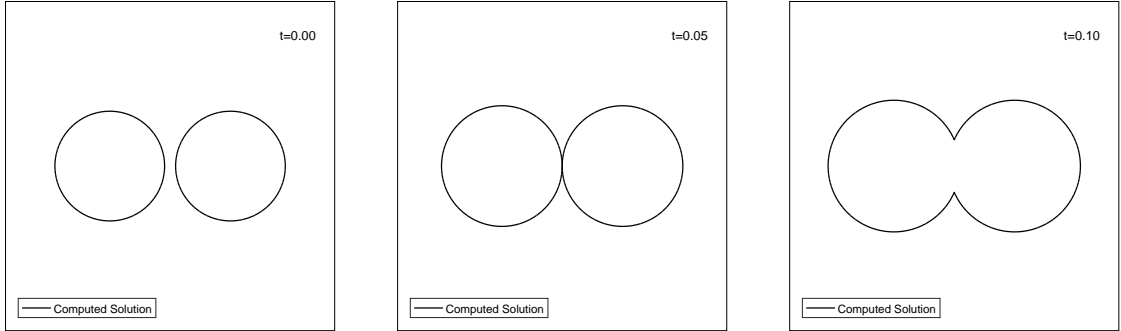


(a) Mesh configuration at $t=0.1500$

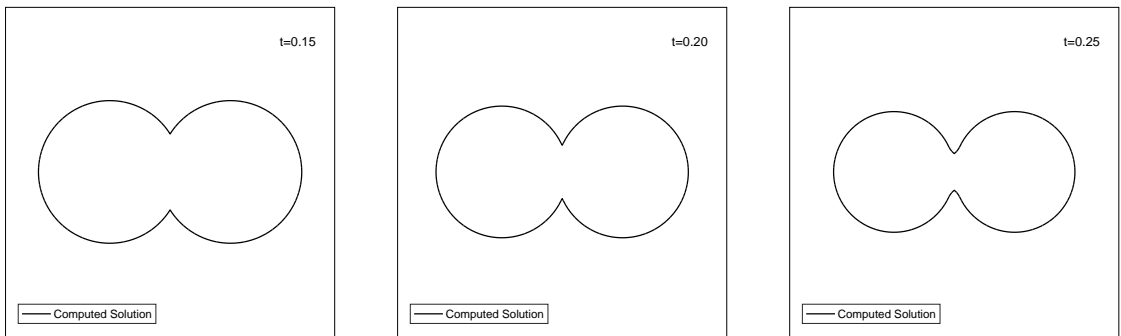


(b) Mesh configuration at $t=0.1554$

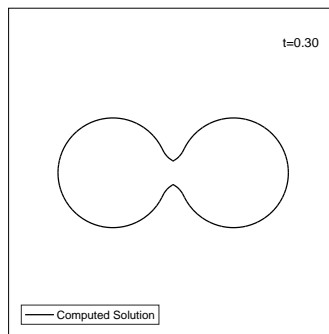
Figure 5.17: Distribution of the signed distance error over the narrow band Cartesian mesh where $h = 0.0375$ and $p = 3$, for two consecutive time steps of the merging circles problem, across which the solution becomes singular. The thick black line denotes the implied position of the interface.



(a) Relative position of domain and interface at $t_e = 0.00$ (b) Relative position of domain and interface at $t_e = 0.05$. (c) Relative position of domain and interface at $t_e = 0.10$.



(d) Relative position of domain and interface at $t_e = 0.15$. (e) Relative position of domain and interface at $t_e = 0.20$. (f) Relative position of domain and interface at $t_e = 0.25$.



(g) Relative position of domain and interface at $t_e = 0.30$.

Figure 5.18: Computed interface position over time for the merging circle problem, on the narrow banded Cartesian mesh where $h = 0.0375$ and $p = 3$.

Chapter 6

Level Set Method: Adaptive Mesh Refinement

With the development of a high-order accurate level set method based on a DG discretisation, the next step is to exploit one of the main advantages afforded by the use of discontinuous finite element spaces; that is the ease with which the method can incorporate hp -adaptivity. This is advantageous as it allows one to concentrate computational effort in areas where more computational effort is required and vice versa, and therefore to make efficient use of the number of degrees of freedom in the system. In this respect there are two key components to be developed. First of all, appropriate criteria for driving the mesh refinement, that is criteria which can be used to decide where refinement and derefinement are necessary and thus promote an efficient distribution of the degrees of freedom within the domain. And secondly an algorithm which can distribute these degrees of freedom based on the chosen criteria. Given these aims this chapter proceeds in the following manner: in Section 6.1, the literature concerning adaptive mesh refinement in the context of level set methods will be reviewed, in Section 6.2 a criterion for mesh refinement will be proposed, in Section 6.3 an algorithm will be proposed and explained which adaptively refines the mesh based on the proposed refinement criterion, in Section 6.4 the proposed refinement criterion/algorithm will be evaluated by computing a number of level set reinitialisation examples problems on hp -adaptive meshes, in Section 6.5 the full hp -adaptive level set methodology will be presented, and finally Section 6.6 concludes the chapter with a number of numerical examples of the full level set evolution on hp -adaptive meshes.

6.1 Literature review: adaptive mesh refinement for level set methods

When it comes to interface problems, it is generally considered that both the area of interest, and the area which would require the highest resolution, is the area surrounding and including the interface itself. This idea has motivated the almost ubiquitous criterion driving mesh adaptivity in the context of level set methods which was stated succinctly in [26] as follows, “split any cell whose edge length exceeds its minimum distance to [the interface]”. In other words, starting from a fixed grid h -refine any element defined by some measure as being near to the level set

interface. A list of references which approach mesh adaptivity in this fashion includes, but is not limited to, [26–32, 118, 126]. A similar but slightly less popular approach is to p -refine elements which are near to the interface; this type of approach can be found in works such as [24, 25]. In general, a refinement criterion such as this has a number of benefits. First of all, only geometric information is required which is easy to compute with level set methods, and therefore the expense associated with error estimation is removed from each refinement step. Similarly, such an approach is likely to focus computation in the area where typically the level set function itself and any underlying physical quantity would require a high resolution. Such an approach is unlikely, however, to be optimal in terms of the number and distribution of the degrees of freedom. For example, one could consider an interface (or a portion of an interface) which is a straight line, aligned with the grid; in such a case it is conceivable that few, low order elements could accurately capture the information relating to the interface in such a region, and high levels of either h or p adaptivity in this region would be wasteful.

Some researchers have attempted to use other criteria for deciding where a mesh should be refined when solving interface problems using the level set method. For example, a refinement criterion is presented by Wei et al. in [159], where an estimate of the curvature of the level set function, on elements intersected by the interface, is used to choose where and how to h -refine the mesh. In their article Wei et al. use this approach to generate a body-fitted mesh and therefore would be of limited utility in the work to be presented here. This idea does have some merit, however. First of all, the curvature of the level set function is again a geometric quantity and therefore simple and cheap to compute, and secondly it is the areas of the problem where the curvature is highest which require the highest resolution to be accurately captured and vice versa. One drawback of such an approach however is that it doesn't provide an estimate of the quality of the mesh, which a standard error estimator would, and as such the mesh will simply have upper and lower limits on the size and/or polynomial order of the elements used and refinement will occur until these limits are satisfied, which could be suboptimal. A number of other methodologies have been presented which simply ignore the level set method when it comes to mesh refinement, and use error estimators for the underlying physical problem to drive mesh refinement, examples of which include [160, 161]. This is an approach which is useful as by using the same mesh for both the physical and the level set problems, the advection velocity will de facto be well resolved where it needs to be using such an approach. However, this does of course mean limited generality, and is of little use for problems not driven by a underlying physical problem. Another methodology is presented in [162], where an error estimate, based on a multi-resolution analysis [163], is used to drive h -refinement.

Almost none of the literature takes advantage of the combination of h and p refinement. It is known that for regions of a problem where the solution is sufficiently smooth, that it is more efficient to increase the polynomial order, p , and in regions where the solution is singular it is more efficient to split a cell, [164]. It is likely therefore that a more efficient refinement strategy would utilise both h and p adaptivity. One example of a DG level set method which does take advantage of an hp -adaptive mesh, is presented for one of the example problems in [33], however, almost all of the details concerning the adaptivity have been obfuscated.

6.2 Refinement criterion

To construct an appropriate criterion for driving mesh refinement, whether this be an error estimate or some other *ad hoc* approach, the first port of call is to examine the equation over which one is trying to control the error by adaptively refining the mesh. In this case, that is the simplified evolution equation, (5.1), which consists of just two quantities, the time derivative of the level set function, $\frac{\partial \phi}{\partial t_e}$, and the magnitude of the advection velocity vector, b . Of these two, only the advection velocity contains information pertaining to the spatial discretisation of the mesh. In looking to construct an error estimate by which one can determine the quality of the mesh, in terms of how accurately a given mesh could be expected to compute a given iteration of the evolution equation, it would therefore be appropriate to determine the error in the computation of the advection velocity vector. As mentioned earlier it is the intention of the author to apply this level set methodology to the topological optimisation of linear elastic structures, and as such it may be appropriate eventually to use an error estimate developed for linear elasticity such as [165], to drive the mesh refinement, as for this given application such a strategy would likely mean that the advection velocity field would be appropriately resolved. In the general case however, it is not necessarily the case that a refinement criterion based on the underlying physical problem would generate a mesh which is optimal for solving the level set equations and of course any such refinement strategy would be problem specific. Thus in this section the focus will be the development of a refinement criterion for the level set methodology which is agnostic of application, and which could then if need be, be combined with an appropriate refinement strategy for the physical problem.

The spatial term in this case being equal to the magnitude of the advection velocity vector results from the fact that level set reinitialisation ensures that the Euclidean norm of the gradient is always equal to unity. It seems to be appropriate in constructing an error estimator then, that one should construct the mesh such that as far as possible the norm of the gradient does in fact approximate unity. For this purpose an error measure has already been defined earlier; the signed distance error measure,

$$E_{\text{SD}}^2(\phi_h, \mathcal{T}) = \sum_{\tau \in \mathcal{T}} \int_{\tau} (|\nabla \phi_h| - 1)^2 \, d\mathbf{x}, \quad (4.56)$$

which is simply the L^2 norm of the difference between the Euclidean norm of the gradient and unity, over the domain. If one were to simply adopt this as the refinement criterion, one immediate issue would be that, if the level set function is not currently a good approximation of a signed distance function, as defined by the signed distance error, that mesh refinement alone will not improve the situation. However, by computing a solution reinitialisation problem until convergence, and then refining the mesh where the reinitialisation has performed poorly, the reinitialisation routine can simply be called again, and as was demonstrated in Section 4.4.1, will converge to a solution with a smaller signed distance error. Therefore reinitialisation and mesh refinement can be combined in a loop to ensure that an appropriate mesh exists to allow a signed distance function to be computed to the current position of the level set interface, with

the quality of this mesh/signed distance function defined by the signed distance error measure.

In this way then, the question of mesh refinement for the evolution problem, has been recast as the question of how to appropriately refine a mesh for the level set reinitialisation problem. Level set reinitialisation is a process which ensures that the unit outward normal of the level set interface, $\hat{\mathbf{n}}_\Gamma$, is equal to the unit outward normal of the level set function, $\hat{\mathbf{n}}_\phi$, all along the interface normal. In other words reinitialisation ensures that the level set function in the direction of transport is well defined and of uniform magnitude along the level set interface. As the level set interface can only evolve along its own unit outward normal at that point, it is therefore likely appropriate to use a measure of signed distance-ness to determine the quality of the mesh for the level set evolution problem. The efficacy of this refinement criterion, when combined with the refinement strategy to be proposed in Section 6.3 is demonstrated through experiment in Section 6.4.

6.3 Refinement strategy

A strategy for mesh refinement is proposed algorithmically in Algorithm 3. The aim of the refinement strategy presented here, is take a level set function, ϕ , on a subset, \mathcal{S} , of the mesh, \mathcal{T} , and perform a series of mesh refinement steps combined with a method for updating the value of the level set function such that after each step the level set function is a more strict approximation of a signed distance function. This refine then update strategy will continue until a combination of ϕ and \mathcal{T}_{NB} are computed which either satisfies a tolerance, `errorTol`, on some error measure, pointed to by the variable `errorHandle`; or until upper limits on the allowed amount of refinement are met, `upperhRefLim` and `upperpRefLim`. The `errorHandle` variable will in all cases point towards either the L^2 error in the solution as defined in Equation (4.53), or the signed distance error estimate defined in Equation (4.56). The algorithm operates on a set of elements, \mathcal{S} , which may be any subset of \mathcal{T} , for example in the case that the algorithm should be executed over the narrow band then, i.e. $\mathcal{S} = \mathcal{T}_{\text{NB}}$.

There are four possible cases which Algorithm 3 can be presented with when updating the level set function, defined by the user inputted `problemHandle` variable. These cases are; `problemHandle = 'initialise'`, whereby after each refinement step an analytical function describing a signed distance function is projected onto the new mesh; `problemHandle = 'initialiseNonSDF'`, that is the initialisation of a level set function which is not a signed distance function, whereby a function will be projected onto the new mesh at each refinement step, which will be reinitialised prior to the next mesh refinement to ensure that the returned function is an approximate signed distance function; `problemHandle = 'reinitialise'`, whereby the solution to the previous reinitialisation will be projected onto the new mesh at each refinement step, which can then be reinitialised on the new mesh to a greater degree of accuracy; and `problemHandle = 'extrapolate'`, whereby the solution to the previous extrapolation problem (5.16) is projected on a subset of elements on the new mesh, before the extrapolation is recomputed to a greater degree of accuracy. More detail about each of the various steps of Algorithm 3 will be discussed in the proceeding subsections.

```

function [ $\phi, \mathcal{T}_{NB}$ ] = refineUpdate ( $\phi^0, \mathcal{S}, \mathcal{T}_{NB}, \text{errorTol}, \text{refineThresholds},$ 
upperhRefLim, upperpRefLim, errorHandle, analyticityThreshold,
problemHandle, maxIt)
[errorGlo,errorLoc] = computeError( $\phi^0$ , errorHandle,  $\mathcal{S}$ );
area = computeDomainArea( $\mathcal{S}$ );
while errorGlo/area > errorTol do
    // GET REFINEMENT FLAGS
    flags = computeRefineFlags(errorLoc, refineThresholds,  $\mathcal{S}$ );
    [pFlags,hFlags] = computeAnalyticityTest(flags,  $\mathcal{S},$ 
analyticityThreshold);
    [pFlags,hFlags] = enforceRefinementLimits(pFlags, hFlags,
upperhRefLim, upperpRefLim,  $\mathcal{S}$ );
    [pFlags,hFlags] = refineSmoothing(pFlags, hFlags,  $\mathcal{S}$ );
    if all(pFlags==0) AND all(hFlags==0) then
        break;
    end

    // REFINE MESH AND PROJECT
     $\mathcal{S}$  = pRefine(pFlags,  $\mathcal{S}$ );
     $\phi$  = L2Project( $\phi$ , pFlags,  $\mathcal{S}, \text{problemHandle}$ );
     $\mathcal{S}$  = hRefine(hFlags,  $\mathcal{S}$ );
     $\phi$  = L2Project( $\phi$ , hFlags,  $\mathcal{S}, \text{problemHandle}$ );

    // UPDATE NARROW BAND
     $\mathcal{T}_{NB1}$  = updateNB( $\phi, \mathcal{T}_{NB}$ );
    if  $\mathcal{T}_{NB1} \setminus \mathcal{T}_{NB} \neq \emptyset$  then
         $\mathcal{T}_T = \mathcal{T}_{NB1} \setminus \mathcal{T}_{NB}$ ;
        extrapolate( $\mathcal{T}_T$ );
        // Refine only on ( $\mathcal{T}_T$ ).
        [ $\phi, \mathcal{T}_{NB}$ ] = refineUpdate( $\phi, \mathcal{T}_T, \mathcal{T}_{NB1}, \text{errorTol}, \text{refineThresholds},$ 
upperhRefLim, upperpRefLim, errorHandle, analyticityThreshold,
'extrapolate', maxIt);
    else
         $\mathcal{T}_{NB} = \mathcal{T}_{NB1}$ ;
    end

     $\mathcal{S}$  = updateCurrentSet( $\mathcal{S}$ ) ;
    if  $\mathcal{S} = \emptyset$  then
        break;
    end

    // UPDATE LEVEL SET FUNCTION ON SET,  $\mathcal{S}$ 
    switch problemHandle do
        case 'initialise'
            do nothing;
        case 'reinitialise' OR 'initialiseNonSDF'
             $\phi$  = reinitialise( $\phi, \mathcal{S}, \text{maxIt}$ );
        case 'extrapolate'
             $\phi$  = extrapolate( $\phi, \mathcal{S}, \text{maxIt}$ );
    end

    [errorGlo,errorLoc] = computeError(errorHandle,  $\mathcal{S}$ );
    area = computeDomainArea( $\mathcal{S}$ );
end

```

Algorithm 3: Mesh refinement algorithm

6.3.1 Initial flagging strategy

After computing the error, where the error measure is defined by the user inputted `errorHandle`, should the global error (the error over the chosen subset of the domain, \mathcal{S}), `errorGlo`, divided by the area of \mathcal{S} , `area`, be greater than the user defined error tolerance, `errorTol`, then the algorithm will attempt to refine the subset of the mesh, \mathcal{S} , based on the error computed locally on each element, `errorLoc`. The `computeRefineFlags` function does this by first determining the maximum elementwise error, and then any element with a local error greater than some user defined percentage of this error is flagged for refinement, `flags[τ]` = +1. Similarly, any element with an error less than some percentage of this maximum error is flagged for derefinement, `flags[τ]` = -1. Any element inside the narrow band with an error between these two values is given a *do not refine* flag, i.e. these elements are actively not flagged for refinement or derefinement, `flags[τ]` = 0. These percentages of the maximum error are input to the algorithm using the `refineThresholds` variable, in all cases here the *refine if greater than* value is chosen to be 0.75 of this maximum error, and the *derefine if less than* value is chosen to be 0.08 of the maximum error, that is `refineThresholds` = [0.08, 0.75].

As no computation occurs outside of the narrow band, in order to ensure the minimum amount of resources are used, the following addendum to the flagging strategy is employed: the set of elements which are outside of the narrow band but share at least one node with any element inside the narrow band, $\tau \in \mathcal{T}_{\text{ONB}}$, are always given a do not refine flag, and all other elements outside of the narrow band but currently active in the domain are flagged for derefinement. The reason for this band of no refinement for the layer of elements just outside the narrow band, is to ensure that no infinite refine/derefine loops occur by which an element just outside the narrow band can get stuck between being refined out of the narrow band and then derefined back inside the narrow band as a result of the rules outlined so far with regards to flagging.

6.3.2 hp-steering criterion

Once this initial set of flags has been determined the next step of the algorithm is to decide whether an element flagged for either refinement or derefinement should be refined in h or p . As mentioned previously, it is known that if the solution on an element is smooth it is more efficient, in terms of number of degrees of freedom, to refine in p than in h , and vice versa [164]. For this reason, there have been a number of attempts in the literature to develop algorithms by which one can estimate the local smoothness of the solution over a domain, so that one can identify the regions of the domain which are smooth enough for it to be most efficient to refine in p and likewise the regions where the solution is non-smooth and it is most efficient to refine in h . It should be noted of course, that not all hp -adaptive strategies presented in the literature use a smoothness estimate to determine whether an h or p refinement strategy should be employed. For example, the three fold strategy presented in [166], simply uses the magnitude of the error estimator driving the refinement as an hp -steering criterion i.e. the elements with the largest errors should be refined in h , the elements with an intermediate level of error in p and no refinement should occur on elements with small levels of error. However, as the aim here

is to maximise efficiency, the focus will be on methods based on smoothness estimates.

The simplest strategy in this regard is to use *a priori* information concerning the location of singularities, which has found use for example in [167]. This unfortunately is not appropriate for the purposes intended here as computing optimal topologies necessarily implies a lack of knowledge *a priori* about the converged solution. A second strategy proposed by Gui and Babuška [168] evaluates the smoothness of a solution on an element by computing the ratio of two local error estimates for numerical solutions of order p_τ and $p_\tau - 1$. By comparing this ratio with what is referred to as the type-parameter, one can then determine whether the refinement should be of p -type indicating sufficient smoothness or vice versa. Houston *et al.* present two methods of smoothness estimation in their paper, [164]. The first of these is a technique which involves computing the decay rate of the coefficients of a Legendre expansion of the solution. It is known that the value of these Legendre coefficients will decay to zero exponentially if the solution is analytic (i.e. smooth), thus a sufficiently fast decay rate indicates a preference for refinement in p over h , and vice versa. The second method presented in [164] is a technique by which one directly approximates the local Sobolev regularity of the solution using the root test on those same Legendre expansion coefficients. More recently Wihler [169], developed a smoothness indicator based on continuous Sobolev embeddings. In [169], it is shown that the continuity coefficients in the relevant Sobolev inequalities will be seen to tend towards a positive constant as the mesh is refined if the solution is smooth on a given element, and tend towards zero otherwise, which can be used to indicate that one should refine in p or h respectively. A comparison of a number of hp -adaptive strategies including the majority of those described above is performed in [170]. What is found is that whilst different strategies perform best (evaluated in terms of efficiency) for different kinds of problems, the method of Houston involving the computation of the decay rate of the coefficients of a Legendre expansion appears to be the best choice as a general strategy across the range of tested problems.

In this work then, the analyticity test presented in [164] which involves on the estimation of the decay rate of the Legendre expansion coefficients will be the chosen method for estimating the smoothness of a function, and will thus form the first step in the decision of whether an element should be refined in p or h . This will comprise the `computeAnalyticityTest` function in Algorithm 3, and can be computed as explained below. Firstly, the Legendre expansion of a function, $\phi_h(x)$, can be written

$$\phi_h(x) = \sum_{i=0}^{\infty} a_i \ell_i, \quad (6.1)$$

where a_i are the Legendre coefficients and ℓ_i the Legendre polynomials. As this is a computational technique, however, an approximation of (6.1) must be made which can be done by computing a finite Legendre expansion, the maximum order of which will align with the interpolation order of the finite element space on the given element i.e. for $i = (0, p_\tau)$. In order to estimate the decay rate of the finite Legendre expansion one can then fit a straight line through a graph of the log of the coefficients, a_i , and i , i.e. $\log |a_i| = mi + c$. The analyticity, Θ_τ , then is a function of the gradient of that line, $\Theta_\tau = \exp(-m)$. If the function is locally analytic the decay rate of the coefficients is exponential and the gradient, m , will be large and thus $\Theta_\tau \rightarrow 0$,

whereas if the function is non-analytic the gradient, m , will be small and $\Theta_\tau \rightarrow 1$. A parameter, `analyticityThreshold`, can then be introduced by which if, $\Theta_\tau \leq \text{analyticityThreshold}$, the solution is considered smooth enough locally for the element to be refined in p , and for, $\Theta_\tau > \text{analyticityThreshold}$, the element should be refined in h . As smooth functions will tend to have values close to zero and nonsmooth functions a wider range of values that nevertheless tend towards one, caution is exercised in favour of non-smooth values (and thus h -refinement) and as such, in this work the value of the parameter is set as `analyticityThreshold`= 0.25. Experimental evidence can be found in works such as [172, 173], which show that such a choice leads to the desired exponential convergence and is therefore a reasonable choice.

There are some special cases such as where a function is odd or even by which the symmetry or antisymmetry of the function will cause patterns of repeating zeros in the Legendre coefficients which can affect the data fitting procedure through the coefficients. To avoid this, a check is made by which if the even numbered coefficients i.e. $i = 2, 4, 6\dots$ are all less than machine precision, the function can be determined to be odd and if the odd numbered coefficients i.e. $i = 1, 3, 5\dots$ are all less than machine precision, the function can be determined to be even. If either one of these conditions is satisfied a smoothing is applied to the Legendre coefficients by which the repeated zero coefficients are set as equal to the average of their adjacent coefficients, or in the extreme cases (i.e. the 0th or p th coefficient) set as equal to their adjacent coefficient (i.e. the 1st or $p - 1$ th coefficients respectively). Similarly, in the case that the order of the function is less than the local order of approximation on the element, there could be repeated zero coefficients at the end of the series by which the coefficients have already decayed to zero. As such, a condition is enforced by which if the highest order Legendre coefficient is less than machine precision, and the function is not odd or even (based on the criteria defined above), the function is assumed to be analytic.

Furthermore, it should be noted that a Legendre expansion is computed about the centre of the element and in one coordinate direction only. This does mean that one can use this technique to determine separately the analyticity of the function in each coordinate direction and use this information to refine the mesh anisotropically, however, in this case the analyticity is simply estimated in both coordinate directions on a given element and the worst case scenario is used, i.e. the function must be analytic in both coordinate directions to be considered analytic and thus the decision be made to refine in p over h .

6.3.3 Enforcement of refinement limits

The aim of this mesh refinement strategy is to focus degrees of freedom in the most problematic areas in terms of some error to most efficiently resolve the level set function until a tolerance on that error is met. In practice, it is often also necessary to set an upper limit on the possible polynomial order an element can be refined to, `upperpRefLim`, and a lower limit on element size, `upperhRefLim`. Whereas the tolerance, `errorTol`, sets the desired level of accuracy with which the mesh is attempting to resolve the level set function, the limits on refinement determine the minimum level of resolution at which the computation will continue if it is unable to reach this desired level of accuracy. The main reason for this is that there will ultimately be limits on memory, which is likely to be the main bottleneck in such computations. However, there may

be other practical real world reasons for these limits, such as tolerances on instrumentation or design tolerances by which smaller elements would be wasteful.

As such, at this stage of the refinement algorithm, the current polynomial order on an element is summed to the p -refinement flag on that element, and if this number is greater than the maximum allowed polynomial order, p_{\max} , then the p -refinement flag is set to do not refine. There is of course also a natural lower limit (or it is possible to choose one, $p_{\min} = 2$ is often used as a lower limit in this work as it can be difficult to accurately compute the analyticity on linear elements) on the minimum allowed polynomial order of the space on the element which is also enforced here, should the sum be less than that number. At this stage, if it is found that an element is flagged for p -refinement or derefinement but is unable to refine/derefine as a result of these limits, the flag is switched to the equivalent h -refinement flag. The same condition is then checked for the h -refinement flag by which the current number of levels of h -refinement is summed to the h -refinement flag and compared with a maximum number of levels of allowed h -refinement (which is equivalent to specifying a minimum element size), and once again a natural minimum (that being 0 levels of h -refinement which is equivalent to the original Cartesian mesh on which the level set function is originally projected). The same procedure also occurs for elements initially flagged for h -refinement and derefinement, by which the upper and lower limits are enforced and in the case that a lower or upper limit is reached, the h -flag is set to do not refine, and if possible the flag can be switched to a p -refinement flag should that not also breach the lower or upper limit on p -refinement.

6.3.4 Flag smoothing

As stated in Section 2.2, in this work a bound on the local variation in h and p is enforced which requires that at most any given element will have one hanging node per edge, and adjacent elements will have a difference of at most one polynomial degree in their bases, and thus will satisfy Equation (2.3). In order to ensure that this is always the case a smoothing algorithm is applied to the h and p refinement flags at this stage which ensures that post-reinitialisation these bounds will be satisfied. In a situation where this would not be satisfied, the flags will be altered following the rule that refinement is always preferable to no refinement, and no refinement is preferable to derefinement.

For the p -flagging strategy, a loop through all of the elements currently in the mesh is performed, whereby for each element, τ^+ , and all of the elements with which the τ^+ shares a face, the current polynomial order of each element, p , is summed to its p -refinement flag, that is, $\rho_{\tau} = p + \mathbf{pFlags}[\tau]$. The value ρ_{τ^-} for each of the neighbouring elements is then subtracted from the ρ_{τ^+} . If the absolute value of the difference $|\rho_{\tau^+} - \rho_{\tau^-}| < 2$, then this means that post-reinitialisation the local bound will not be satisfied. As per the preference for refine over do not refine, and do not refine over derefine, the element with the smaller $|\rho_{\tau}|$, will have its p -flag increased by one. As each smoothing can necessitate further smoothing, this procedure is performed iteratively until no more smoothing is required. The same loop is then computed for h -refinement flags, with an additional step, that being, for an h -derefinement to occur, in the quadtree paradigm, all four of the children elements are required to have a derefinement flag. In the case that any of the four children have an h -derefinement flag, and any of the other three

children have an h -flag that is not a derefinement flag, then all of the children with derefinement flags are replaced with do not refine flags.

6.3.5 Mesh refinement

Once the flags have been computed on the current mesh, should, as a result of the limits on refinement, and or, the smoothing algorithm, all of the flags on the mesh be set to do not refine, then the loop is broken as the best possible resolution of the level set function given the limits on refinement, has already been achieved. Otherwise the next step is to refine the mesh. Given the Eulerian nature of the level set method, one can begin with a Cartesian mesh, and then use a simple and efficient quadtree structure for h -refinement. An element flagged for h -refinement can be split into 4 equally sized square elements, and similarly if all 4 children elements, are flagged for h -derefinement then they can be replaced by their original parent element. Then, as modal hierarchical basis functions are used on each element, p -refinement/derefinement is also simple and amounts to including/removing (respectively) the higher order degrees of freedom to the given element.

6.3.6 Project the solution onto the new mesh

For any element flagged for h -refinement/derefinement, the level set function is determined on the new element, $\phi_h^{\text{new}}(\mathbf{x}, t_e)$, by interpolating on the old mesh (which for derefinement would mean interpolating on all four children elements) the value of the old level set function at the Gauss point positions on the new mesh, $\phi_h^{\text{old}}(\mathbf{x}, t_e)$, and computing an L^2 projection, that is

$$\left(\phi_h^{\text{new}}(\mathbf{x}, t_e) - \phi_h^{\text{old}}(\mathbf{x}, t_e), v_h \right)_\tau = 0, \quad \forall v_h \in V_p. \quad (6.2)$$

Executing a p -refinement on a given element, τ , simply amounts to initialising degrees of freedom associated with the new high-order polynomials, and setting the value of the level set function at these degrees of freedom to zero. Whereas p -derefinement, simply requires the removal of the degrees of freedom associated with the old higher-order polynomial bases.

If the refinement algorithm is being called as part of the initialisation routine, defined by, `problemHandle = 'initialise'` or `problemHandle = 'initialiseNonSDF'`, then at this stage $\phi_h^{\text{old}}(\mathbf{x}, t_e) = \phi^0$, that is the analytical function defining the initial condition will simply be projected onto the refined mesh at each refinement step.

6.3.7 Update the narrow band

Once the level set function has been projected onto the new mesh, the set of elements comprising the narrow band, \mathcal{T}_{NB} , needs to be updated. The same procedure as explained in Section 5.2 for determining the set of elements currently in the narrow band is used here, initially the new narrow band set will be labelled \mathcal{T}_{NB1} . If the set of elements comprising the narrow band changes such that elements which were not in the old narrow band set \mathcal{T}_{NB} , are in the new narrow band set \mathcal{T}_{NB1} , which might happen after a derefinement, then the level set function on these elements can be generated using the extrapolation technique presented in Section 5.2.1. It is then possible, if need be, to recursively call the `refineUpdate` function, on the set $\mathcal{T}_T = \mathcal{T}_{\text{NB1}} \setminus \mathcal{T}_{\text{NB}}$, to ensure that this extrapolation also satisfies the desired tolerance on the error, `errorTol`. On the

recursive calls of the `refineUpdate` function, it is the new narrow band, \mathcal{T}_{NB1} , which is passed in and therefore is updated as such. The two other possibilities are that the set comprising the new narrow band, \mathcal{T}_{NB1} , is equal to the previous narrow band, \mathcal{T}_{NB} , or contains only elements which also belong to the previous narrow band, \mathcal{T}_{NB} . In both of these cases, no further action needs to be taken and the narrow band can be updated $\mathcal{T}_{\text{NB}} = \mathcal{T}_{\text{NB1}}$.

The set of elements being operated on, \mathcal{S} , then needs to be updated based on the rules by which it exists. The most common case will be that the set of elements currently being operated on is the entire narrow band; in this case \mathcal{S} can simply be updated by setting $\mathcal{S} = \mathcal{T}_{\text{NB}}$. Another common case will be that where the set of elements to be operated on is defined, $\mathcal{S} = \mathcal{T}_T$, that is the set of elements which were previously outside of the narrow band, but are now members of the narrow band. In this case, the set of elements comprising \mathcal{S} is updated as the children of the elements originally in \mathcal{S} which are also still inside the narrow band after it is updated, \mathcal{T}_{NB1} . As by definition these elements are on the edge of the narrow band, it is possible that by the combination of mesh refinement and updating the narrow band, the set of elements comprising, \mathcal{S} , can become equal to the empty set, \emptyset , at which point the refinement loop will break.

6.3.8 Update the level set function

Once the mesh has been refined, should the `problemHandle` indicate that this is either a reinitialisation or an extrapolation, it should now be possible to compute either of these problems and have them converge such that the error is reduced. If the `problemHandle` variable indicates that a reinitialisation is to occur at this stage the Elliptic Reinitialisation method presented in Section 4.4 will be computed until it satisfies a stopping criterion. If instead the `problemHandle` variable indicates that an extrapolation is to occur the extrapolation method presented in Section 5.2.1 will instead be solved until a stopping criterion is satisfied. For either of these problems, one of the stopping criteria will be defined by the `maxIt` input variable, which defines a maximum number of iterations for which the problem can be computed. After the level set function has been updated (should this be required), the global error estimate is again computed, and should the converged solution of the reinitialisation problem still not satisfy the tolerance on the error, the refinement loop continues, else the refinement loop breaks.

6.4 Level set reinitialisation with adaptive mesh refinement: numerical examples

In order to investigate the efficacy of the proposed refinement strategy a number of numerical examples will be presented including the repetition of the example reinitialisation problems presented in Section 4.4.1, using *hp*-adaptive meshes. In these example problems, a function which is not a signed distance function will be projected onto an initial mesh, which is then narrow banded, and the initially projected level set function will then be reinitialised on the narrow band. At this point the refinement algorithm will be called with the following inputs, $[\tilde{\phi}, \mathcal{T}_{\text{NB}}] = \text{refineUpdate}(\tilde{\phi}^0, \mathcal{T}_{\text{NB}}, \mathcal{T}_{\text{NB}}, 1\text{e-}6, [0.08 \ 0.75], 8, 8, \text{'signedDistanceError'}, 0.25, \text{'initialiseNonSDF'}, 100)$. Each example problem will be defined by the initial level set function, $\tilde{\phi}^0$, which is passed into the function, these will be defined separately for each example

below. All of the other parameters will be constant for each of the reinitialisation examples to be presented in this section. The set of elements to be operated on in each case is all of the elements in the initial narrow band, that is $\mathcal{S} = \mathcal{T}_{\text{NB}}$. The error measure to be used to drive the refinement in this case is defined as the signed distance error measure. An element will be flagged for refinement if the local error on that element is greater than 75% of the element with the maximum error, and flagged for derefinement if the local error on that element is less than 8% of the element with the maximum error. Whether the solution on an element is sufficiently smooth, Θ_τ , to be refined in p over h is defined by the parameter `analyticityThreshold`, which is chosen as `$\Theta_\tau < \text{analyticityThreshold} = 0.25$` . Once the refinement has occurred, the problem handle, `initialiseNonSDF`, indicates that the initial function should be reprojected onto the new mesh, and then the reinitialisation problem solved once again on the new mesh. The reinitialisation problem is computed until one of its stopping criteria is satisfied, which is either the criterion defining convergence, that is $\|\tilde{\phi}^{m-1} - \tilde{\phi}^m\|_2 < 10^{-10}$ or a maximum number of iterations of the reinitialisation problem have been computed, which for all examples here is chosen as 100 iterations. It is noted here that the examples in this section do not use the Anderson acceleration algorithm presented in Section 5.3, instead using the standard Picard iterative method, so that the results can be compared with those in Chapter 4. The refinement loop will be broken once either, a threshold on the error estimate is satisfied, that is $E_{\text{SD}} < 10^{-6} \times \text{area}$, where `area` is the area of the narrow band, or where all of the elements flagged for further refinement are maximally refined given the upper bounds on refinement, which in this case will be 8 levels of h -refinement, and p -refinement up to and including 8th order polynomials. After each refinement step error data will be collected in the DG, and L^2 norms, as well of course as the value of the signed distance error which is used to drive the refinement, all of which are defined in Section 4.4.1.1.

6.4.1 **hp-convergence study: circular interface**

The first experiment presented here is the reinitialisation of a level set function which can be described analytically by the quadric (4.58) in the domain $\Omega = (-2, 2)^2$, which corresponds to a circular interface centred at the origin. The signed distance function with the same interface, and therefore the analytical solution to the reinitialisation of (4.58) is stated in Equation (4.59). An hp -convergence study is performed, as described above, whereby the level set function is initialised on a Cartesian mesh with square elements of size, $h = 0.4$, of uniform polynomial order, $p = 2$, which is then, after each reinitialisation, adaptively refined and the initial level set function reprojected onto the new mesh, which continues in a loop until one of the stopping criteria is satisfied.

Figure 6.1 presents the error at which the Elliptic Reinitialisation method converges, for the circular interface problem on a sequence of adaptively refined meshes. The x -axis denotes a measure of the density of each of the meshes by taking the square root of the total number of degrees of freedom inside the narrow band. The reported number of degrees of freedom is chosen as the number of degrees of freedom inside the narrow band, as this is the computational domain over which both the problem is solved and the error is computed, and whilst there are more degrees of freedom in the domain (outside the narrow band) no solution exists on those

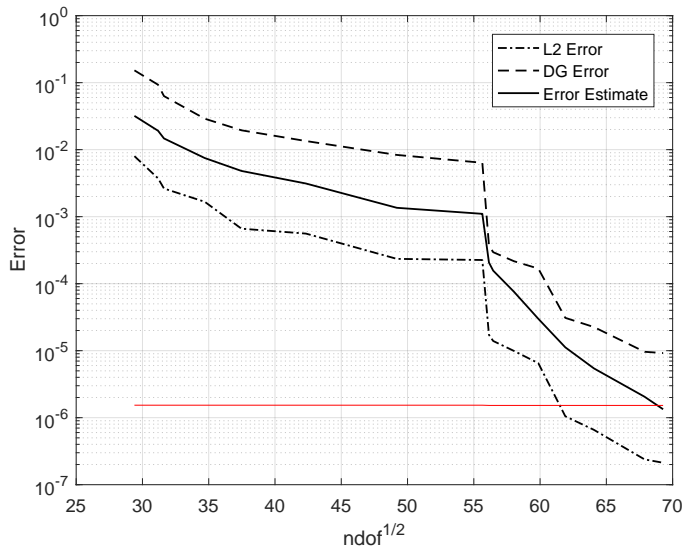


Figure 6.1: Error against the square root of the number of degrees of freedom (ndof), for the solution of the circular interface reinitialisation problem, on an hp -adaptively refined mesh. The red line denotes the error tolerance defining one of the stopping criteria, which can change as the area of narrow band changes.

elements. Should the curve plotted on a semi-log plot of error against the square root of the number of degrees of freedom be a straight line with a negative gradient, one can say that the error decreases exponentially with mesh refinement. It is the aim of an hp -adaptive refinement strategy, that the error should decrease exponentially with increased mesh refinement.

In Figure 6.1 it can be seen that there is a region of roughly exponential convergence. Initially there is a region of slower convergence which corresponds to the set of meshes within which the singularity at the centre of the conic signed distance function describing the circular interface, still exists within the computational domain, as the mesh isn't sufficiently fine such that the criterion defining the narrow band (see Section 5.2) is sufficient to remove this part of the mesh from the narrow band. In this region the convergence is relatively slow. As soon as the mesh is sufficiently refined to remove the singularity the error decays much more quickly. During the initial region of slower convergence it can be noted that even though the largest errors are near to the singularity at the centre of the circular interface on the reinitialised level set function and thus the refinement flags appropriately positioned, the `computeAnalyticityTest` function computes that the solution on these elements is smooth enough for p -refinement. The reason for this is that the analyticity estimate is based on a finite Legendre expansion, which for low order p will only have $p + 1$ points which could be too few to compute an accurate estimate. This is compounded with the fact that when there are few Legendre coefficients, the coefficient associated with the constant Legendre function has greater weight on the apparent decay rate of the remaining coefficients and thus the same singular function with greater magnitude will appear to decay faster. For these reasons the poor accuracy in the singular region is maintained until the upper limit on p -refinement is reached, that being $p = 8$ for this example problem, at which point the `enforceRefinementLimits` function, will switch the refinement flags from p to

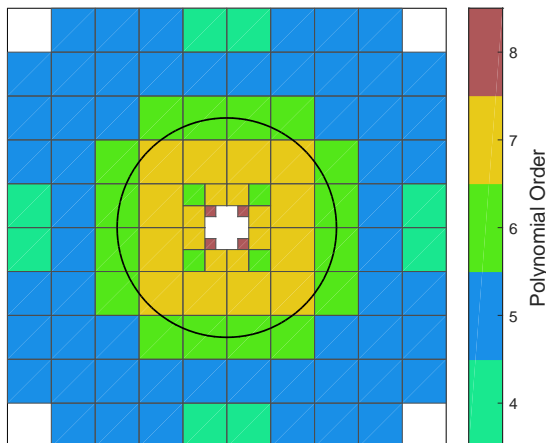


Figure 6.2: Final computed mesh configuration for the circular interface reinitialisation problem on an hp -adaptively refined mesh where the colour of the element denotes the polynomial order of that element. The thick black line denotes the computed interface position.

h -refinement flags, which finally allows the mesh to become fine enough to remove the singularity from the narrow band. This is reflected in the final mesh shown in Figure 6.2. Beyond this point the problem is smooth and the rate of convergence increases.

It can also be seen in Figure 6.1, that the signed distance error roughly bounds the L^2 error from above and the DG error from below. In the earlier numerical examples in this thesis where a level set function is reinitialised on a series of fixed Cartesian meshes, generally speaking this same pattern could be noticed. This was less true during evolution, as reinitialisation during evolution can allow one to maintain a constant signed distance error, especially in the case that the curvature of the interface remains roughly constant throughout the evolution, whereas the DG and L^2 error norms are defined relative to an analytical solution and therefore additional error at each step which accumulates over time using these measures is not reflected in the signed distance error estimate. However, there still seems to be a relationship between these quantities as of course an interface evolving such that it becomes more complicated means that it is more difficult to capture on a mesh and therefore both reinitialise and evolve.

The mesh shown in Figure 6.2, is the first mesh upon which the converged solution to the circular interface reinitialisation problem satisfies a stopping criterion of the mesh refinement loop, in this case the tolerance on the signed distance error. It can be seen in this figure, first of all, that the area of greatest refinement is near to the singularity at the vertex of the conic level set function which describes the signed distance function to the circular interface. The effect of the smoothing algorithm enforcing the bounds on local variation in both h and p is also obvious from the figure from the radial staircase pattern in the polynomial order spreading outwards from the singularity. Another point of note is that due to the regular distribution of square elements in the initial mesh, which is vastly maintained due to the smoothness of the solution away from the singularity, the level set interface in elements near to the $x = 0$ and $y = 0$ planes aligns more congruently with the background grid, whereas this is not the case elsewhere which is reflected in the position of the lowest order elements. As will be the case

for all of the remaining examples in this chapter, the thick black line denoting the position of the level set interface is drawn using the built-in MATLAB function `contour`, interpolating on each element at a set of $(p_\tau + 1)^2$ equidistant interpolation points on each element. Further, it should also be noted that the white space inside the domain indicates the positions of the set of elements which are not inside the narrow band, and thus that white is never used as a colour to denote polynomial order.

6.4.2 **hp-convergence study: square interface**

A second example problem to be presented here is the reinitialisation of a square interface, which is not aligned with the grid, and which is offset from the origin slightly such that the singular regions will not align with mesh nodes. The initial level set function to be projected to the level set interface can be described by the analytical function,

$$\tilde{\phi}^0(\mathbf{x}) = \max \left(\left| \frac{x}{2} - \frac{\sqrt{3}y}{2} - 0.02 \right|, \left| \frac{x}{2} + \frac{\sqrt{3}y}{2} \right| \right) - 1.06666, \quad (6.3)$$

on the domain, $\Omega = (-2, 2)^2$. The initial interface associated with Equation (6.3) and its relative position in the domain can be seen in Figure 6.3. This initial level set function is a signed distance function everywhere except at the edges of the pyramidic level set function which coincides with the vertices of the square interface. Thus Equation (6.3) is the analytical solution to the problem. By virtue of the singular regions in the problem, however, the signed distance error is large over the domain, and based on this criterion reinitialisation is required. Reinitialisation is dissipative and will therefore tend to round these corners, however, the boundary condition will aim to keep the position of the interface unchanged, and thus the shape should largely remain unchanged. An *hp*-convergence study is performed, as described above, whereby the level set function is initialised on a Cartesian mesh with square elements of size, $h = 0.4$, of uniform polynomial order, $p = 2$, which is then, after each reinitialisation, adaptively refined and the initial level set function reprojected onto the new mesh, which continues in a loop until one of the stopping criteria is satisfied.

The variation in error with mesh density for the reinitialisation of the square interface on an *hp*-adaptively refined mesh is shown in Figure 6.4. What can be observed is that the magnitude of the negative gradient is increasing with refinement indicating exponential convergence, the rate of which is increasing. This is because *h*-refinement is focussed in the singular regions which causes a narrowing of the narrow band and thus the size of the singular regions to shrink. The interface itself is singular in this case which of course cannot be narrow banded out of the domain, however, given the maximum possible levels of refinement, the size of the singular regions in the domain will be appropriately minimised, as can be seen in Figure 6.5. One point of note however, is that in this instance, due to the singular regions always remaining inside the narrow band, the stopping criterion based on the error over the area of the narrow band, denoted in Figure 6.4 by the red line, cannot be satisfied given the allowed level of refinement. Also it can be noted, even in the case of a singular interface, that the proposed error estimate maintains the apparent relationship with the L^2 norm, bounding it roughly from above, and with the DG

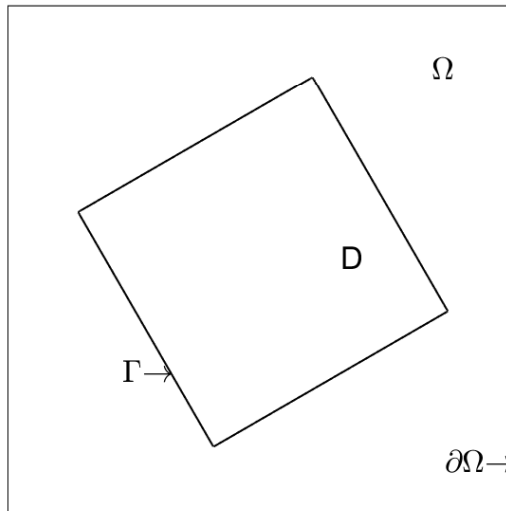


Figure 6.3: Relative position of the initial interface and the domain boundaries for the square interface reinitialisation problem on the domain $\Omega = (-2, 2)^2$.

norm, following roughly the pattern in this error norm and bounding it from below.

Figure 6.5(a) shows the first mesh upon which the converged solution to the square interface reinitialisation problem satisfies a stopping criterion of the mesh refinement loop, which in this case is as a result of the level set function being maximally refined given the allowed upper limits on refinement. It can be seen in Figure 6.5 that almost all of the refinement is along the singular regions; Figure 6.5(b) shows that the elements are maximally refined in both h and p where the solution is singular and most of the remaining refinement in the rest of the domain is just to enforce the local bounds on variation in h and p . The position of the computed level set interface is largely maintained by virtue of the boundary condition, although as can be seen in Figure 6.5(b), the element which is cut by the singular part of the interface is unable to sharply capture this feature, although this loss in accuracy is minimised by the adaptive refinement. This aligns with what would be expected for the given problem.

6.4.3 hp-convergence study: smooth star interface

The next example is another more difficult problem, referred to in Section 4.4.1.4 as a smooth star interface reinitialisation problem. That is the level set function is initialised as the L^2 projection of

$$\tilde{\phi}^0(\mathbf{x}) = x^2 + y^2 - \left(1 - 0.4 \sin\left(6 \arctan\left(\frac{y}{x}\right)\right)\right), \quad (6.4)$$

in the domain $\Omega = (-2, 2)^2$. This is similar to the smooth star interface problem from Section 4.4.1.4, however, this time chosen such that the level set interface has a slightly larger curvature. The shape of such an interface and its position relative to the domain boundaries can be seen in Figure 6.6. The signed distance function to this smooth star interface can be computed

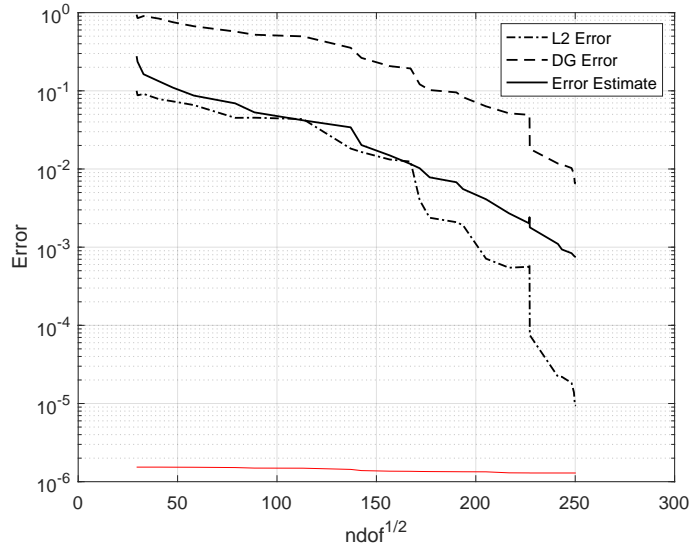


Figure 6.4: Error against the square root of the number of degrees of freedom (ndof), for the solution of the square interface reinitialisation problem, on an hp -adaptively refined mesh. The red line denotes the error tolerance defining one of the stopping criteria, which can change as the area of narrow band changes.

numerically by solving the minimisation problem

$$\begin{aligned} \tilde{\phi}(\mathbf{x}, \vartheta) &= \min_{\vartheta} \psi \\ &= \min_{\vartheta} \left(\sqrt{(x - \sqrt{1 - 0.4 \sin(6\vartheta)} \cos(\vartheta))^2 + (y - \sqrt{1 - 0.4 \sin(6\vartheta)} \sin(\vartheta))^2} \right). \end{aligned} \quad (6.5)$$

That is finding the roots to the equation,

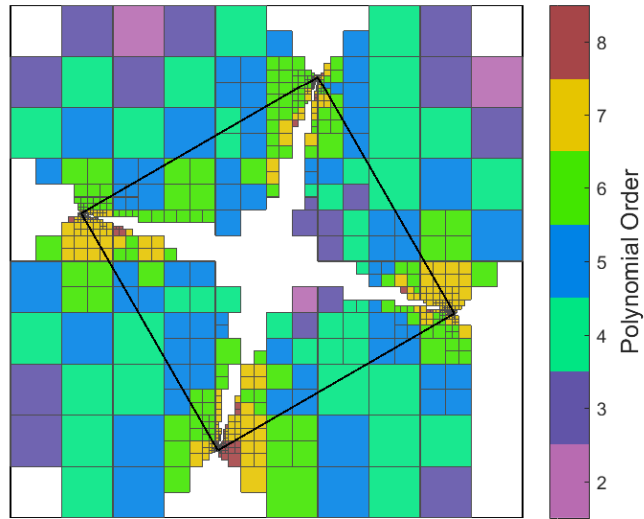
$$\begin{aligned} \frac{d\psi}{d\vartheta} &= 2 \left(\sin(\vartheta) \sqrt{1 - 0.4 \sin(6\vartheta)} + \frac{(1.2 \cos(\vartheta) \cos(6\vartheta))}{\sqrt{1 - 0.4 \sin(6\vartheta)}} \right) \left(x - \sqrt{1 - 0.4 \sin(6\vartheta)} \cos(\vartheta) \right) \\ &\quad + 2 \left(y - \sin(\vartheta) \sqrt{1 - 0.4 \sin(6\vartheta)} \right) \left(-\sqrt{1 - 0.4 \sin(6\vartheta)} \cos(\vartheta) + \frac{(1.2 \sin(\vartheta) \cos(6\vartheta))}{\sqrt{1 - 0.4 \sin(6\vartheta)}} \right) \\ &= 0, \end{aligned} \quad (6.6)$$

using a bisection method with a tolerance of $|\vartheta| < 10^{-15}$. Using the numerical solution above to find the value of the solution at a point \mathbf{x} , the gradient of the signed distance function to the smooth star interface can be computed using a first-order finite difference method as follows

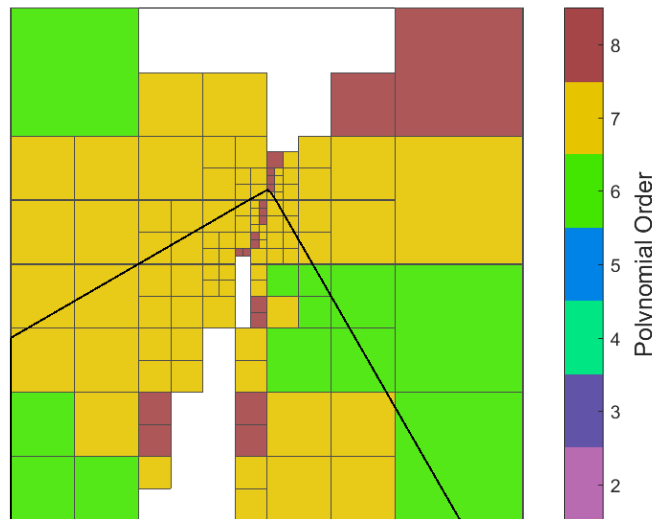
$$\left\{ \begin{array}{c} \frac{\partial \tilde{\phi}(x, y)}{\partial x} \\ \frac{\partial \tilde{\phi}(x, y)}{\partial y} \end{array} \right\} = \left\{ \begin{array}{c} \frac{\tilde{\phi}(x, y) - \tilde{\phi}(x + \Delta x, y)}{\Delta x} \\ \frac{\tilde{\phi}(x, y) - \tilde{\phi}(x, y + \Delta x)}{\Delta x} \end{array} \right\}, \quad (6.7)$$

with spatial step size, $\Delta x = 10^{-12}$.

Again an hp -convergence study is performed, as described above, whereby the level set func-



(a) Entire narrow banded adaptive mesh over the domain.



(b) Mesh near to one of the singular regions.

Figure 6.5: Final computed mesh configuration for the square interface reinitialisation problem on an hp -adaptively refined mesh where the colour of the element denotes the polynomial order of that element. The thick black line denotes the computed interface position.

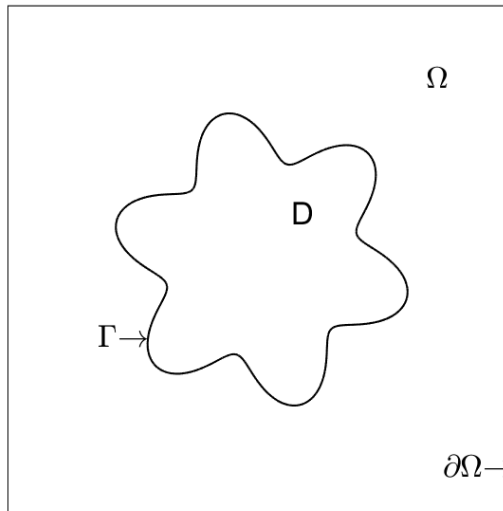


Figure 6.6: Relative position of the initial interface and the domain boundaries for the smooth star interface reinitialisation problem, defined by Equation (6.4), on the domain $\Omega = (-2, 2)^2$.

tion is initialised on a Cartesian mesh with square elements of size, $h = 0.4$, of uniform polynomial order, $p = 2$, which is then, after each reinitialisation, adaptively refined and the initial level set function reprojected onto the new mesh, which continues in a loop until one of the stopping criteria is satisfied. The variation in error with mesh density during this study for the smooth star interface is presented in Figure 6.7. As with the previous two examples it can be seen that there is roughly exponential convergence, with initially a slower rate of convergence where the singularities remain inside the narrow band and then a faster rate of convergence once the singularities are narrow banded out. Also it can again be seen that the error estimate roughly follows the pattern of the DG norm, bounding it from below, and providing an upper bound on the L^2 norm. Towards the end of the hp -convergence study, however, it can be seen that the rate of convergence slows again and almost begins to stagnate. One reason for this is that the error tolerance is not satisfied by the time those elements requiring refinement have reached the maximum allowed polynomial order of $p = 8$. As these elements are still flagged for refinement, the `enforceRefinementLimits` function switches the flag from p to h , and h -refinement is known to be less efficient than p for smooth problems, and therefore this ultimately limits the rate of convergence. This can be seen to be the case by noticing where the error tolerance shrinks as a result of shrinking narrow band (see the red line on Figure 6.7), which occurs when the elements near to the interface become small as a result of h -refinement which coincides with the stagnating error.

This issue, however, is perhaps more insidious than simply enforcing an upper limit on the amount of allowed p -refinement. For example, an important question one could ask at this point, is ‘why is the method unable to satisfy the error tolerance?’; especially given that the problem is smooth, the problem is discretised using what would be considered by many in the finite element community as very high-order polynomial bases, and the Elliptic Reinitialisation method solving this type of problem on Cartesian meshes of fixed polynomial order were shown

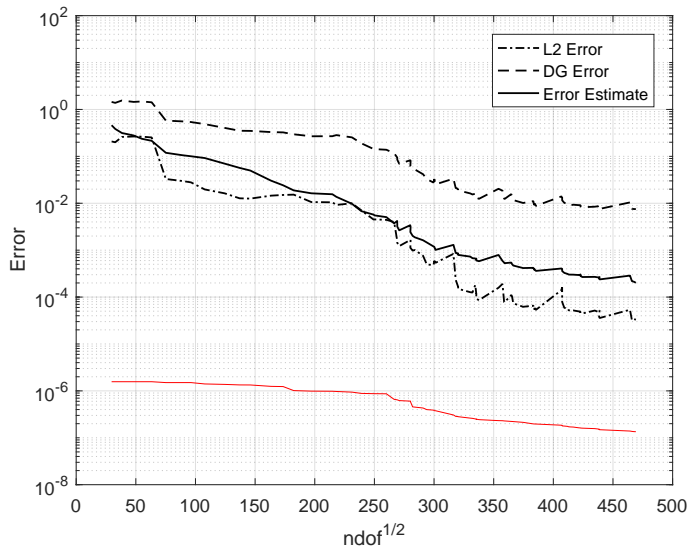


Figure 6.7: Error against the square root of the number of degrees of freedom (ndof), for the solution of the smooth star interface reinitialisation problem, initialised as in Equation (6.4), on an hp -adaptively refined mesh. The red line denotes the error tolerance defining one of the stopping criteria, which can change as the area of narrow band changes.

in Section 4.4.1.4 to converge optimally. One reason for this is that at each refinement step, when the reinitialisation routine is called, the required number of iterations for the reinitialisation to be stopped as a result of satisfying the convergence criterion, would, if allowed to continue until that point, dwarf the chosen maximum number of iterations. In other words the errors presented in Figure 6.7 are always after 100 iterations, as this is the maximum allowed number of iterations, which on the increasingly well refined meshes seems to not be large enough for the new mesh to outperform the previous mesh. This was not observed in similar earlier examples, as there was no limit on the maximum number of iterations.

Investigating the relative change in the solution, as well as the error in the solution, for this (or any) example problem, using the Elliptic Reinitialisation method (using Picard’s method as the iterative solver), one can see that both of these quantities decrease monotonically when plotted against the number of computed iterations. Furthermore, the numerical examples in Section 4.4.1, present evidence suggesting that when the Elliptic Reinitialisation method is allowed to reach this convergence criterion, the method will converge optimally in h for fixed p . In other words, without consulting practical limitations on time and memory, the Elliptic Reinitialisation method is capable of returning a suitable signed distance function. For this specific example however, the convergence rate of the Picard iterative method is so slow that it almost stagnant, and thus placing a limit on the number of iterations, for reasons of practicality, seems to necessarily mean that the convergence of the refinement will similarly be limited.

It was attempting to overcome this issue which lead investigations into methods such as Anderson acceleration (see Section 5.3) and even Newton schemes (results not presented) for the Elliptic Reinitialisation method, as well as the various temporal discretisations of the Parabolic Reinitialisation method. What was found was that attempts at increasing the rate of convergence

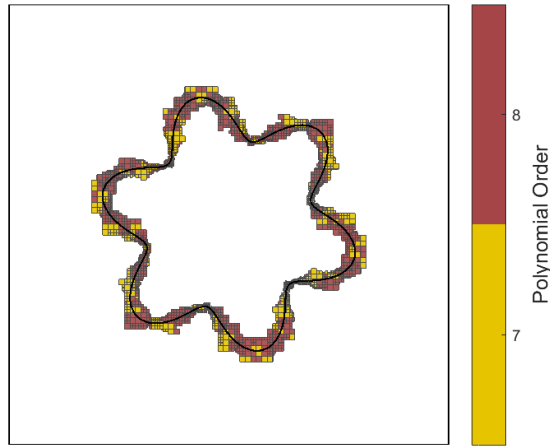


Figure 6.8: Final computed mesh configuration for the smooth star interface reinitialisation problem, initialised as in Equation (6.4), on an hp -adaptively refined mesh where the colour of the element denotes the polynomial order of that element. The thick black line denotes the computed interface position.

would invariably lead to instability. One of the advantages of Anderson acceleration in this respect, and why it is used here, is that when instabilities become apparent it can simply revert to Picard, which has demonstrated stability when solving these problems. It seems ultimately then, that in order to overcome this issue one must either; allow more time for reinitialisation, that is choosing a larger maximum allowed number of iterations, and/or use more powerful hardware and therefore allow for higher levels of refinement and attempt to save lost time using by parallelising the computation. Discussions relating to this issue follow in Sections 6.4.4 and 6.6.4, however, beyond that stated here a suitable solution was unfortunately not found during this research period.

The final mesh configuration for the adaptively refined smooth star reinitialisation problem is shown in Figure 6.8. As a result of the stagnation in the error, the solution never reaches the desired tolerance on the signed distance error. Instead the mesh is configured such that it reaches the upper limit on first p and then eventually h , and is therefore deemed to be optimally refined given these limits. This explains why the mesh looks how it does with the majority of the elements being of order, $p = 8$, and forming a much narrower narrow band near to the interface than either of the previous examples.

6.4.4 hp -convergence study: elliptic interface

As the behaviour presented in the case of smooth star interface problem, is not reflected in the circular interface reinitialisation example, the complexity of the circular interface problem can be increased slightly to investigate this change in behaviour. To increase the difficulty of the circular interface problem one can generalise the quadric defining the circular interface by simply introducing a major and minor axis and thus the circular interface becomes an ellipse. More specifically one can project an initial level set function defined as

$$\tilde{\phi}^0 = \left(\frac{x}{a}\right)^2 + \left(\frac{x}{b}\right)^2 - 1, \quad (6.8)$$

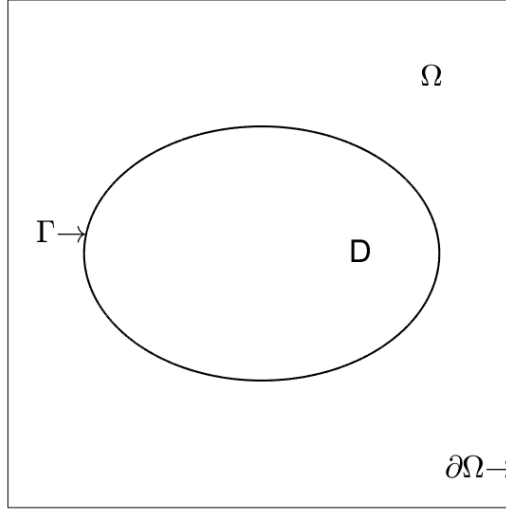


Figure 6.9: Relative position of the initial interface and the domain boundaries for the elliptical interface reinitialisation problem on the domain $\Omega = (-2, 2)^2$.

choosing $a = 1.4$ and $b = 1$, onto the domain $\Omega = (-2, 2)^2$. The relative position of the interface to the domain boundaries as a result of this projection can be seen in Figure 6.9. The signed distance function with the same interface, and therefore the solution to the reinitialisation of (6.8) can be found at a point \mathbf{x} by computing the real roots, $\vartheta_{\mathbf{x}} = \Re(\vartheta_x)$, to the equation

$$(a^2 - b^2) \vartheta_x^4 - (2a^2x(a^2 - b^2)) \vartheta_x^3 + \left(a^2 (a^2x^2 + b^2y^2 - (a^2 - b^2)^2) \right) \vartheta_x^2 + (2a^4x(a^2 - b^2)) \vartheta_x - (a^6x^2) = 0. \quad (6.9)$$

Then one can compute

$$\vartheta_{\mathbf{y}} = \text{sign}(y) \sqrt{1 - \frac{\vartheta_{\mathbf{x}}^2}{a^2}}, \quad (6.10)$$

to get the analytical solution at a point \mathbf{x} , which be stated

$$\tilde{\phi} = \min \left\{ \sqrt{(x - \vartheta_{\mathbf{x}})^2 + (y - \vartheta_{\mathbf{y}})^2} \right\}. \quad (6.11)$$

Equation (6.9) is solved in this work using the MATLAB built-in function `roots`, which computes the eigenvalues of the companion matrix to the polynomial (6.9). Where a known gradient of the solution is required for the computation of the DG norm, this can be computed numerically by first finding the known solution at points in the domain, and using a first-order finite difference method, as in Equation (6.7), again with spatial step size, $\Delta x = 10^{-12}$.

Again an hp -convergence study is performed, as described previously, whereby the level set function is initialised on a Cartesian mesh with square elements of size, $h = 0.4$, of uniform polynomial order, $p = 2$, which is then, after each reinitialisation, adaptively refined and the initial level set function reprojected onto the new mesh, which continues in a loop until one of the stopping criteria is satisfied. Figure 6.10 shows the results of the hp -convergence study for

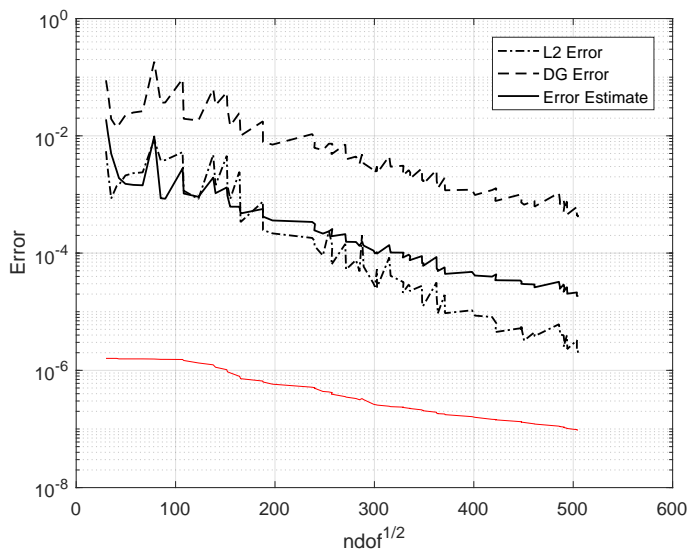
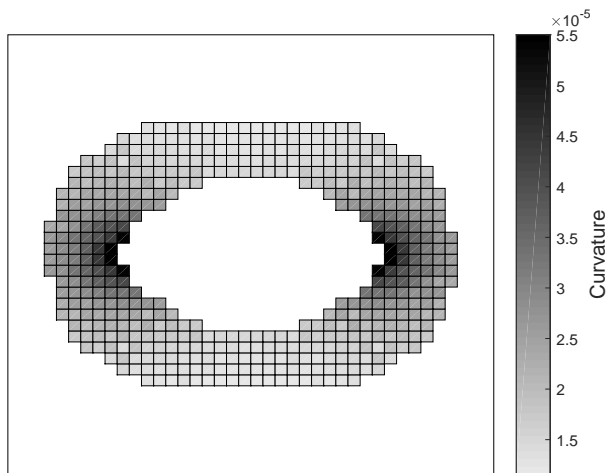


Figure 6.10: Error against the square root of the number of degrees of freedom (ndof), for the solution of the elliptical interface reinitialisation problem, on an hp -adaptively refined mesh. The red line denotes the error tolerance defining one of the stopping criteria, which can change as the area of narrow band changes.

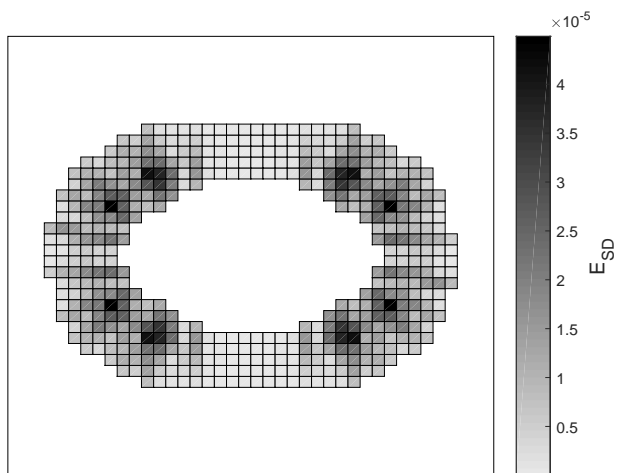
the elliptical interface reinitialisation problem. For this problem, there is once again a period of slow convergence to begin with, whereby the singular region at the centre of the ellipse remains inside the narrow band. Once the mesh is sufficiently refined for the narrow band to be narrow enough that these regions no longer exist within the computational domain the rate of exponential convergence increases. Once again for this problem the mesh becomes maximally refined based on the upper limits placed on h and p before it can satisfy the error tolerance. Whilst the stagnation which was seen for the smooth star example does not appear to be present in this example, the issue with the smooth star problem remains present here in that the required number of Picard iterations is much larger than the imposed limit, which seems to limit the rate of convergence of this refinement study.

That this is not an issue for the circular interface reinitialisation problem but is for the elliptical interface reinitialisation problem implies that there is something about the increased curvature of the problem which is causing the reinitialisation method to struggle to solve the problem in an efficient manner. To see the effect that this increased curvature has, one can compute the reinitialisation of the elliptical interface problem for 100 iterations using a fixed mesh with $h = 0.1$ and $p = 5$. Figure 6.11 shows the variation in the L^2 and signed distance error measures as well as the curvature, over the domain for the solution of this reinitialisation problem. It can be seen in Figure 6.11 that the region with the largest curvature, that being the area near the vertices of the ellipse, is not the area of greatest error in either of the norms. This implies that it is either the variation in curvature, and/or the orientation of the interface relative to the grid which seems to be the cause of this issue.

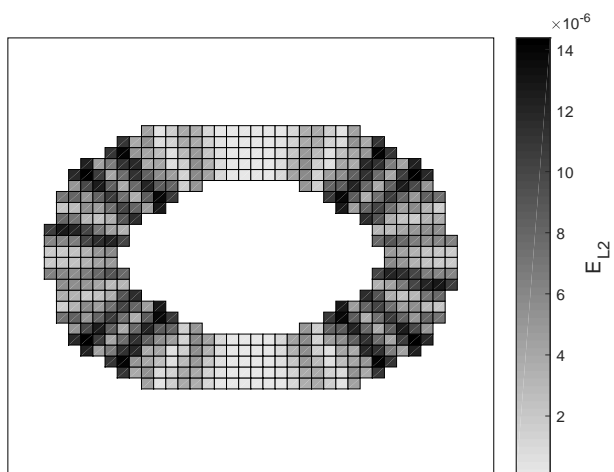
Recomputing the circular interface reinitialisation problem defined by the initial level set



(a) Curvature

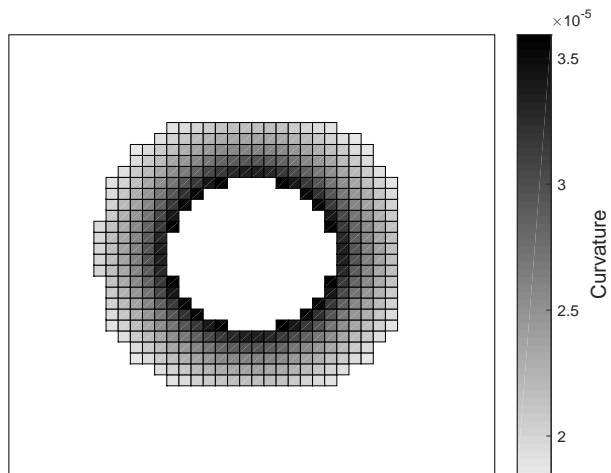


(b) Signed distance error

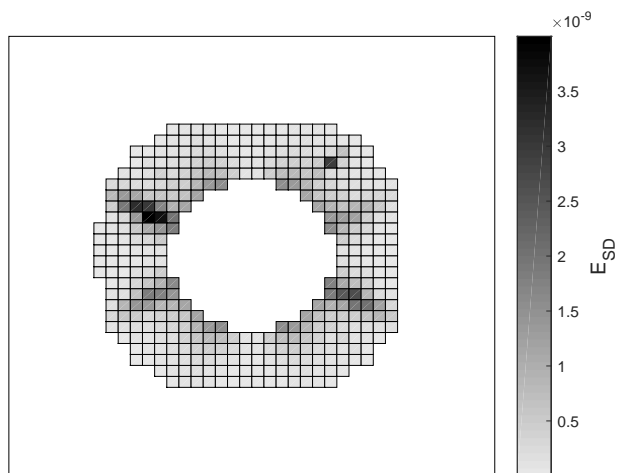


(c) L^2 error

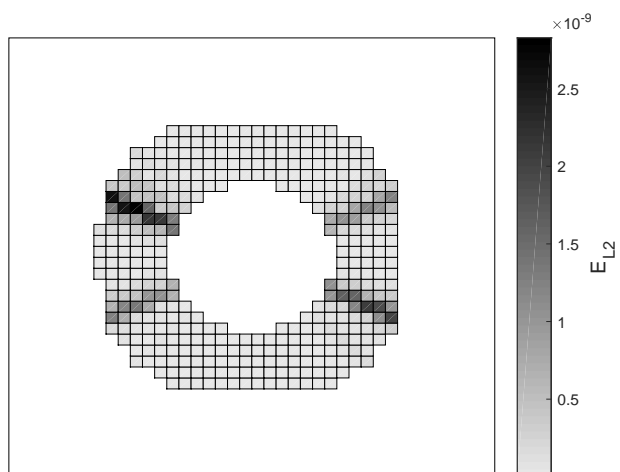
Figure 6.11: Error and curvature distributions for the elliptic interface reinitialisation problem on a narrow banded Cartesian mesh after 100 iterations.



(a) Curvature



(b) Signed distance error



(c) L^2 error

Figure 6.12: Error and curvature distributions for the circular interface reinitialisation problem on a narrow banded Cartesian mesh after 100 iterations.

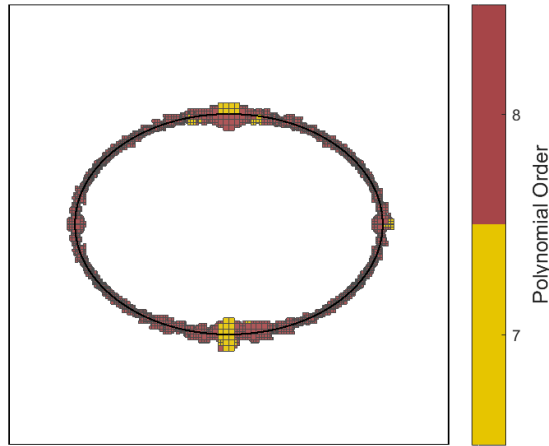


Figure 6.13: Final computed mesh configuration for the elliptical interface reinitialisation problem on an hp -adaptively refined mesh where the colour of the element denotes the polynomial order of that element. The thick black line denotes the computed interface position.

function (4.58) on the same fixed mesh produces the results shown in Figure 6.12. Despite being well resolved almost everywhere, the area of greatest error for the circular interface problem, aligns with a similar region as in the ellipse problem; that being where the orientation of the interface begins to diverge from the grid in one axis. A possible solution to this issue might be to use r -adaptivity to reorient the elements along the interface such that there is greater congruence in the alignment of the interface with the element edges, however, this is beyond the remit of the work to be presented here. As such we instead move forward with the knowledge that in looking to the appropriate balance between accuracy and computational expense the stopping criteria for the reinitialisation problem are of paramount importance. A similar example in this respect will be explored in Section 6.6.4, where an evolution problem is presented by which an initially circular interface evolves into an ellipse.

Figure 6.13 shows the first mesh upon which the converged solution to the elliptical interface reinitialisation problem satisfies a stopping criterion of the mesh refinement loop, which in this case is as a result of the level set function being maximally refined given the allowed upper limits on refinement. For the reasons discussed above the computed mesh, is highly refined in both h and p and as such, is reminiscent of the mesh computed during the smooth star interface reinitialisation problem.

6.4.5 hp -convergence study: multiple interfaces

The final reinitialisation example will be a repeat of the multiple interface problem presented in Section 4.4.1.5. For this example, the function defined by Equations (4.62) and (4.63) will be L^2 projected onto the domain, $\Omega = (-2, 2)^2$. The interface will form three closed interfaces of varying curvature, two of which are nested within the third; the domain/interface configuration can be seen in Figure 4.12. The signed distance function to the multiple interfaces can be computed at a point \mathbf{x} , by solving the following problem numerically,

$$\tilde{\phi}(\mathbf{x}) = \max(\tilde{\phi}_k(\mathbf{x})), \quad k = 1, 2, 3, \quad (6.12)$$

where

$$\begin{aligned}\tilde{\phi}_1(\mathbf{x}) &= \min_{\vartheta_1} \psi_1, \\ &= \min_{\vartheta_1} \left(\left[(x - (1 + 0.8 \sin^2(\vartheta_1)) \cos(\vartheta_1))^2 + (y - (1 + 0.8 \sin^2(\vartheta_1)) \sin(\vartheta_1))^2 \right]^{\frac{1}{2}} \right),\end{aligned}\quad (6.13)$$

$$\begin{aligned}\tilde{\phi}_2(\mathbf{x}) &= \min_{\vartheta_2} \psi_2, \\ &= \min_{\vartheta_2} \left(\left[(x - (0.3 - 0.075 \sin(4\vartheta_2)) \cos(\vartheta_2))^2 + \right. \right. \\ &\quad \left. \left. ((y - 0.8) - (0.3 - 0.075 \sin(4\vartheta_2)) \sin(\vartheta_2))^2 \right]^{\frac{1}{2}} \right),\end{aligned}\quad (6.14)$$

$$\begin{aligned}\tilde{\phi}_3(\mathbf{x}) &= \min_{\vartheta_3} \psi_3, \\ &= \min_{\vartheta_3} \left(\left[(x - (0.48 - 0.08 \sin^2(4\vartheta_3)) \cos(\vartheta_3))^2 + \right. \right. \\ &\quad \left. \left. ((y + 0.65) - (0.48 - 0.08 \sin^2(4\vartheta_3)) \sin(\vartheta_3))^2 \right]^{\frac{1}{2}} \right).\end{aligned}\quad (6.15)$$

That is by finding the roots to the following

$$\begin{aligned}\frac{\partial \psi_1}{\partial \vartheta_1} &= x \sin(\vartheta_1) (1.6 \sin^2(\vartheta_1) + 2) - 3.2x \sin(\vartheta_1) \cos^2(\vartheta_1) + \cos(\vartheta_1) (2.56 \sin^5(\vartheta_1) + \\ &\quad 3.2 \sin^3(\vartheta_1) - 4.8y \sin^2(\vartheta_1) - 2y) + \cos^3(\vartheta_1) (2.56 \sin^3(\vartheta_1) + 3.2 \sin(\vartheta_1)) = 0,\end{aligned}\quad (6.16)$$

$$\begin{aligned}\frac{\partial \psi_2}{\partial \vartheta_2} &= 2 (0.75 \sin(\vartheta_2) (0.4 - 0.1 \sin(4\vartheta_2)) + 0.3 \cos(\vartheta_2) \cos(4\vartheta_2)) (x - 0.75 \cos(\vartheta_2) (0.4 - \\ &\quad 0.1 \sin(4\vartheta_2))) + 2 ((y - 0.8) - 0.75 \sin(\vartheta_2) (0.4 - 0.1 \sin(4\vartheta_2))) \\ &\quad (0.3 \sin(\vartheta_2) \cos(4\vartheta_2) - 0.75 \cos(\vartheta_2) (0.4 - 0.1 \sin(4\vartheta_2))) = 0,\end{aligned}\quad (6.17)$$

$$\begin{aligned}\frac{\partial \psi_3}{\partial \vartheta_3} &= 2 (0.8 \sin(\vartheta_3) (0.6 - 0.1 \sin^2(4\vartheta_3)) + 0.64 \sin(4\vartheta_3) \cos(\vartheta_3) \cos(4\vartheta_3)) (x - \\ &\quad 0.8 \cos(\vartheta_3) (0.6 - 0.1 \sin^2(4\vartheta_3))) + 2 ((y + 0.65) - 0.8 \sin(\vartheta_3) (0.6 - \\ &\quad 0.1 \sin^2(4\vartheta_3))) (0.64 \sin(\vartheta_3) \sin(4\vartheta_3) \cos(4\vartheta_3) - 0.8 \cos(\vartheta_3) (0.6 - 0.1 \sin^2(4\vartheta_3))) \\ &= 0,\end{aligned}\quad (6.18)$$

using a bisection method with a tolerance of $|\vartheta| < 10^{-15}$. Where a known gradient of the solution is required for the computation of the DG norm, this can be computed numerically by first finding the known solution at points in the domain, and using a first-order finite difference method, as in Equation (6.7), again with spatial step size, $\Delta x = 10^{-12}$.

An hp -convergence study will be computed as described previously on an initially Cartesian mesh, with square elements of size, $h = 0.4$, and of uniform polynomial order, $p = 2$. The results of the hp -convergence study for the multiple interfaces reinitialisation problem are shown in Figure 6.14. The signed distance error mirrors the previous examples by demonstrating

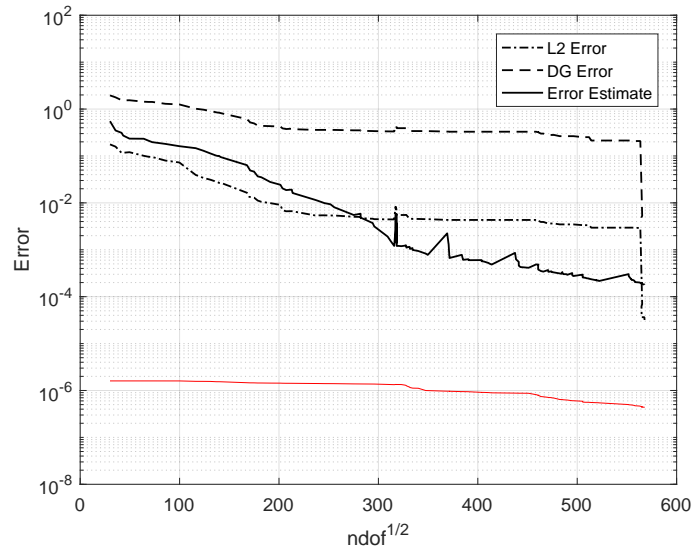


Figure 6.14: Error against the square root of the number of degrees of freedom (ndof), for the solution of the multiple interfaces reinitialisation problem, on an hp -adaptively refined mesh. The red line denotes the error tolerance defining one of the stopping criteria, which can change as the area of narrow band changes.

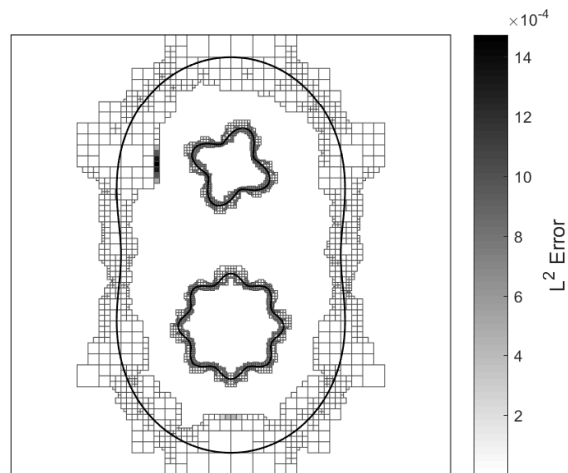


Figure 6.15: Distribution of L^2 error over the mesh for the multiple interfaces reinitialisation problem on a mesh where the analytical solution does not align with the unique viscosity solution.

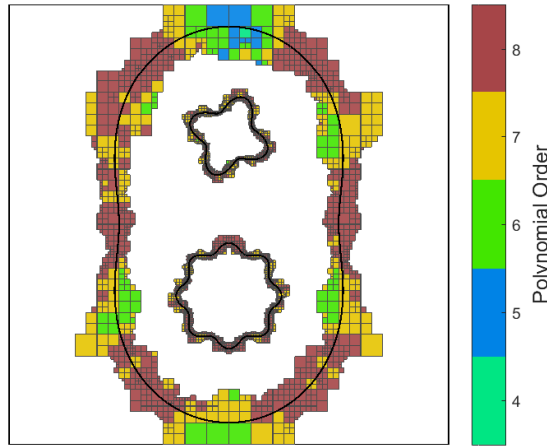


Figure 6.16: Final computed mesh configuration for the multiple interfaces reinitialisation problem on an hp -adaptively refined mesh where the colour of the element denotes the polynomial order of that element. The thick black line denotes the computed interface position.

exponential convergence, the rate of which increases as the problem becomes smoother by the removal of singular regions with h -refinement. The error in the L^2 and DG norms stagnates for many iterations. This is a function of the fact that the unique viscosity solution to the reinitialisation problem is defined by the initial condition, (4.62). If one considers the narrow band in this case to be three separate narrow bands each associated with one of the three interfaces, then some elements can belong to one narrow band, but have interpolation points which are physically closer to one of the other interfaces. Therefore the analytical solution, which simply computes the minimum distance from a given point to the interface doesn't align with the unique signed distance function which solves the reinitialisation problem. This can be seen in Figure 6.15 which shows the distribution of the L^2 error for one of the meshes during this period. This should be noted when evaluating the results of this experiment. Eventually these elements get narrow banded out of the domain and for the final few iterations of the refinement loop the error returns to what would be the expected value based on the previous results, that is, the error estimate is greater than the L^2 error and smaller than the DG error.

Figure 6.16 shows the first mesh upon which the converged solution to the multiple interfaces reinitialisation problem satisfies a stopping criterion of the mesh refinement loop. Again as a result of the more complex interface, the criterion which causes the refinement loop to break is that all of the refinement flags are set to *do not refine* due to the upper limits on h and p refinement being reached. As before this manifests in high levels of both h and p refinement, and thus a narrow, narrow band. As there are multiple interfaces in this problem, these high levels of refinement occur mostly around the two most curvaceous interfaces with lower levels of refinement surrounding the third interface. Although as was the case even with the elliptical interface problem from earlier, where the least complex interface is oriented such that it does not align with the mesh, high levels of refinement are still required to accurately compute the reinitialisation problem.

6.5 hp-adaptive level set method

This section will extend the level set methodology presented in Section 5.4, which comprised an algorithm for evolving interfaces on narrow banded Cartesian meshes of uniform arbitrary order to now include *hp*-adaptive meshes using the `refineUpdate` function (Algorithm 3) explained earlier in this chapter.

6.5.1 Evolution algorithm

A problem to be solved using the proposed level set methodology, on an *hp*-adaptive mesh, with an initial Cartesian mesh of elements of size h , with uniform polynomial order, p , will take the form presented in Algorithm 4. The first step of the algorithm is to partition the mesh and to initialise the level set function. Here this involves calling the `refineUpdate` function, to repeatedly L^2 project and adaptively refine the mesh, until the mesh is able to sufficiently resolve the level set function, where ‘sufficiently resolved’ is defined by satisfying a tolerance on the L^2 error in the projection. Once a sufficiently refined and narrow, narrow banded mesh, has been generated for representing the initial condition, the evolution loop can begin. The only change here from the evolution algorithm on Cartesian meshes of fixed polynomial order, Algorithm 2, is that after the evolution, the reinitialisation and the updating of the narrow band at each time step, the `refineUpdate` function is called, which checks whether the level set function on the mesh satisfies a tolerance on the signed distance error and in the case that it doesn’t computes the refine-reinitialise loop. Once the refine-reinitialise loop is broken, by satisfaction of one of its stopping criteria, the evolution loop then continues until $t_e^m = T$, at which point the level set function at each time step is output to the user.

It should be noted that the various input parameters to the `refineUpdate` function, are subject to change, however, those stated in Algorithm 4 will be those used for the remainder of the example problems in this chapter.

6.6 Level set evolution with adaptive mesh refinement: numerical examples

The full level set evolution algorithm with *hp*-adaptive mesh refinement, that is as defined in Algorithm 4, will now be evaluated using a number of example problems which are described in Section 5.4.2.

6.6.1 hp-convergence study: growing-shrinking-growing circle

6.6.1.1 Initialisation

The first example problem is the growing-shrinking-growing (GSG) circular interface problem, introduced in Section 5.4.2 and which was solved on a series of fixed Cartesian meshes in Section 5.4.2.1. For this problem, an initial level set function which is the signed distance function to a circular interface of radius $r = 1$, that is as defined in Equation (5.19), is L^2 projected onto the domain $\Omega = (-2, 2)^2$. The initial mesh consists of square elements of size $h = 0.4$ and of polynomial order $p = 2$, which after the initial projection is narrow banded and passed into the

Input: $\Omega, \phi^0, \text{advectionType}, T$
Output: $\phi_h(\mathbf{x}, t_e)$

Initialise problem parameters: ϕ_h^0, \mathcal{T} ;
Compute initial Narrow Band, \mathcal{T}_{NB} ;
if ϕ_h^0 *is not a signed distance function* **then**
 | Reinitialise Level set function;
 | $[\phi, \mathcal{T}_{\text{NB}}] = \text{refineUpdate}(\phi_h^0, \mathcal{T}_{\text{NB}}, \mathcal{T}_{\text{NB}}, 1\text{e-}9, [0.08 \ 0.75], 4, 8,$
 | ‘L2Error’, 0.25, ‘initialiseNonSDF’, 100);
else
 | $[\phi, \mathcal{T}_{\text{NB}}] = \text{refineUpdate}(\phi_h^0, \mathcal{T}_{\text{NB}}, \mathcal{T}_{\text{NB}}, 1\text{e-}6, [0.08 \ 0.75], 4, 8,$
 | ‘SignedDistanceError’, 0.25, ‘initialise’, 100);
end
while $t_e^m < T$ **do**
 | Compute Advection Velocity Vector, \mathbf{b} ;
 | Compute time step, Δt_e ;
 | Evolve Level Set Interface by solving (5.6), $\phi_h^m(\mathbf{x})$;
 | Reinitialise level set function, $\phi_h^m(\mathbf{x})$;
 | Update narrow band, \mathcal{T}_{NB} ;
 | **if** *elements move from outside to inside narrow band* **then**
 | | Extrapolate the level set function onto those elements, $\phi_h^m(\mathbf{x})$;
 | **end**
 | $[\phi, \mathcal{T}_{\text{NB}}] = \text{refineUpdate}(\phi_h^m, \mathcal{T}_{\text{NB}}, \mathcal{T}_{\text{NB}}, 1\text{e-}6, [0.08 \ 0.75], 4, 8,$
 | ‘signedDistanceError’, 0.25, ‘reinitialise’, 100);
 | $t_e^{m+1} = t_e^m + \Delta t_e$;
end

Algorithm 4: *hp*-adaptive narrow banded level set evolution algorithm

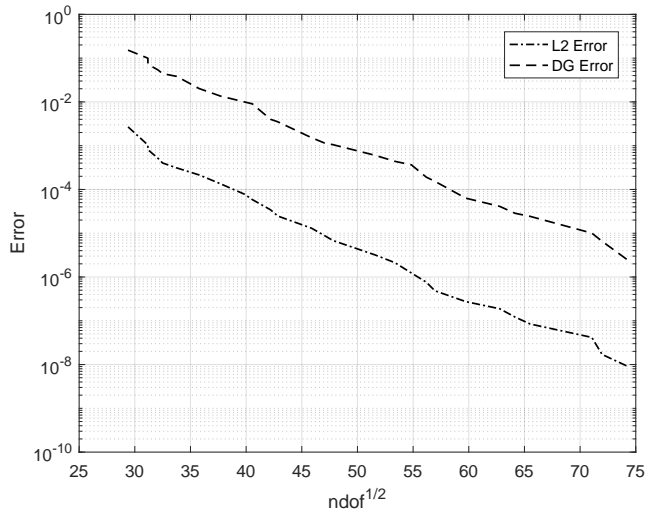


Figure 6.17: Error against the square root of the number of degrees of freedom (ndof), for the initialisation of the growing-shrinking-growing circular interface problem, on an hp -adaptively refined mesh.

refinement loop, Algorithm 3, to generate a new mesh upon which the projection error is less than a tolerance chosen in this case as $\text{errorTol}=10^{-9}\times\text{area}$.

The variation in error with mesh density for the initialisation of the circular interface on the adaptively refined mesh is shown in Figure 6.17. In this case, the singular region at the vertex of the conic level set function is removed from the mesh almost immediately, which allows the initialisation algorithm to converge quickly upon an appropriate initial mesh. Why this is the case here but wasn't during the circular interface reinitialisation problem for example (see Section 6.4.1), is that in this case the `computeAnalyticityTest` function immediately recognises where the solution is not smooth, (presumably as projection is less diffusive than reinitialisation when the singularity is aligned with a mesh node) which leads to the desired h -refinement and a narrowing of the narrow band which removes the singularity.

The error tolerance defining one of the stopping criteria is chosen stricter during the initialisation, as compared with during the evolution, for two reasons. First of all, the initial function will be chosen as a signed distance function, defined by an analytical function. This means it is simple and cheap to compute the initialisation and therefore it is useful to get as good a mesh as possible during this stage as this will promote accuracy during at least the initial iterations of the evolution. The second reason is that results from previous examples in this thesis suggest that the signed distance error estimate which will be used for refinement during the evolution tends to bound the L^2 error from above by at least an order of magnitude, and therefore a smaller tolerance is used to ensure that the tolerance on the signed distance error is also satisfied by the initialisation which is based on the L^2 error.

Figure 6.18 shows the first mesh upon which the converged solution to the initialisation of the GSG circular interface problem satisfies a stopping criterion of the mesh refinement loop, in this case satisfying the error tolerance. The mesh computed is reminiscent of the mesh computed earlier for the circular reinitialisation problem (see Section 6.4.1), in that h -refinement is focussed

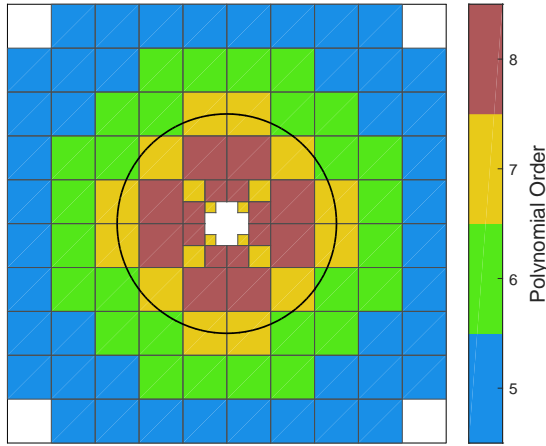


Figure 6.18: Final computed mesh configuration for the initialisation of the GSG circular interface problem on an hp -adaptively refined mesh where the colour of the element denotes the polynomial order of that element. The thick black line denotes the computed interface position.

at the origin where the solution is singular, and p -refinement is focussed elsewhere where the error is large but the solution is smooth. One point of note is that even for this simple initialisation problem, i.e. a circular interface defined by an analytical function which is known to be a signed distance function, relatively large polynomial bases are required to capture parts of the solution to the desired degree of accuracy. This shows the necessity of hp -adaptivity for problems of this type, and goes some way to explain some of the difficulties encountered earlier, for example, with the reinitialisation of the ellipse (see Section 6.4.4) and the smooth star interfaces (see Section 6.4.3), where an upper limit on the polynomial order, $p = 8$, was enforced.

6.6.1.2 Evolution

Once the initialisation has been completed the evolution can begin. The evolution for the GSG circular interface problem will proceed, driven by an advection velocity defined in Equation (5.20), which can be described as growth at a constant rate in the period $t_e = (0, 0.15]$, shrinking at the same constant rate in the period, $t_e = (0.15, 0.45]$, followed by a period of constant growth again in the period, $t_e = (0.45, 0.6)$, such that the interface should arrive finally at its initial position at time, $t_e = T = 0.6$. The function describing the analytical solution over the domain for all time is defined in Equation (5.21). If at any point during the evolution the signed distance error grows such that $E_{SD} > 10^{-6} \times \text{area}$, this will trigger the refine-reinitialisation loop (Algorithm 3), which will continue in a while loop until either the tolerance on the error is satisfied, or the mesh becomes maximally refined.

Figure 6.19 shows how the error varies over pseudotime, during the evolution for the GSG circular interface problem on an hp -adaptively refined mesh. As there is no h -refinement away from the origin, the mesh is fairly coarse throughout the evolution. This means that the area of the narrow band remains constant for the majority of the evolution. The two instances where this is not true are when the interface expands to its largest around $t_e = 0.15$, and when the interface shrinks to its smallest around $t_e = 0.45$, whereby the narrow band tracks the evolving interface and changes shape (this can be seen in Figure 6.20). During the first of these two

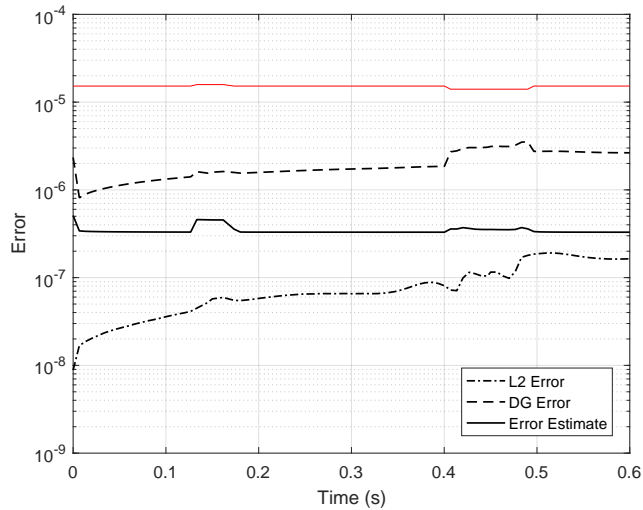
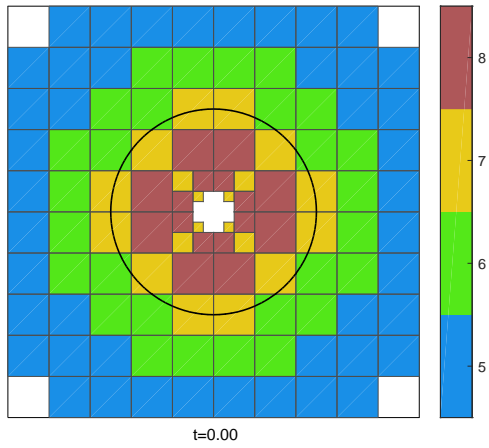


Figure 6.19: Error over pseudotime for the GSG circular interface problem on an hp -adaptively refined mesh. The red line denotes the error tolerance defining one of the stopping criteria, which can change as the area of narrow band changes.

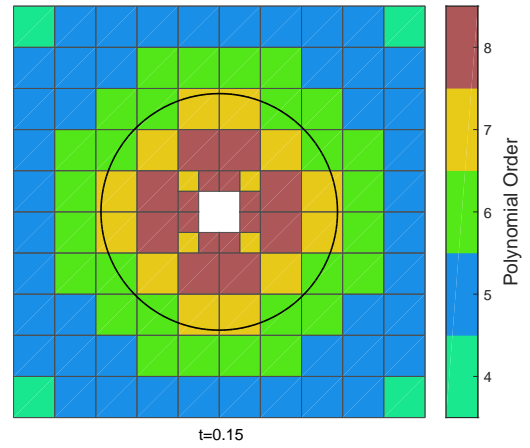
periods, the size of the narrow band grows and with it the error; this is most notably apparent in the signed distance error, but can also be seen in the L^2 and DG norms too. Once the interface begins to shrink again, the area of the interface returns to its original size, and those elements which moved from outside to inside the narrow band once again leave the narrow band and with it the additional error associated with them. During the second change in the shape of the narrow band the area of the narrow band decreases but the elements which move from outside to inside the narrow band in the same instant, are closer to the singularity and have a much higher curvature to capture and thus the error increases once again. Once the interface grows again, these elements eventually leave the narrow band and the additional error associated with them also leaves. As would be expected the cumulative error at each time step increases over time which is reflected in the increasing error in the L^2 and DG norms over time, with the oscillations explained by the changing shape of the narrow band as noted above.

The red line in Figure 6.19, demonstrates the criterion defining whether refinement is necessary. In this example the signed distance error never approaches this line and as such for all time, no further refinement needs to occur. This is to be expected as this is such a simple problem; the level set function evolves as a rigid body in space, and thus the signed distance error is constant over time varying only due to the change in the size of the narrow band.

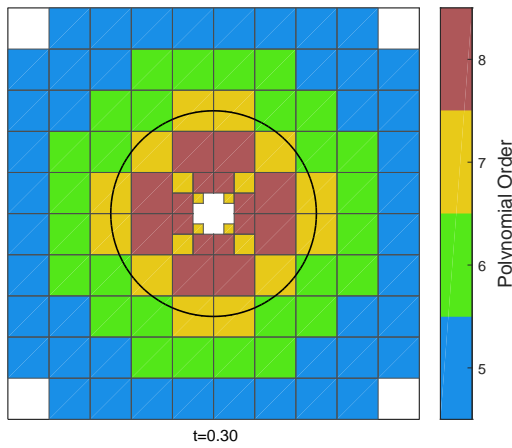
Comparing the results for the hp -adaptive mesh with the fixed mesh for the same problem in Section 5.4.2.1, it can be observed that the results are similar both in terms of how the error varies over time and the absolute value of the errors. The main difference is that as a coarser mesh of higher-order elements are used here, the evolution in the shape of the narrow band is much less severe, which results in less variation in the error over time. Also using the hp -adaptive mesh for the problem is more efficient both in terms of the number of degrees of freedom required, that is during the evolution, an average of 12769 degrees of freedom are required for the fixed mesh (where the average is computed as the total number of degrees of freedom in



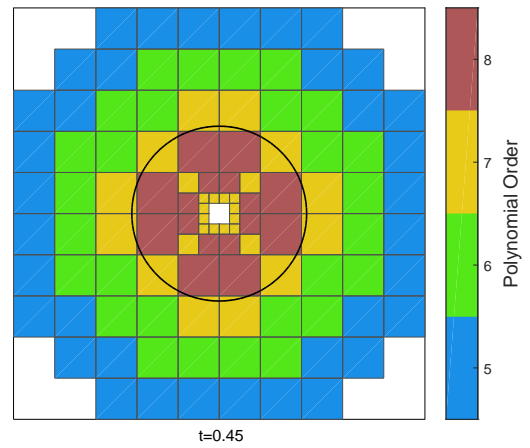
(a) Mesh and interface position at $t_e = 0.00$.



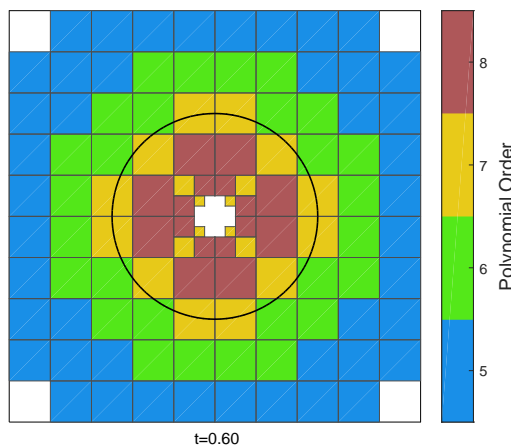
(b) Mesh and interface position at $t_e = 0.15$.



(c) Mesh and interface position at $t_e = 0.30$.



(d) Mesh and interface position at $t_e = 0.45$.



(e) Mesh and interface position at $t_e = 0.60$.

Figure 6.20: Configuration of mesh and interface position over time for the GSG circular interface evolution problem, on an hp -adaptively refined mesh. The thick black line denotes the computed interface position.

the narrow band at each time step divided by the total number of time steps), whereas just an average of 5511 are required for the adaptive mesh, as well as the number of time steps required, which is 219 for the fixed mesh versus just 89 for the adaptive mesh. The results for the fixed mesh mentioned here is that mesh where $h = 0.05$ and $p = 3$.

Figure 6.20 shows some snapshots of the evolving interface, as well as the evolving adaptive mesh over time for the GSG circular interface problem. As mentioned earlier no additional refinements are required for this example problem after the initialisation of the mesh and as such all that is required during the evolution is for some of the elements outside of the narrow band move inside, and vice versa, to maintain the correct narrow band width based on the position of the evolving interface. Although not plotted, the errors are sufficiently small such that there would be no observable difference between the computed and analytical position of the level set interface.

6.6.2 Growing-shrinking-growing problem with pure h -refinement

It is interesting to see how the proposed hp -adaptive version of the level set method compares to a similar method which is only h -adaptive. To do this the growing-shrinking-growing (GSG) circular interface example problem will be repeated, however this time, the hp -adaptivity routine of Algorithm 3 will be modified such that only h -flags are considered which means that no criterion is required to decide whether it is preferable to refine in h or p , and likewise flags can no longer change from h to p once maximum h -refinement had been achieved for a given element.

6.6.2.1 Initialisation

Once again, an initial level set function which is the signed distance function to a circular interface of radius $r = 1$, that is as defined in Equation (5.19), is L^2 projected onto the domain $\Omega = (-2, 2)^2$. The initial mesh consists of square elements of size $h = 0.4$ and of polynomial order $p = 2$, which after the initial projection is narrow banded and passed into the modified refinement loop, to generate a new mesh upon which the projection error is less than a tolerance chosen in this case as `errorTol=10-8×area`. There are two reasons for the slight change in error tolerance here as compared with the previous example which uses the hp -adaptive method. Firstly, when initialising the mesh for the h -adaptive problem, in order to satisfy the stricter error tolerance the method required larger upper limits on h -refinement, i.e. the tolerance couldn't be satisfied, which felt unfair for comparative purposes. And secondly, choosing the error tolerance as stated above leads to comparable absolute errors in L^2 norm for both problems, allowing for an easier comparison in the efficiency of each of the methods.

The variation in error with mesh density for the initialisation of the circular interface on the h -adaptively refined mesh is shown in Figure 6.21, along with the same error data from the previous example using the hp -adaptivity routine as presented in Figure 6.17. It can be seen once again, that immediately the singularity at the peak of the cone describing the circular interface is removed from the domain by refining in h and updating the narrow band. This happens in the same way for both the h and hp methods, which can be seen as the error curves for both methods in Figure 6.21 initially overlap. The error curves for the hp -adaptive method then continue to descend with approximately the same gradient owing to the fact that the problem is

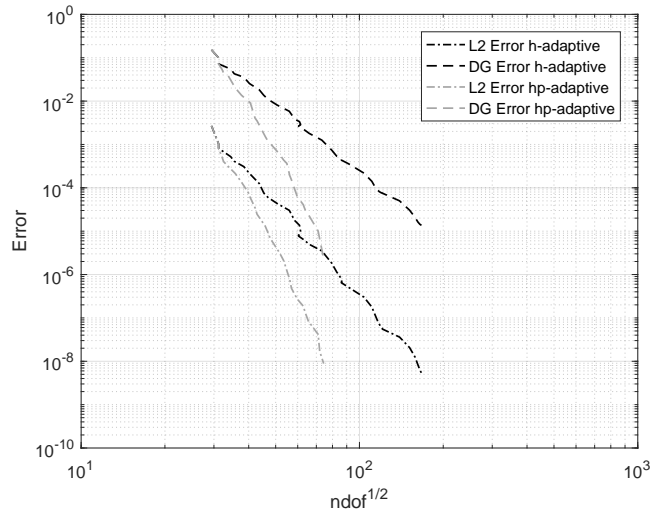


Figure 6.21: Comparison of errors against the square root of the number of degrees of freedom (ndof), for the initialisation of the growing-shrinking-growing circular interface problem, on an h -adaptively and hp -adaptively refined mesh.

then smooth enough to take advantage of the increased efficiency of p -refinement, whereas the error curves for the purely h -adaptive case then slow to a gentler gradient which is nevertheless maintained until the error threshold is satisfied. In order to satisfy the tolerance on error for the L^2 projection of the explicit function (5.19) onto the narrow banded domain, requires 27900 degrees of freedom, which can be compared with the 5532 for the hp -adaptively refined mesh.

The difference between h and hp adaptivity becomes obvious when comparing the initial meshes produced in each case for this problem as can be seen in Figure 6.22. Many more elements of significantly smaller size are required to satisfy the error threshold, resulting in a much narrower narrow band. Examination of Figure 6.22 also provides evidence for some of the comments made in Sections 6.4.3 and 6.4.4. That is, firstly, along the $x = 0$ and $y = 0$ planes where the relative position of the interface and the grid is well aligned, less refinement is required to achieve the desired degree of accuracy. And also again one can notice the inherent difficulty in modelling even simple shapes to high levels of accuracy using the level set method on the chosen set of finite element spaces as evidenced by the high levels of refinement required just to project the initial function onto the domain.

6.6.2.2 Evolution

Figure 6.23 presents the error over pseudotime for the GSG problem on the h -adaptively refined mesh, as well as the same error data from the previous example problem on the hp -adaptively refined mesh as presented in Figure 6.20. It can be seen immediately that one of the main differences between the two methods is that due to the much smaller elements required to satisfy the error tolerance, the time step required as stated in (5.11) leads to many more iterations for the same time period, 1597 iterations for the h -adaptive method versus 89 for the hp -adaptive. This also leads to an increased requirement for refinement and reinitialisation throughout the evolution, which can be seen where the value of the error estimate approaches the refinement

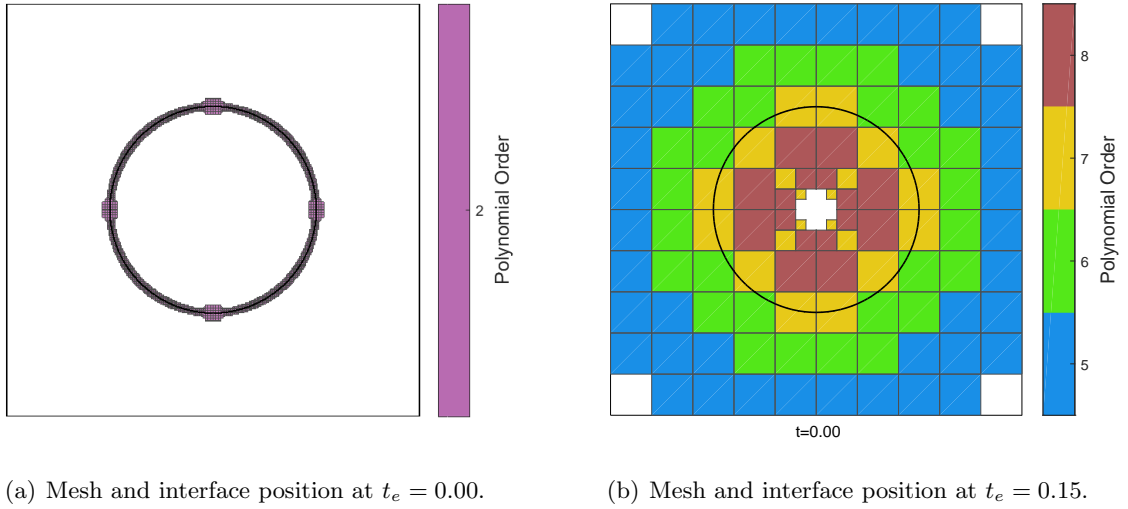


Figure 6.22: Configuration of mesh and interface position over time for the GSG circular interface evolution problem, on an h -adaptively and hp -adaptively refined mesh. The thick black line, in both cases, denotes the computed interface position.

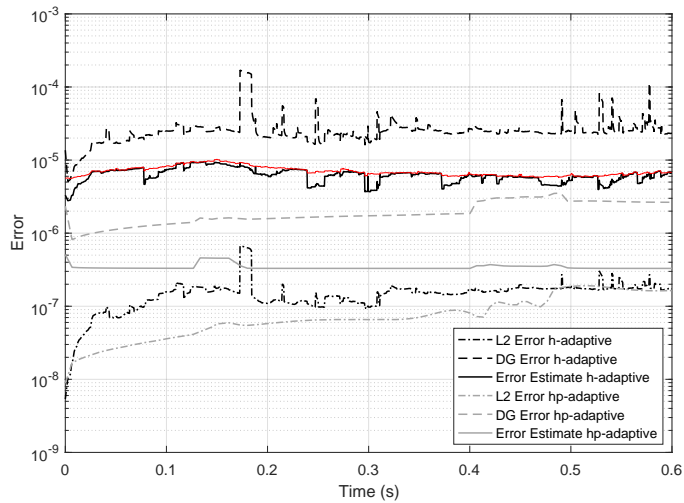
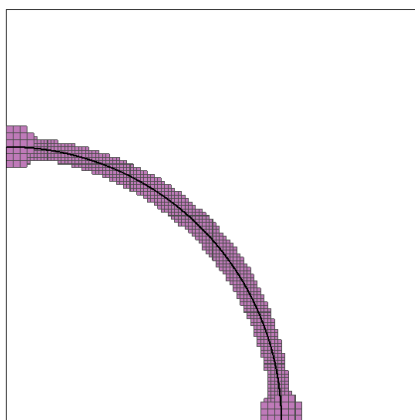
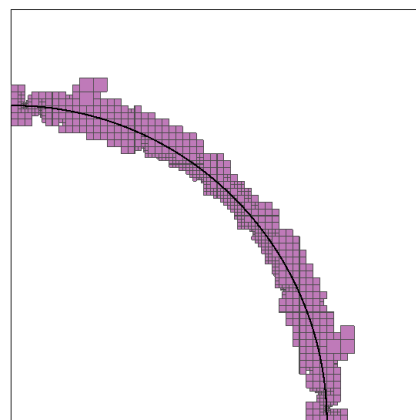


Figure 6.23: Comparison of errors over pseudotime for the GSG circular interface problem on an h -adaptively and hp -adaptively refined mesh. The red line denotes the error tolerance defining one of the stopping criteria, which can change as the area of narrow band changes.



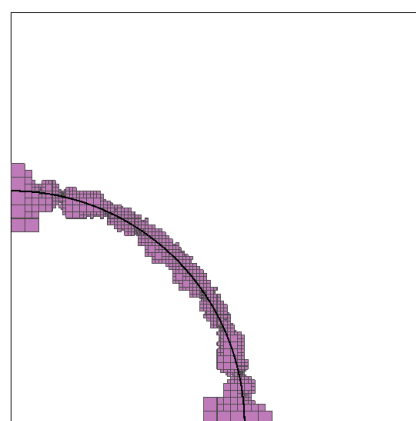
(a) Mesh and interface position at $t_e = 0.00$.



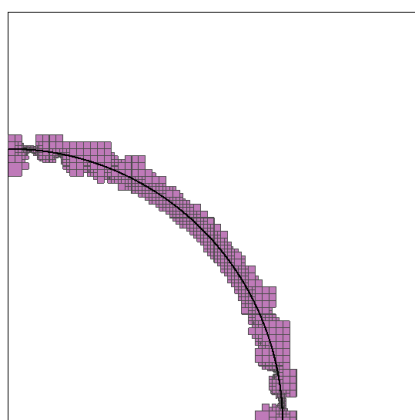
(b) Mesh and interface position at $t_e = 0.15$.



(c) Mesh and interface position at $t_e = 0.30$.



(d) Mesh and interface position at $t_e = 0.45$.



(e) Mesh and interface position at $t_e = 0.60$.

Figure 6.24: Configuration of mesh and interface position over time for the GSG circular interface evolution problem, on an h -adaptively refined mesh. The thick black line denotes the computed interface position. Due to the rotational symmetry of the problem only displayed is the region $(0, 1.5)^2$, to improve visibility for such small h .

criterion denoted by the red line in Figure 6.23. Furthermore, it can be noted that the average number of degrees of freedom throughout the evolution is 29692 for the h -adaptive method which can be compared to the 5511 for the hp -adaptive method. Thus, the comparison performed here ultimately shows what would be expected in that pure h -refinement is less efficient both in terms of number of degrees of freedom and real time taken than the hp -adaptive method. Another point of note is that it can be seen in Figure 6.23 that the L^2 error in the solution is similar for the duration of the evolution for both of the adaptive strategies, whereas the error in the gradient measured using both the error estimate and DG norm is approximately an order of magnitude smaller for the hp -adaptive strategy over the h -adaptive strategy.

The mesh and interface configuration at each of a number of time steps during the h -adaptive study is shown in Figure 6.24 (which shows the mesh and interface only in the region $(0, 1.5)^2$ so as to improve the visibility of the smaller elements; the other 3 quadrants are implied by symmetry). It can be seen that the general shape of the mesh remains similar to the initialisation throughout the evolution. When the circular interface expands the mesh can become slightly coarser as the curvature decreases and as the interface shrinks vice versa. Again although not plotted there would be no observable difference between the computed and analytical position of the level set interface owing to the small error levels maintained throughout the evolution.

6.6.3 hp -convergence study: translating circle

6.6.3.1 Initialisation

The next example problem is a repeat of the translating circle problem introduced in Section 5.4.2.2. In this case, an initial level set function which is a signed distance function to a circular interface of radius, $r = 0.15$, centred at $\mathbf{x} = (-0.25, 0)$, that is as defined in Equation (5.22), is L^2 projected onto the domain $\Omega = (-0.5, 0.5) \times (-0.2, 0.2)$. The initial mesh consists of square elements of size, $h = 0.04$, and of polynomial order, $p = 2$, which after the initial projection is narrow banded and passed into the refinement loop, Algorithm 3, to generate a new mesh upon which the projection error is less than a tolerance chosen in this case as `errorTol`= $10^{-9} \times \text{area}$.

The variation in error with mesh density for the initialisation of the translating circular interface problem is shown in Figure 6.25. On the initial Cartesian mesh, the singularity falls inside an element. The L^2 projection of the singularity onto an element with a polynomial basis, generates a smooth, yet poor approximation of the chosen signed distance function, which by virtue of its smoothness is adaptively refined in p initially. Once the upper limit on p is reached the flags switch to h and the singularity appropriately removed. As mentioned prior this is not an uncommon mode of behaviour for this piece of the proposed method, see Section 6.4.1. This explains the initially limited convergence in Figure 6.25. The mesh generated by this initialisation routine, see Figure 6.26, is similar to the previous example problems where a circular interface is represented by an adaptive mesh.

6.6.3.2 Evolution

The evolution for the translating circular interface problem will be driven by an advection velocity defined in Equation (5.23), which can be described as a translation of the circular interface in the positive x -direction at a constant rate. This evolution will occur over the

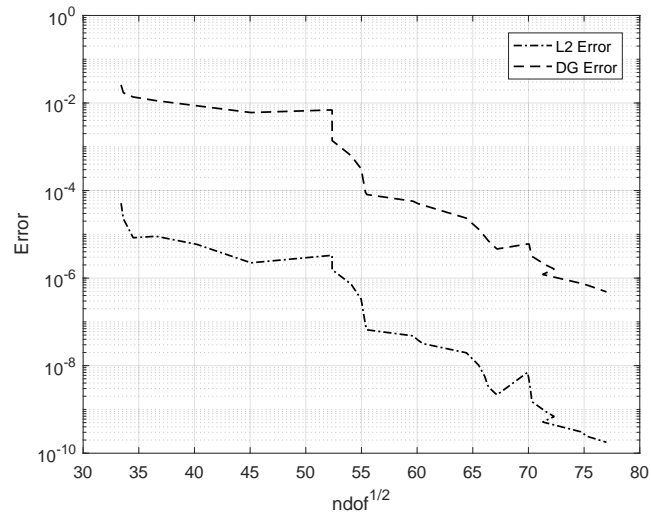


Figure 6.25: Error against the square root of the number of degrees of freedom (ndof), for the initialisation of the translating circular interface problem, on an hp -adaptively refined mesh.

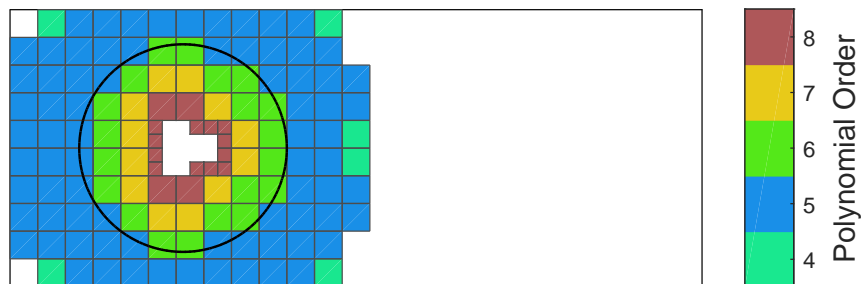


Figure 6.26: Final computed mesh configuration for the initialisation of the translating circular interface problem on an hp -adaptively refined mesh where the colour of the element denotes the polynomial order of that element. The thick black line denotes the computed interface position.

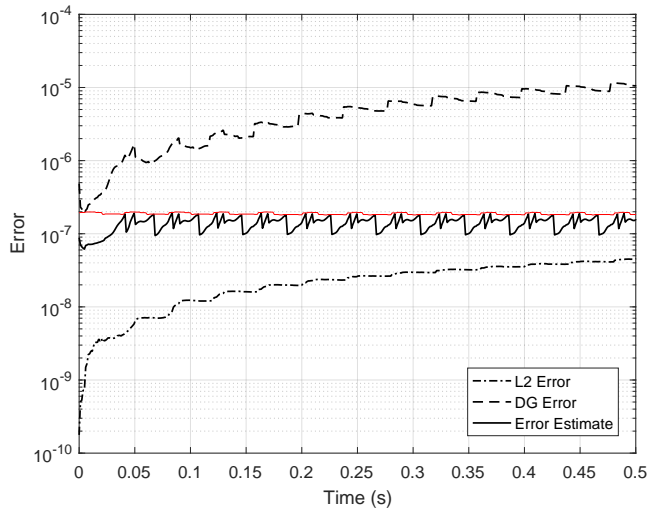
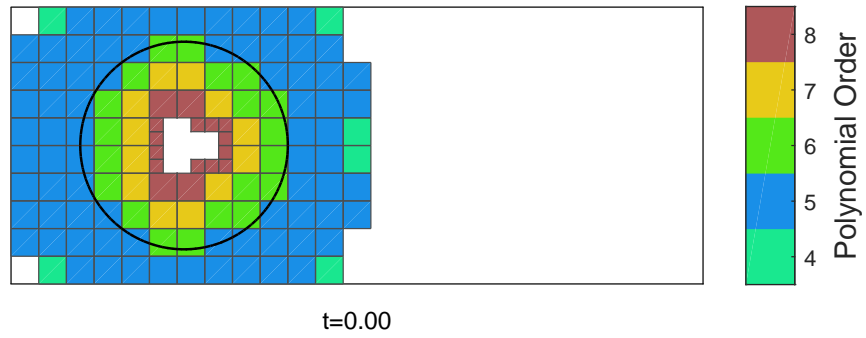


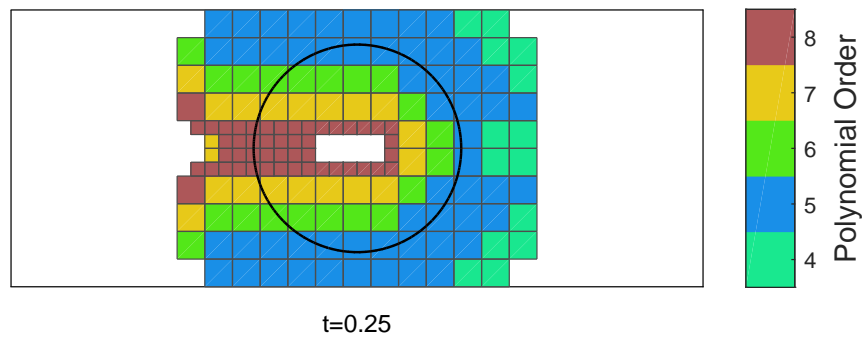
Figure 6.27: Error over pseudotime for the translating circular interface problem on an hp -adaptively refined mesh. The red line denotes the error tolerance defining one of the stopping criteria, which can change as the area of narrow band changes.

interval $t_e = (0, 0.5)$. The explicit function describing the analytical solution to the translating circular interface problem is that stated in Equation (5.24). If at any point during the evolution the signed distance error grows such that $E_{SD} > 10^{-6} \times \text{area}$, this will trigger the refine-reinitialisation loop (Algorithm 3), which will continue in a while loop until either the tolerance on the error is satisfied, or the mesh becomes maximally refined.

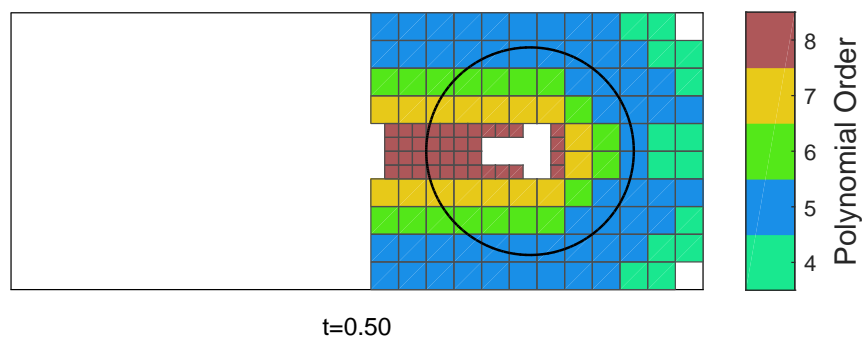
Figure 6.27 shows how the error varies over pseudotime for the duration of the translating circle problem on an hp -adaptively refined mesh. The error in the L^2 and DG norms, grows over time as would be expected, due to the error at each previous iteration compounding at each time step. There are oscillations in these error norms, which again reflect the evolving nature of the domain, as the narrow band tracks the evolving interface. This was also noticed when the same problem was computed on a fixed narrow banded mesh. There is a pattern in these oscillations as the time step is constant throughout the evolution (as a result of the size of the most refined element remaining constant) which means there is a repeated pattern in the relative position of the interface and the grid. As the narrow band evolves, elements which were outside the narrow band move inside the narrow band. Those elements entering from the right hand side of the mesh tend to be under-refined for the desired level of error, as the refinement strategy ensures elements outside of the narrow band have minimal refinement (whilst still satisfying the bounds on local variation) to reduce wasted memory. When these elements enter the narrow band, the signed distance error measure spikes which triggers the refine-reinitialisation routine which ensures these elements are sufficiently refined to capture the level set function to the desired tolerance on error. This happens in a patterned fashion too for the same reason. Comparing the results for the translating circle problem on an adaptive mesh, with the results for a fixed mesh (see Section 5.4.2.2), it can once again be seen that fewer degrees of freedom are required for the adaptive mesh, where an average of 7285 across the iterations are required for the desired level of accuracy, versus an average of 9432 degrees of freedom for the fixed mesh, which is true



(a) Mesh and interface position at $t_e = 0.00$.



(b) Mesh and interface position at $t_e = 0.25$.



(c) Mesh and interface position at $t_e = 0.50$.

Figure 6.28: Configuration of mesh and interface position over time for the translating circular interface evolution problem, on an hp -adaptively refined mesh. The thick black line denotes the computed interface position.

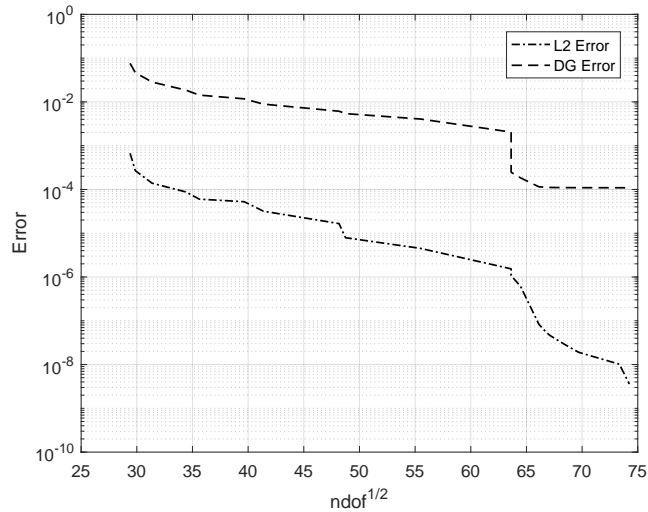


Figure 6.29: Error against the square root of the number of degrees of freedom (ndof), for the initialisation of the shearing circular interface problem, on an hp -adaptively refined mesh.

despite the errors on the adaptive mesh being almost an order of magnitude smaller. Similarly as coarser elements can be used, the time step can be larger, and the number of iterations likewise smaller for the adaptive mesh where 426 iterations of the evolution loop are required, compared to the 901 on the fixed mesh where $h = 0.01$ and $p = 3$.

Figure 6.28, shows a number of snapshots of the mesh and the interface during the evolution for the translating circular interface problem. Again although not plotted, the errors are sufficiently small such that there would be no observable difference between the computed and analytical position of the level set interface.

6.6.4 hp -convergence study: shearing circle

6.6.4.1 Initialisation

The next example is the shearing circular interface problem introduced in Section 5.4.2.3. An initial level set function which is the signed distance function to a circular interface of radius, $r = 0.5$, centred at the origin, that is as described in Equation (5.25), is L^2 projected onto the domain $\Omega = (-1, 1)^2$. The initial mesh consists of square elements of size $h = 0.2$ and $p = 2$, which after the initial projection is narrow banded and passed into the refinement loop, to generate a mesh upon which the L^2 error in the projection satisfies the tolerance, $\text{errorTol} = 10^{-9} \times \text{area}$.

The variation in error with mesh density associated with the initialisation of the circular interface for the shearing circle problem can be seen in Figure 6.29. Once again this is the initialisation of a circular interface, driven by the L^2 error in the solution. In this case the analyticity estimate indicates that the solution in the elements near where the singularity falls is sufficiently smooth to be refined in p . Once the upper limit on p refinement is reached, the flags switch to h -flags should further refinement be required in the region, at which point the singularities are removed and the refinement loop can converge quickly upon a mesh which can represent the level set function with the desired level of accuracy. The stagnation in the DG

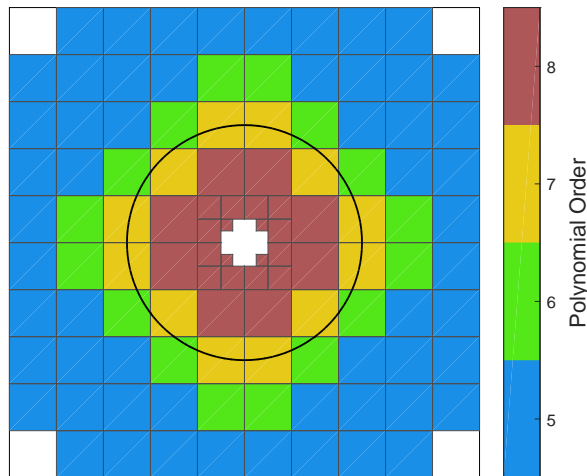


Figure 6.30: Final computed mesh configuration for the initialisation of the shearing circular interface problem on an hp -adaptively refined mesh where the colour of the element denotes the polynomial order of that element. The thick black line denotes the computed interface position.

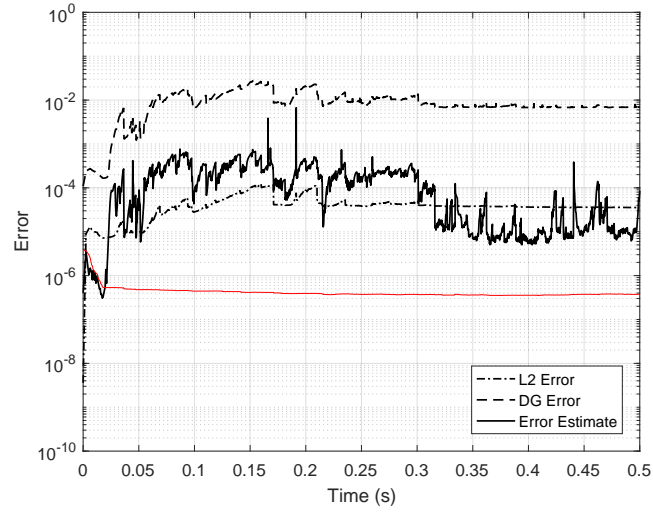
norm is a numerical artefact from the truncation error when computing the analytical gradient using a finite difference method.

Figure 6.30 shows the first mesh upon which the converged solution to the initialisation of the shearing circular interface problem satisfies a stopping criterion of the mesh refinement loop. The computed mesh is similar to all of the meshes upon which circular interfaces have been projected in the example problems on adaptive meshes thus far.

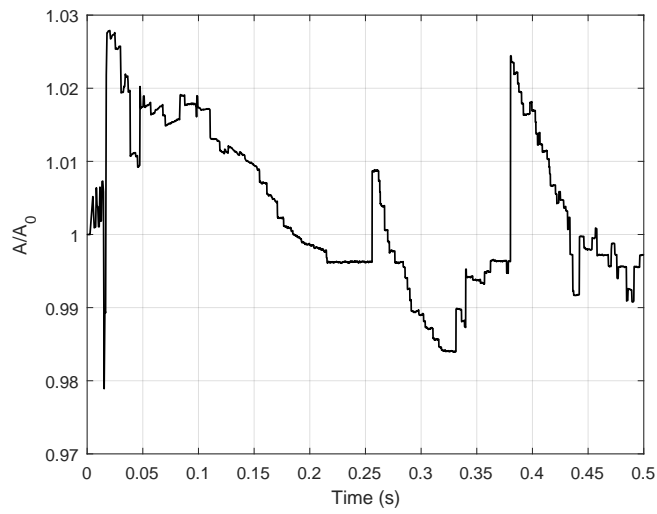
6.6.4.2 Evolution

The evolution can then proceed, driven by the advection velocity defined in Equation (5.26), which will, in the interval $t_e = (0, 0.5)$, cause the initially circular interface to shear, stretch and rotate. The function describing the analytical solution to the shearing circle problem over time, is given in Equation (5.28). If at any point during the evolution the signed distance error grows such that $E_{SD} > 10^{-6} \times \text{area}$, this will trigger the refine-reinitialisation loop (Algorithm 3), which will continue in a while loop until either the tolerance on the error is satisfied, or the mesh becomes maximally refined.

Figure 6.31 presents the variation in error over pseudotime for the shearing circle problem on an hp -adaptively refined mesh. For this problem the mesh is initialised so as to accurately represent a circular interface, which is a problem with which the proposed method has been successful throughout this chapter. As the evolution begins the circle begins to shear and become more elliptic. As seen in the reinitialisation example problems in Section 6.4, there is some relationship between the either the curvature or the orientation of the interface relative to the grid, and the ability to solve the reinitialisation in a timely fashion without using greater levels of refinement than are to be used here. This example illuminates just how slight this additional complexity needs to be before $p = 8$ elements and a maximum of 100 Picard iterations of the reinitialisation, are no longer sufficient to represent the level set function with the desired



(a) Error against pseudotime



(b) Area ratio against pseudotime

Figure 6.31: Error and area ratio over pseudotime for the shearing circular interface problem on an hp -adaptively refined mesh. The red line denotes the error tolerance defining one of the stopping criteria, as this varies with the area of narrow band.

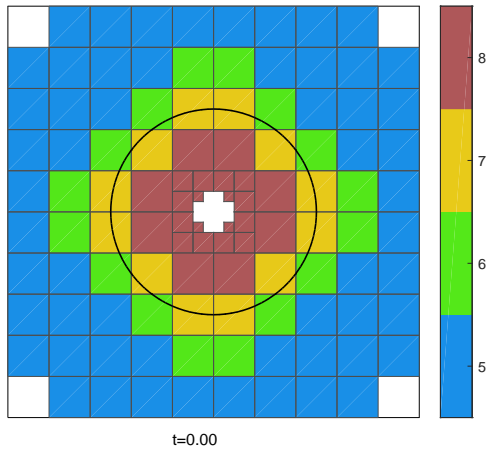
level of accuracy. Figure 6.32 shows the interface and mesh configurations at a number of points in the interval $t_e = (0, 0.01)$ and the mesh required to satisfy the error tolerance. It can be seen that after very few time steps, almost the entire mesh consists of $p = 8$ elements.

One point of note with this example, however, is that during the period $t_e = (0, 0.025)$, the shape of the interface is more complex than the initial circle but not so complex that the limits on refinement and number of Picard iterations are insufficient to satisfy the tolerance on the signed distance error. During this period the refinement strategy also has a positive effect on the error in the L^2 and DG norms. These results can be seen in Figure 6.33. This is promising as it implies that the proposed method is capable of generating a solution with any desired level of accuracy for any problem given enough computing power.

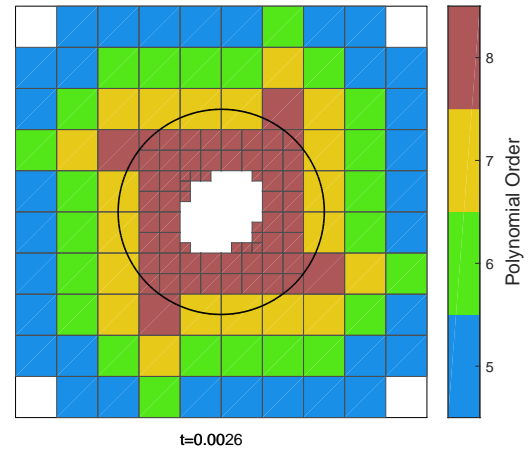
Also shown in Figure 6.31, is that beyond $t_e = 0.025$ the error grows as the upper limits on refinement are reached and the error reflects the best possible solution given these limits as opposed to satisfying the tolerance on the error. In the interval $t_e = (0.025, 0.3)$ the change in the shape and the orientation of the shape manifests as an error which is in general growing but oscillates considerably, as reinitialisation continues at each step to attempt to decrease the error and the shape becomes increasingly difficult to capture on a given mesh. Beyond around $t_e = 0.3$, the signed distance error decreases but continues to oscillate, the error in the L^2 and DG norms remains constant, which is something not seen in previous examples. The decrease in error at this stage does seem to align with an observation in Section 6.4.4, that being that certain orientations of the interface relative to the background grid are more difficult to capture than others, even if the curvature is larger.

Compared with the solution of the same problem on a fixed mesh (again see Section 5.4.2.3), the error is reduced in both the L^2 and signed distance error measures. The error seems to have increased in the DG norm, however, again this is likely to do with truncation errors in computing the known gradient using a finite difference method, which are larger due to the much large number of interpolation points at which the error is computed. This increase in accuracy does appear to have come at cost however, as the number of degrees of freedom in the system for the fixed mesh grows from 12800-15680, as the shape grows for the fixed mesh, compared to 29431-147231 for the adaptive mesh. Also, the smaller elements and much higher polynomial orders used in the simulation on the adaptive mesh causes a significant reduction in the time step increasing the number of iterations required for the simulation from 110 for the fixed mesh, to 2806 for the adaptive mesh. Given what has been observed across the numerical examples in this chapter, increasing the limit of allowed p -refinement would likely result in a more economical use of degrees of freedom as well as reducing the required time step, and improving the accuracy, especially given that other than the singularity near to the origin this problem is smooth everywhere.

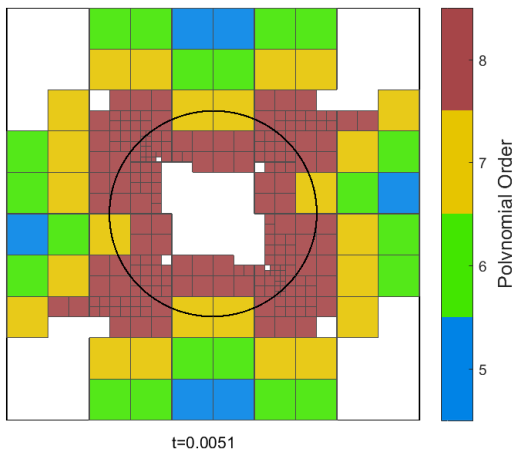
Also shown in Figure 6.31 is the area ratio over time. Area ratio denotes the relative change in the area of the shape, which given that the advection velocity is divergence free should remain constant throughout the evolution. Despite a larger variance in the area ratio over time, compared with the same problem computed on a fixed mesh (see Section 5.4.2.3), the area ratio here tends to oscillate around unity which is an improvement over the fixed mesh where the area



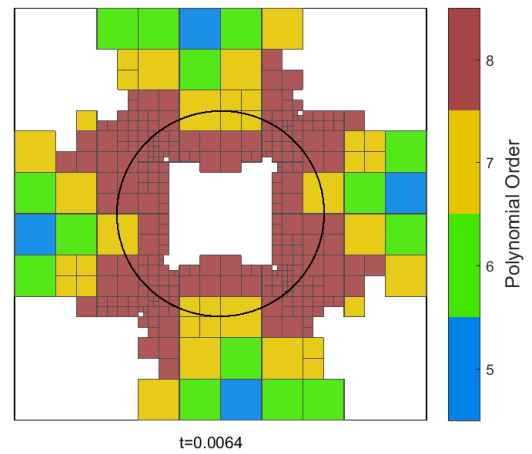
(a) Mesh and interface position at $t_e = 0.0000$.



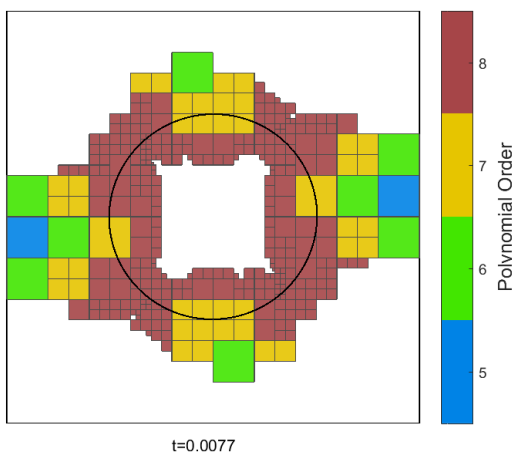
(b) Mesh and interface position at $t_e = 0.0026$.



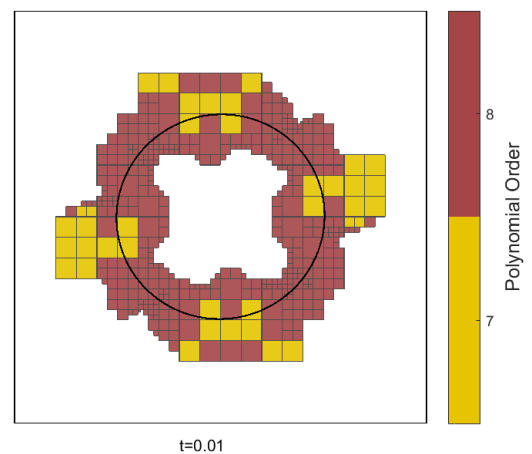
(c) Mesh and interface position at $t_e = 0.0051$.



(d) Mesh and interface position at $t_e = 0.0064$.



(e) Mesh and interface position at $t_e = 0.0077$.



(f) Mesh and interface position at $t_e = 0.0100$.

Figure 6.32: Configuration of mesh and interface position over time for the shearing circular interface evolution problem, on an hp -adaptively refined mesh, in the interval $t_e = (0, 0.01)$. The thick black line denotes the position of the discrete interface.

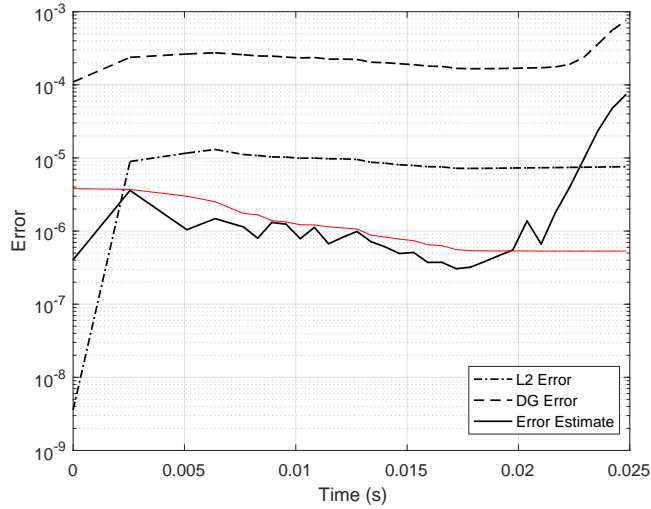


Figure 6.33: Error over time, for the shearing circular interface problem, on an hp -adaptively refined mesh in the time period $t_e = (0, 0.025)$. The red line denotes the error tolerance defining one of the stopping criteria, which can change as the area of narrow band changes.

ratio began to grow over time with no indication that this would decrease if the simulation was to continue.

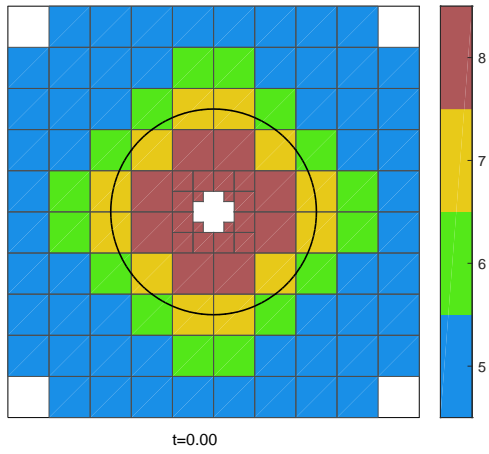
Figure 6.34 shows snapshots of the interface and the mesh over time, for the shearing circle interface problem. What can be seen is that the initial circular interface is captured well using very few, high order elements. Higher levels of refinement can be seen as the shape of the interface becomes more and more difficult to resolve well given the limits on the refinement, reflecting what was seen earlier for the elliptical interface reinitialisation problem (see Section 6.4.4).

6.6.5 hp -convergence study: merging circles

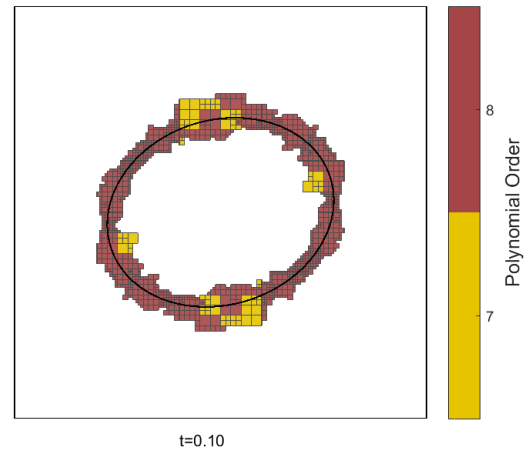
6.6.5.1 Initialisation

The final example to be presented in this chapter is similar to the merging circular interface example from Section 5.4.2.4. The same initial signed distance function from that problem, which takes the form of two circles of radius, $r_1 = r_2 = 0.5$, centred at $\mathbf{x} = (0.55, 0)$ and $\mathbf{x} = (-0.55, 0)$, as stated in Equation (5.30), is L^2 projected onto the domain, $\Omega = (-1.5, 1.5)^2$. The initial mesh consists of square elements of size, $h = 0.3$, and of polynomial order, $p = 2$, which after the initial projection is narrow banded and passed into the refinement-projection loop, to generate a mesh upon which the L^2 error in the projection satisfies the tolerance, $\text{errorTol} = 10^{-9} \times \text{area}$.

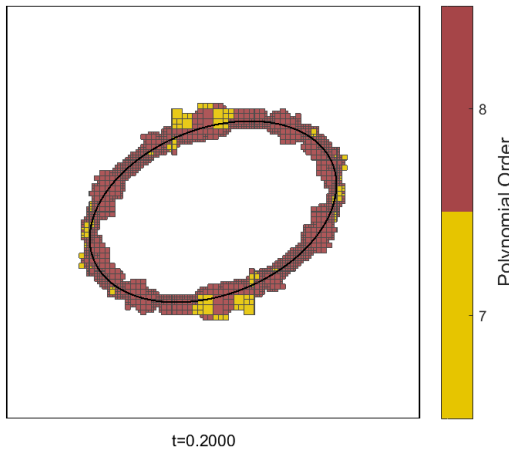
The variation in error with mesh density for the initialisation of the merging circles example problem is shown in Figure 6.35. The problem converges exponentially towards the desired mesh by immediately h -refining near the singularities at the centre of each of the two circles, after which the problem is considered smooth enough everywhere to refine everywhere in p until the error tolerance on the initial projection is satisfied. Interestingly, there is another singular region



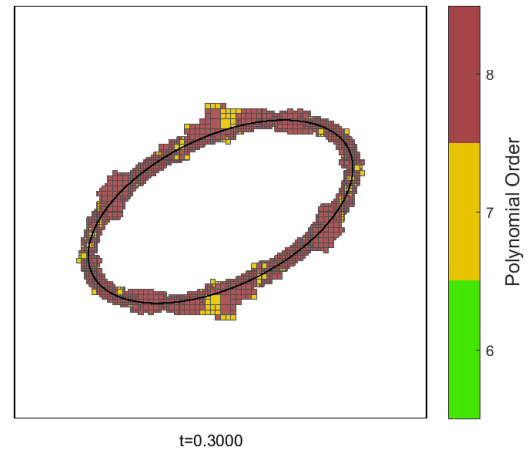
(a) Mesh and interface position at $t_e = 0.00$.



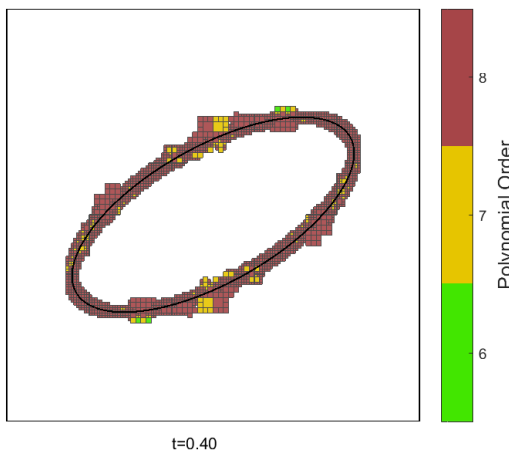
(b) Mesh and interface position at $t_e = 0.10$.



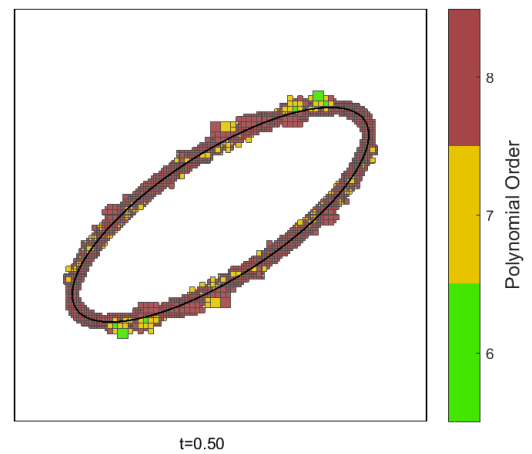
(c) Mesh and interface position at $t_e = 0.20$.



(d) Mesh and interface position at $t_e = 0.30$.



(e) Mesh and interface position at $t_e = 0.40$.



(f) Mesh and interface position at $t_e = 0.50$.

Figure 6.34: Configuration of mesh and interface position over time for the shearing circular interface evolution problem, on an hp -adaptively refined mesh. The thick black line denotes the position of the discrete interface.

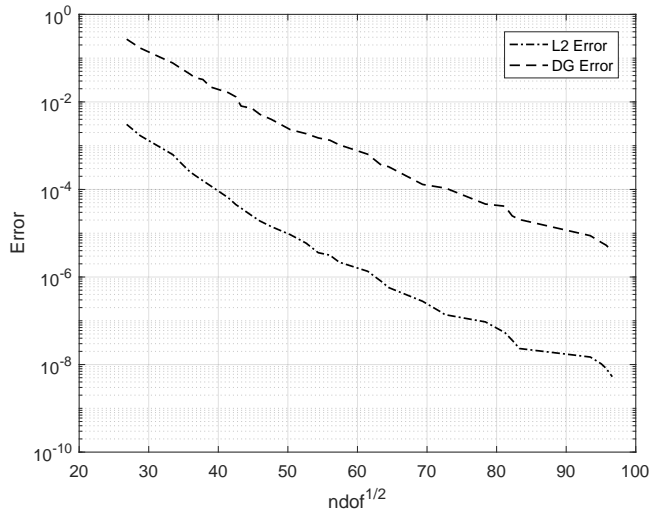


Figure 6.35: Error against the square root of the number of degrees of freedom (ndof), for the initialisation of the merging circular interfaces problem, on an hp -adaptively refined mesh.

in this example along the $x = 0$ plane, however this falls exactly along the edges of elements and thus the curvature of the solution in the elements either side of this singularity is relatively slight and thus is captured well by the initial mesh. The resultant mesh can be seen in Figure 6.36.

6.6.5.2 Evolution

The advection velocity for the merging circles example in this case can be stated as follows

$$b = \begin{cases} -1, & t_e < 0.35, \\ 1, & t_e \geq 0.35, \end{cases} \quad (6.19)$$

which will drive evolution in the interval $t_e = (0, 0.7)$. This advection velocity will cause the initial two circular interfaces to grow at a constant equal rate, before meeting at the origin at $t_e = 0.05$. The difference between this example and the example in Section 5.4.2.4, is that the two merging circles will then continue to grow until $t_e = 0.35$. This decision was made to ensure that the elements at the origin get narrow banded out of the domain, so as to test the effect of the extrapolation at this singular region once the interface begins to shrink again. Beyond $t_e = 0.35$ the advection velocity will cause the new shape created by the merging of the circles to shrink. Again as no new holes can be created when evolving a function using the level set method (this is not strictly true as interfaces can split as a result of the finite nature of the proposed methodology) one should expect that the new shape should take the form of a dumbbell, the exact shape of which is a function of the discretisation. For this reason there is not an explicit function which can be used to compute the error against, and thus only the signed distance error will be presented for this example. During the evolution of the level set function the threshold on the signed distance error which will trigger the refine-reinitialise loop can be stated $E_{SD} > 10^{-6} \times \text{area}$.

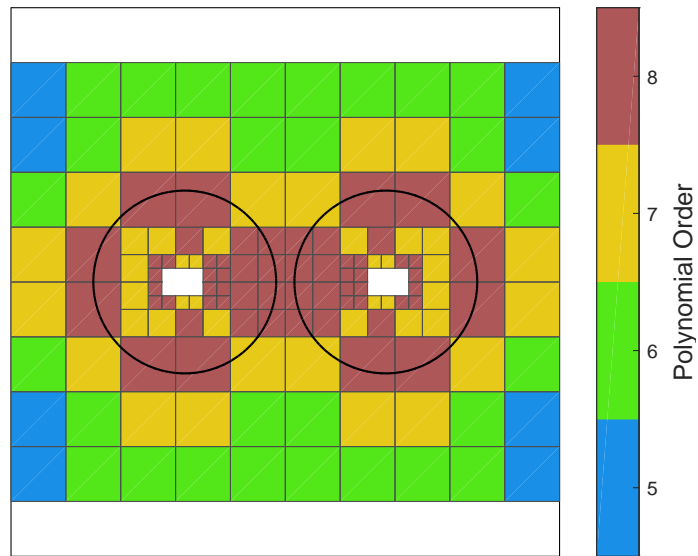


Figure 6.36: Final computed mesh configuration for the initialisation of the merging circular interfaces problem on an hp -adaptively refined mesh where the colour of the element denotes the polynomial order of that element. The thick black line denotes the computed interface position.

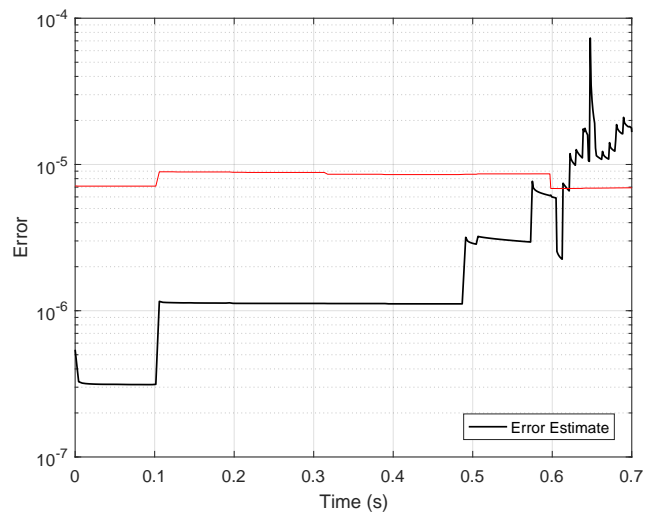
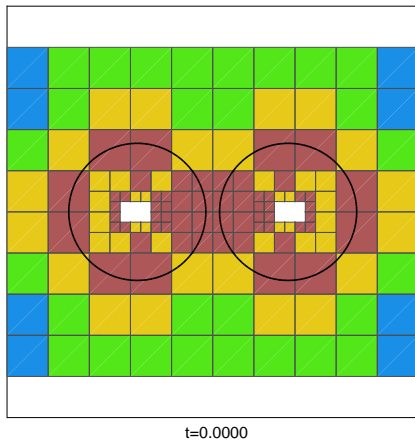
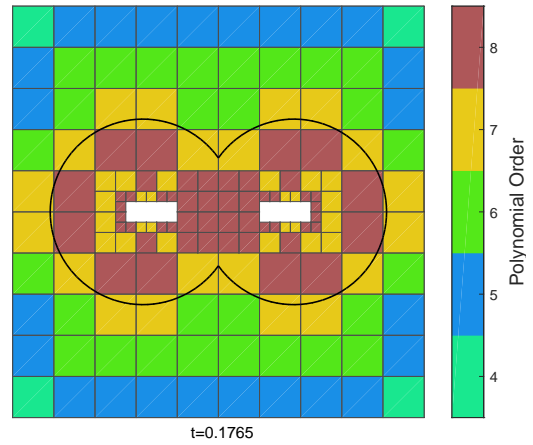


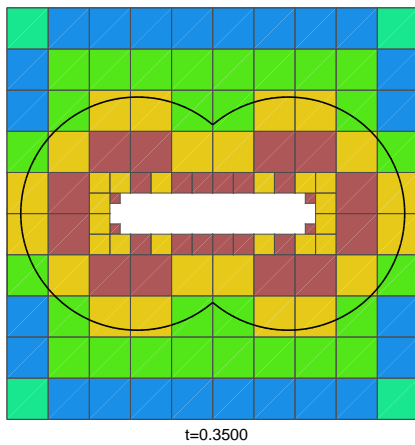
Figure 6.37: Error over pseudotime for the merging circular interfaces problem on an hp -adaptively refined mesh. The red line denotes the error tolerance defining one of the stopping criteria, which can change as the area of narrow band changes.



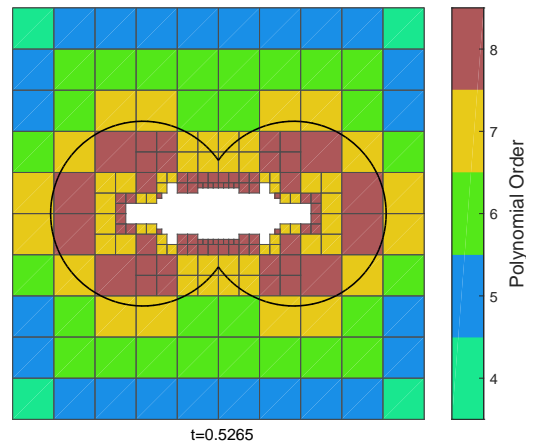
(a) Mesh and interface position at $t_e = 0.0000$



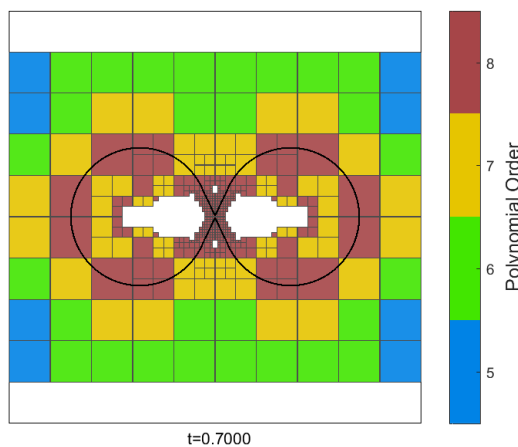
(b) Mesh and interface position at $t_e = 0.1765$.



(c) Mesh and interface position at $t_e = 0.3500$.



(d) Mesh and interface position at $t_e = 0.5265$.



(e) Mesh and interface position at $t_e = 0.7000$.

Figure 6.38: Configuration of mesh and interface position over time for the merging circular interfaces evolution problem, on an hp -adaptively refined mesh. The thick black line denotes the computed position of the interface.

Figure 6.37 shows how the error varies over time for the merging circles example problem on an hp -adaptively refined mesh. After the initialisation, with the stricter error tolerance to be satisfied, the error remains constant in the interval $t_e = (0, 0.48)$. This is because the advection field is constant over the domain for this example, and thus the level set function moves as a rigid body in space. The step in error at around $t_e = 0.1$ is a result of the evolving narrow band tracking the growing interface and thus the lesser refined elements from outside the narrow band moving into the narrow band and with it additional error. This causes the error to increase but not so much that the refinement error threshold is crossed, and thus the simulation continues. After $t_e = 0.35$ the interface begins to shrink. It can be seen in Figure 6.38 that beyond this point, as the interface begins to shrink an extrapolation will have to occur on the elements near to the centre of the domain. As these extrapolations begin the error in the solution begins to grow. As with the analogous problem on the fixed mesh, the extrapolation on the elements close to the $x = 0$ plane, uses information about the gradient from neighbouring elements either side of a singularity, which introduces additional errors into the system. Until around $t_e = 0.61$, the refine-reinitialise routine is able to control these errors. Beyond this point the mesh is maximally refined in this area of largest error and the simulation continues in spite of this. Beyond this point the oscillations in the error which were observed in the shearing circle problem in Section 6.6.4 can be observed once again here, and likely therefore for the same reasons.

Figure 6.38 shows snapshots of the interface and the narrow banded mesh over time, for the merging circular interfaces problem. Whilst not a direct analogue, compared with the fixed mesh, it can be seen that there is much less distortion in the original circles as a result of the increased levels of accuracy afforded by the adaptive mesh.

6.7 Summary

In this chapter an hp -adaptive mesh refinement strategy is proposed for use with the DG discretised narrow banded level set methodology presented in Chapters 4 and 5, driven by a measure of the local signed distance error. Whilst for simple reinitialisation problems the proposed strategy allows for exponential convergence and thus the use of significantly less resources to achieve a given level of accuracy for both reinitialisation and evolution problems, as the complexity of the example problems grows the required amount of resources can quickly grow beyond the capability of a given machine. One reason for this issue is that the convergence rate of the fixed point iterative method used to linearise the reinitialisation and extrapolation problems is shown to decrease with the complexity of the shape implied by the level set interface. As the refinement strategy depends upon repeated series of refinements and reinitialisations until a tolerance is satisfied, slow convergence of the iterative method can mean that the reinitialisation does not achieve the maximum level of accuracy possible for the given mesh before hitting the maximum allowed number of iterations. As the fixed point iteration is shown to be stable and to converge monotonically (when no Anderson acceleration is applied), this implies that given enough time it should be possible to solve the more complex smooth problems using meshes of lower resolution. The trade-off between compute time and memory requirements however, does seem in the current state to swing wildly in favour of one or the other. After further investigation it

can be seen that where the desired level of accuracy is small, even for very simple problems, the repeated L^2 projection of the signed distance function to a circle on an adaptively refining mesh for example, that the required resolution of the discretisation must be high. The implication of this is that capturing interfaces of any complexity on finite element meshes with piecewise polynomial approximation spaces is always going to be a resource intense process. Ultimately then the solutions to these issues likely lies in the investigation of alternative solvers for the reinitialisation/extrapolation methods which have improved convergence properties, and/or the use of high performance computing techniques to find increases in efficiency, however, this is unfortunately beyond of the scope of the research to be presented here. The next chapter proceeds by taking the proposed hp -adaptive DG discretised narrow banded level set methodology and using it in the context which it was originally intended, that is as part of a level set based topology optimisation method.

Chapter 7

Topology Optimisation

Topology optimisation is the most general form of structural optimisation and is concerned with finding the position of the boundary of a given problem domain which minimises an objective functional whilst satisfying a set of constraints. In the last 30 years, topology optimisation has become an increasingly popular tool used by engineers, especially in the early stages of the design of a structure, because it is a methodology which is capable by means independent of a designer to compute designs which can be considered optimal in terms of a given set of criteria, which could include for example structural response, manufacturability or cost [174]. A growing desire for the use of topology optimisation in industry can be seen by the number of companies who now offer topology optimisation modules as part of their commercial CAD/CAE software packages. A notable initial example of this is Altair Engineering's OptiStruct which was first made available in the mid 1990's, however, since then many of the more popular CAE companies have followed suit. To name just a few, Ansys' Ansys Mechanical, Autodesk's Inventor Nastran, and Dassault Systèmes' Abaqus for example, all now offer proprietary topology optimisation modules, and this is a trend which is likely to continue as the academic literature concerning topology optimisation continues to be developed. Furthermore, as the availability of topology optimisation software continues to grow, so does the number of examples of industrial components whose design has benefited from the use of topology optimisation techniques. For example, a list of just a few structures designed in this way includes: the rib structure of the Airbus A380's leading edge droop nose [175]; the front wing of Force India's 2008 F1 car [176]; and improved hip prostheses [177] which are beginning to be used in real patients undergoing total hip arthroplasty.

The aim of this chapter then, is to apply the proposed DG discretised *hp*-adaptive, narrow banded, level set methodology developed in the preceding chapters to topology optimisation. The chapter proceeds as follows: Section 7.1 presents a review of the literature concerning existing approaches to topology optimisation, Section 7.2 and its subsections present the theory necessary to take the proposed level set methodology and apply it to the solution of a generic topology optimisation problem, Section 7.3 and its subsections present the theory necessary to compute the minimum compliance design of a linear elastic structure with a volume constraint using a level set methodology, and Section 7.4 concludes the chapter with a relevant numerical example.

7.1 Literature review: numerical methods for topology optimisation

Much of the early work on structural optimisation focuses on the development of analytical solutions for the optimal design of structures comprised of discrete elements. A notable first in this respect, especially considering the date of the work, is the 1904 article of Michell [178]. In this work Michell presents analytical optimality criteria for the layout of truss structures so as to minimise the amount of material used in the construction of a given structure. In the 1970's, Prager and Rozvany [179–181], expanded upon the work of Michell by presenting practical analytical solutions for problems involving the optimal design of elastic and perfectly plastic grillages (structures comprised of beams). And more recently, Rozvany and Lewinski [182–184], again expanded upon the work of Michell by extending the range of analytical solutions for least weight structures to a variety of domain shapes, loading configurations and boundary conditions. These initial works on structural optimisation do have limited utility for those interested in the optimal design of continuum structures, however, they can still provide some value to the reader as the analytical solutions to benchmark problems can be a useful verification tool for analogue benchmark topology optimisation problems for continuum structures.

The first attempts at the optimisation of continuum structures, using FE discretisations, were mostly focussed on size optimisation, i.e. optimisation problems in which the design variable is the thickness or cross sectional area of the structure. A seminal work in this regard was presented by Rossow and Taylor in their 1973 paper on the variable thickness sheet model [185]. The underlying idea of the variable thickness sheet model was to divide the problem domain into a number of subdomains each of which would have an associated thickness to be optimised. The thickness of a given subdomain becoming close to or equal to zero then implies a change in the shape of a structure towards one which requires less material, and therefore is closer to optimal. In the 1990's, the idea of optimising a shape by removing finite amounts of material from the problem domain was extended leading to the development of a group of methods for computing the optimal design of continuum structures known as hard-kill methods (where the phrase 'hard-kill' refers to the fact that a given subdomain either contains material or doesn't, with no intermediate). The most well known of the hard-kill methods is known as Evolutionary Structural Optimisation (ESO) which was introduced by Xie and Steven [1] in 1993. In ESO each evolutionary step consists of a finite element analysis followed by the removal of any element which meets an elimination criterion, until a steady state solution is reached. Later, Querin [186] developed an Additive Evolutionary Structural Optimisation (AESO), which did the opposite of ESO, it started with a simple structure and used appropriate decision making to add new elements. The combination of ESO and AESO then led to the development of a Bidirectional Evolutionary Structural Optimisation (BESO) [187], in which elements could be added or removed allowing the solutions of ESO and AESO to become closer to optimal. One of the main advantages, and one of the main reasons for the popularity of hard-kill methods is their simplicity and the ease with which they can be integrated into existing FE solvers, often only requiring the implementation of a simple post-processing algorithm. The heuristic nature

of hard-kill methods however makes it difficult to determine whether a computed solution is in fact optimal, and furthermore, it has been shown that for some problems (even quite simple ones), the method converges towards a solution which is highly non-optimal [188].

Another popular group of topology optimisation methods are known as density-based methods. The first of these methods to be developed is known as the homogenisation method which was developed in 1988 by Bendsøe and Kikuchi [2]. The homogenisation method considers a domain filled with a composite material, the constituents of which are a material part and a periodic distribution of small rectangular voids. Using homogenisation theory there is a relationship between the effective material properties in the composite and the size and orientation of the rectangular holes. The optimisation problem then is a size optimisation problem, with the size and orientation of the holes (and therefore the density of the composite), as the design variable. Whilst being popular in commercial software for a while, more recently the popularity of homogenisation approaches has waned with increasing interest in a similar method known as the Solid Isotropic Microstructure with Penalisation (SIMP) method [3]. This shift has been attributed to homogenisation methods being uneconomical and without significant advantages relative to the SIMP method [189]. The SIMP method, instead of introducing a porous medium with a variable density, introduces an artificial density directly, and allows in each element in the mesh this artificial density to vary continuously between 0 and ρ_{max} . In order to avoid elements of intermediate density, the density in each element is penalised by a power law during optimisation which forces the density towards either extremum. In this sense, the SIMP method is similar to the method of variable sheet thickness, and in fact if the power law is linear these methods are equivalent. One of the main issues with the SIMP method is that the computed solutions seem to be highly dependent on both the background mesh and the exponent of the penalty term [189]. Another issue with density-based methods generally, is their lack of generality, only finding use for problems involving specific material models (linear elasticity), and specific objective functionals (compliance, eigenfrequency) [9]. And furthermore, an issue with both hard-kill and density based methods, is that the optimal structures computed using these methods are unable to develop a smooth boundary as the (oftentimes square) elements comprising the domain are either full or empty, which can lead to the aptly named issues of checkerboarding and staircasing, see for example [190].

These are problems which do not occur with a group of topology optimisation methods referred to as boundary variation methods, which are a group of methods concerned with manipulation of the problem boundaries themselves. A number of explicit boundary variation methods were developed in the 1980's, whereby the boundary was defined using either polynomial or spline representations, see for example [4–6]. These explicit methods were found to be plagued with their own set of issues, however, including: extreme computational expense due to remeshing; stability issues caused by element distortion after remeshing; and a requirement for the initial guess to be ‘good enough’ in order to converge to an optimal solution. After the development of the level set method in 1988, it was suggested that one possible application of level sets may be topology optimisation, as a technique for implicit boundary variation. A first attempt at a level set based topology optimisation method was presented in the year

2000 with a paper by Sethian and Weigmann, [107] who combined the level set method with an ESO methodology. What later developed as a more favourable approach to level set based topology optimisation was the shape sensitivity approach as discussed in [8, 9, 191, 192]. The shape sensitivity approach to level set based topology optimisation involves evolving the level set function at each iteration along the descent gradient of the Lagrangian of the given objective functional to be minimised, and thus closer to an optimal configuration. This is achieved by computing an advection velocity field at each iteration based on the variation in the Lagrangian with respect to a change in the current topology of the structure to be optimised. The use of the level set method in this context has grown in popularity since the early 2000's as by defining the boundary of a structure by an implicit function, one alleviates a number of issues with the aforementioned hard-kill and density based topology optimisation methods. In particular, as the boundary of the structure no longer has to conform to the boundaries of the elements, there is no need for elements of intermediate density, phenomena such as checkerboarding and staircasing do not exist, and the mechanical model can be represented much more accurately. Other advantages include: the alleviation of mesh-dependent phenomena (the optimal solution also does not depend on the background mesh); the ease with which one can deal with complex geometries; and the admissibility of complex topological changes (such as the merging of holes). Phase field methods have also found use as a technique for implicit boundary variation, initial attempts at which can be found in [11]. The distribution of material in a domain using the phase field method is implied by a smoothed Heaviside function, referred to as the phase field function, where the extremal values of the function imply two phases (material and void for example). Optimal shapes can be computed using a phase field approach in a manner similar to the level set approach by appropriate evolution of the the phase field function and thus the implied interface. The evolution equations for the phase field method however, take the form of a fourth order Cahn-Hilliard equation, although this is often simplified into a coupled pair of second order PDEs [13]. The phase field function taking the form of a smoothed Heaviside function means that the boundary between the two phases has finite thickness. For this reason, phase field based topology optimisation methods are sometimes considered to be related to density-based methods, in that the phase field variable can be considered as a density, and as such similar techniques are employed using both strategies to deal with elements of intermediate density in the transition region. Whilst there is a growing literature concerning phase field approaches to topology optimisation, the level set method is the generally preferred of the two methods in this context. It has been suggested that the main reasons for this include: the inability to precisely determine the position of the interface in the transition region [12]; and the computational expense associated with solving the equations driving the evolution of the phase field function [12–14].

Level set based topology methods, however, are of course not without flaw. One of the main issues with level set based topology optimisation methods is that when combined with a sufficiently stable time stepping method, topological changes such as the formulation of new holes cannot be introduced during optimisation. This means that the obtained optimal topology will be dependent on the initial distribution of holes and material in the domain, and furthermore,

having an initial guess with many holes is not necessarily sufficient to overcome this. It is for this reason that the development of mechanisms which allow hole nucleation are an active area of research, see for example [193] for a discussion surrounding and proposed solutions to this issue. Other noted issues with such a methodology are criticisms of the classical level set method itself; for example, numerical instabilities can be prevalent unless regularisation methods such as reinitialisation are introduced at which point the accuracy and rate of convergence become dependent on the efficacy of these techniques [190]. This has led to research into improved approaches for computing the evolution of level set boundaries, such as adopting the use of standard mathematical programming techniques such as the method of moving asymptotes [21], or by exploiting the advantages of other discretisation methods such as the eXtended Finite Element Method (XFEM) [16–18] and the Boundary Element Method (BEM) [19, 20].

The main area of novelty in this chapter continues in the vein of those approaches previously stated, in that by alleviating some of the issues with the classical level set method, the information pertaining to which has formed the bulk of this thesis so far, one may then be able to more effectively solve topology optimisation problems using the level set method. As this was the final part of the research to be completed here, however, time restrictions have limited the scope of this part of the research. Therefore, as topology optimisation is such a wide and varied topic, the main focus of the work to be presented will be on the specific example of minimally compliant linear elastic structures with a volume constraint.

7.2 Topology optimisation

A generic topology optimisation problem could be stated as follows

$$\min_{\Omega^{\phi^+} \in \mathcal{U}} R(\Omega^{\phi^+}), \quad (7.1)$$

$$\text{subject to } \begin{cases} \chi_i(\Omega^{\phi^+}) = 0, & \text{for } i = 1, 2, \dots, N_e, \\ \chi_j(\Omega^{\phi^+}) \geq 0, & \text{for } j = N_e + 1, N_e + 2, \dots, N_\chi. \end{cases} \quad (7.2)$$

The solution to (7.1), is a shape (a subdomain or distribution of material over a problem domain), denoted in this case, Ω^{ϕ^+} , in a set of admissible shapes, \mathcal{U} , which minimises a function of that shape, $R(\Omega^{\phi^+})$. The generic optimisation problem (7.1) has an associated set of constraints which form (7.2) and which can be denoted, $\{\chi_i\}_{i=1}^{N_\chi}$, where the total number of constraints is denoted, N_χ , and the total number of equality constraints is denoted N_e . The admissibility of a given shape Ω^{ϕ^+} , which minimises $R(\Omega^{\phi^+})$, is defined by the set of constraints $\{\chi_i\}_{i=1}^{N_\chi}$, in that an inadmissible shape is one which fails to satisfy at least one of these constraints.

7.2.1 Level set based topology optimisation

In order to use the level set method to define the topology of a structure to be optimised, Ω^{ϕ^+} , one can simply use the interface of a level set function to define the boundaries of a structure, which will be denoted Γ . The value of the level set function at any point in a problem domain can then be used to determine whether that point is inside the material part of the structure or an empty part of the domain. In this case, the portion of the domain where, $\phi > 0$, will

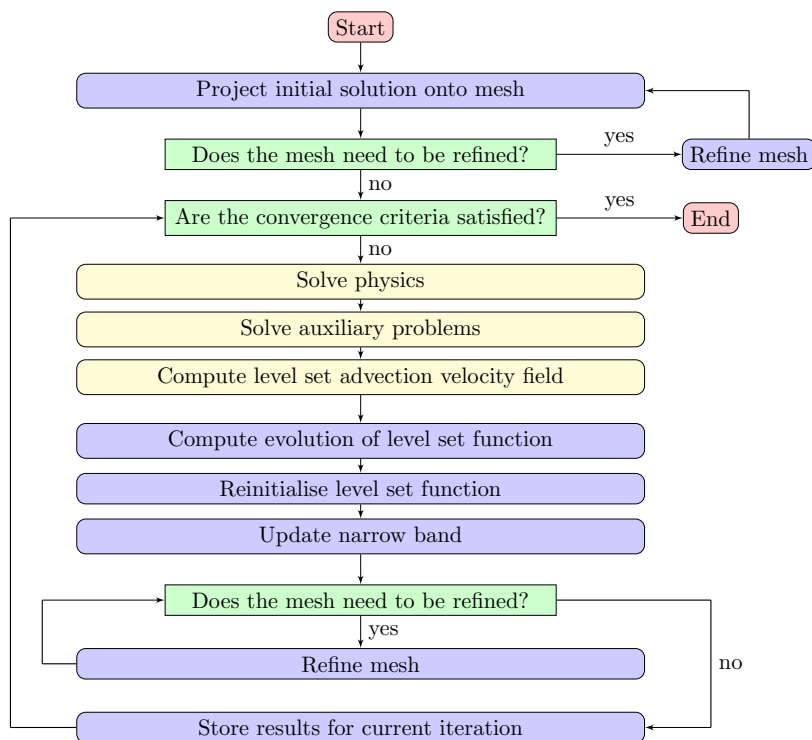


Figure 7.1: Flow chart demonstrating the computational procedure for level set based topology optimisation based on a shape sensitivity analysis.

be defined as the material part of the domain, (i.e. $\Omega^{\phi^+} = \Omega \setminus D$ using the previously defined notation), and the portion of the domain where, $\phi < 0$, will be defined as the empty part of the domain, (i.e. $\Omega \setminus \Omega^{\phi^+} = D$). The real crux of using the level set method as part of the technology for solving a topology optimisation problem, such as that stated in (7.1) and (7.2), is the question of how to then evolve the level set interface, and thus the problem topology, from an initial guess, or any given iteration thereafter, towards an optimum. The approach used here will follow the procedure presented by Allaire *et al.* in [9]. In [9], Allaire uses shape sensitivity analysis to compute the variation in an objective functional with respect to a change in shape of the current topology of a structure undergoing optimisation, at a given time step. The information gained by computing this variation can then be used to compute an advection velocity field over the domain, evolution by which will lead to a new topology at the next time step which diminishes the value of the objective functional. How exactly this is achieved will be detailed in Sections 7.2.2 and 7.2.3.

Figure 7.1 then, shows how the relevant level set technology fits together to form a generic methodology for solving topology optimisation problems using shape sensitivity analysis. It can be seen that in comparison with examples of level set evolution from earlier in the thesis there are only two main differences. First of all, at the beginning of each evolution loop a number of auxiliary problems will need to be solved in order to compute the advection velocity field. This will depend on the specific problem to be solved but will likely include solving the equations

defining the physics of the underlying problem, as well as any other equations arising from the shape sensitivity analysis regarding the chosen set of constraints. And secondly, computing the evolution of the level set function over a specific period of pseudotime is no longer of concern, and instead the evolution is computed until a set of convergence criteria are satisfied, again which will be informed by the objective functional to be minimised and the chosen set of constraints.

7.2.2 Shape sensitivity analysis

Shape sensitivity analysis is a method of computing the variation in system response caused by a change in the shape of the system. To do this the idea of the shape derivative, [196], is introduced which is a measure of how the response function, in this case, the functional to be minimised, R , varies as the domain, Ω^{ϕ^+} , changes shape. The shape derivative of the objective functional can be obtained by computing the derivative of its Lagrangian with respect to the domain, Ω^{ϕ^+} , in the direction ω . For the generic problem (7.1) and its constraints (7.2), the Lagrangian can be stated,

$$\mathcal{L}(\Omega^{\phi^+}, \lambda) = R(\Omega^{\phi^+}) - \sum_i \lambda_i \chi_i(\Omega^{\phi^+}), \quad (7.3)$$

where $\{\lambda_i\}_{i=1}^{N_\chi}$ are Lagrange multipliers associated with each of the constraints in (7.2), and thus the shape derivative takes the form

$$R'(\Omega^{\phi^+})(\omega) = \frac{\partial \mathcal{L}(\Omega^{\phi^+}, \lambda)}{\partial \Omega^{\phi^+}}(\omega). \quad (7.4)$$

Furthermore, for an objective functional or Lagrangian which could be stated

$$R(\Omega^{\phi^+}) = \mathcal{L}(\Omega^{\phi^+}) = \int_{\Omega^{\phi^+}} j(\mathbf{x}) \, d\mathbf{x} + \int_{\Gamma} l(\mathbf{x}) \, ds, \quad (7.5)$$

where $j(\mathbf{x})$ and $l(\mathbf{x})$ are two arbitrary functions, it can be noted that the directional derivative only depends on the normal trace, $\omega \cdot \mathbf{n}_\Gamma$, of the boundary Ω^{ϕ^+} [9], denoted Γ (where again Γ equivalently refers to the zero isocontour of the level set function) and therefore the shape derivative of a Lagrangian of the form (7.5) can be stated as follows

$$\begin{aligned} R'(\Omega^{\phi^+})(\omega) &= \int_{\Omega^{\phi^+}} \nabla \cdot (\omega j(\mathbf{x})) \, d\mathbf{x} + \int_{\Gamma} \omega \cdot \mathbf{n}_\Gamma (\nabla l(\mathbf{x}) \cdot \mathbf{n}_\Gamma + (\nabla \cdot \mathbf{n}_\Gamma) l(\mathbf{x})) \, ds, \\ &= \int_{\Gamma} \omega \cdot \mathbf{n}_\Gamma (j(\mathbf{x}) + \nabla l(\mathbf{x}) \cdot \mathbf{n}_\Gamma + (\nabla \cdot \mathbf{n}_\Gamma) l(\mathbf{x})) \, ds, \end{aligned} \quad (7.6)$$

where $(\nabla \cdot \mathbf{n}_\Gamma)$ is the curvature of the domain boundary (or equivalently the level set interface).

7.2.3 Computing the advection velocity field for a generic level set based topology optimisation

To evolve the level set interface along the descent gradient then amounts to appropriately choosing an advection velocity vector. Again for the generic objective functional to be minimised as

stated above, the shape derivative stated in Equation (7.6) can be rewritten

$$R'(\Omega)(\omega) = \int_{\Gamma} f \omega \cdot \mathbf{n}_{\phi} \, ds, \quad (7.7)$$

where for conciseness the variable f is used to denote $f = (j(\mathbf{x}) + \nabla l(\mathbf{x}) \cdot \mathbf{n}_{\Gamma} + (\nabla \cdot \mathbf{n}_{\Gamma})l(\mathbf{x}))$. Thus, the descent direction is given by the vector field

$$\omega = -f \mathbf{n}_{\phi}. \quad (7.8)$$

It is shown in [9] that evolving the shape in this direction ensures a decrease in the value of the Lagrangian. An appropriate choice for the advection velocity vector, evolution driven by which will minimise the Lagrangian is therefore, the component of the descent direction vector normal to the level set interface and which therefore can be stated

$$b = \omega \cdot \mathbf{n}_{\phi} = -f. \quad (7.9)$$

It should be noted that the advection velocity as stated in (7.9) would only exist on the domain boundary, Γ , and therefore it is generally a requirement that one extend this velocity field from the boundary over the entire domain. A simple way of doing this is to extend the velocity at each point along the boundary, such that it is constant along its normal. In the case that the shape derivative contain quantities which exist throughout the domain, another approach is to compute the shape derivative everywhere in the domain, which then implies the value of the advection velocity field everywhere in the domain. As there exists a number of methods by which one can compute these extension velocities, the reader is referred to the review paper [190] for a discussion of a number of these approaches, specific to level set based topology optimisation methods.

7.3 Minimum compliance of a linear elastic structure subject to a volume constraint

In order to demonstrate how the information presented in Section 7.2 can be used in practice, one can consider the common topology optimisation example of the computation of a maximally stiff (or equivalently, minimally compliant) linear elastic structure subject to a constraint on the volume of the material used in the design.

7.3.1 Linear elasticity equations

In order to use the proposed level set methodology to solve a minimum compliance problem, a DG solution to a linear elasticity problem will firstly be introduced. The variation in stress for a linear elastic structure under an applied traction can be stated as follows

$$\begin{aligned} \nabla \cdot \boldsymbol{\sigma}(\mathbf{u}) &= 0, & \mathbf{u} &\in \Omega, \\ \mathbf{u} &= 0, & \mathbf{u} &\in \partial\Omega_D \text{ and} \\ \mathbf{u} \cdot \hat{\mathbf{n}} &= \mathbf{g}_N, & \mathbf{u} &\in \partial\Omega_N, \end{aligned} \quad (7.10)$$

where $\mathbf{u} = (u_x, u_y)$, denotes the displacement field, and thus the variable, $\boldsymbol{\sigma}(\mathbf{u}) = \{\sigma_{xx}, \sigma_{yy}, \sigma_{xy}\}$ denotes the stress recovered from the displacement solution. There is a homogeneous Dirichlet boundary condition over the part of the boundary of the computational domain denoted, $\partial\Omega_D$, and \mathbf{g}_N denotes the surface tractions applied on the Neumann boundary, $\partial\Omega_N$. Discretising the problem (7.10) using SIPG leads to the well known variational formulation, see for example [165], which can be stated, find $\mathbf{u}_h \in V_{\mathbf{p}}^2(\mathcal{T})$ such that

$$B_{\text{LE}}(\mathbf{u}_h, \mathbf{v}_h) = J_{\text{LE}}(\mathbf{v}_h), \quad \forall \mathbf{v}_h \in V_{\mathbf{p}}^2, \quad (7.11)$$

where

$$B_{\text{LE}}(\mathbf{u}, \mathbf{v}) = (\boldsymbol{\sigma}(\mathbf{u}), \boldsymbol{\varepsilon}(\mathbf{v}))_{\mathcal{T}} - \langle \{\{\boldsymbol{\sigma}(\mathbf{u})\}\}, \llbracket \mathbf{v} \rrbracket \rangle_{S(\mathcal{T}) \cup \partial\Omega_D} - \langle \llbracket \mathbf{u} \rrbracket, \{\{\boldsymbol{\sigma}(\mathbf{v})\}\} \rangle_{S(\mathcal{T}) \cup \partial\Omega_D} + \langle \mu \llbracket \mathbf{u} \rrbracket, \llbracket \mathbf{v} \rrbracket \rangle_{S(\mathcal{T}) \cup \partial\Omega_D}, \quad (7.12)$$

and

$$J_{\text{LE}}(\mathbf{v}) = \langle \mathbf{g}_N, \mathbf{v} \rangle_{\partial\Omega_N}, \quad (7.13)$$

and where $\boldsymbol{\varepsilon}(\mathbf{u}) = \{\varepsilon_{xx}, \varepsilon_{yy}, \varepsilon_{xy}\}$ denotes the recovered strain. For the example problem to be presented here, an assumption of plane stress is made and thus the constitutive law relating the stress and strain can be stated

$$\begin{Bmatrix} \sigma_{xx} \\ \sigma_{yy} \\ \sigma_{xy} \end{Bmatrix} = \frac{Y}{1-\nu^2} \begin{bmatrix} 1 & \nu & 0 \\ \nu & 1 & 0 \\ 0 & 0 & \frac{1-\nu}{2} \end{bmatrix} \begin{Bmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ 2\varepsilon_{xy} \end{Bmatrix}, \quad (7.14)$$

where

$$\begin{Bmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ 2\varepsilon_{xy} \end{Bmatrix} = \begin{bmatrix} \frac{\partial v_x}{\partial x} & 0 \\ 0 & \frac{\partial v_y}{\partial y} \\ \frac{\partial v_x}{\partial y} & \frac{\partial v_y}{\partial x} \end{bmatrix} \begin{Bmatrix} u_x \\ u_y \end{Bmatrix}, \quad (7.15)$$

and where Y and ν denote Young's modulus and Poisson's ratio respectively. Also based on this plane stress assumption, the discontinuity penalisation parameter, μ can be computed as follows when solving the linear elasticity equations

$$\mu = \frac{Y}{1-\nu^2} \frac{10p^2}{h}. \quad (7.16)$$

Finally, as the solution variable is vector valued in this the case, the DG finite element space over which the solution is sought can be stated

$$V_{\mathbf{p}}^2(\mathcal{T}) := \left\{ \mathbf{v}_h \in [L^2(\Omega)]^2 : \forall \tau \in \mathcal{T}, \mathbf{v}_h|_{\tau} \circ f_{\tau} \in [\mathcal{Q}_{p_{\tau}}(\hat{\tau})]^2 \right\}, \quad (7.17)$$

where the superscript reflects the dimensionality of the solution variable.

7.3.2 Minimum Compliance

To minimise the compliance of a linear elastic structure then, is equivalent to minimising the strain energy in the structure and thus, using the notation introduced above, the minimum compliance problem with a volume constraint can be stated

$$\min_{\Omega^{\phi^+} \in \mathcal{U}} \mathcal{C}(\Omega^{\phi^+}) = (\boldsymbol{\sigma}(\mathbf{u}) \cdot \boldsymbol{\varepsilon}(\mathbf{u}))_{\Omega^{\phi^+}} = \langle \mathbf{g}_N, \mathbf{u} \rangle_{\Gamma_N}, \quad (7.18)$$

$$\text{subject to } \begin{cases} (7.10), \\ \frac{|\Omega^{\phi^+}|}{|\Omega|} - A_{\text{req}} = 0, \end{cases} \quad (7.19)$$

where Γ_N is the part of the boundary of the shape, Ω^{ϕ^+} , upon which the tractions are applied (which will be equivalent to $\partial\Omega_N$ in practice), where A_{req} denotes the required volume ratio and where

$$\frac{|\Omega^{\phi^+}|}{|\Omega|} = \frac{\int_{\Omega} \theta(\phi) \, d\mathbf{x}}{\int_{\Omega} d\mathbf{x}}. \quad (7.20)$$

The constraints in (7.19) consist of the linear elasticity equations which define the strain energy and a constraint on the ratio of the domain which can be filled with material, which defines the set of feasible solutions. Ignoring for the time being the volume constraint, as this requires special consideration, the Lagrangian associated with the minimisation of the compliance, i.e. (7.18), can be stated

$$\begin{aligned} \mathcal{L}(\Omega^{\phi^+}, \mathbf{u}, \boldsymbol{\lambda}_p) = & \int_{\Gamma_N} \mathbf{g}_N \cdot \mathbf{u} \, ds + \int_{\Omega^{\phi^+}} \boldsymbol{\sigma}(\mathbf{u}) \cdot \boldsymbol{\varepsilon}(\boldsymbol{\lambda}_p) \, d\mathbf{x} - \int_{\Gamma_N} \mathbf{g}_N \cdot \boldsymbol{\lambda}_p \, ds \\ & - \int_{\Gamma_D} (\boldsymbol{\lambda}_p \cdot \boldsymbol{\sigma}(\mathbf{u})\mathbf{n}_{\Gamma} + \mathbf{u} \cdot \boldsymbol{\sigma}(\boldsymbol{\lambda}_p)\mathbf{n}_{\Gamma}) \, ds, \end{aligned} \quad (7.21)$$

where $\boldsymbol{\lambda}_p$ is a Lagrange multiplier associated with the linear elasticity equations, (7.10). The shape derivative of this functional is a classical result, a derivation of which can be found [9], which can be stated

$$\frac{\partial \mathcal{L}(\Omega^{\phi^+}, \mathbf{u}, \boldsymbol{\lambda}_p)}{\partial \Omega^{\phi^+}}(\omega) = \int_{\Gamma} \omega \cdot \mathbf{n}_{\Gamma} (\boldsymbol{\sigma}(\mathbf{u}) \cdot \boldsymbol{\varepsilon}(\boldsymbol{\lambda}_p)) \, ds = \int_{\Gamma} (-\boldsymbol{\sigma}(\mathbf{u}) \cdot \boldsymbol{\varepsilon}(\mathbf{u}))\omega \cdot \mathbf{n}_{\Gamma} \, ds, \quad (7.22)$$

which follows as the problem can be shown to be self adjoint and thus $\boldsymbol{\lambda}_p = -\mathbf{u}$. As such in order to compute a minimally compliant topology using a level set based topology optimisation method, one can compute the evolution of a level set function driven at each step by an advection velocity vector which can be stated as follows

$$b = \langle \boldsymbol{\sigma}(\mathbf{u}) \cdot \boldsymbol{\varepsilon}(\mathbf{u}) \rangle_{\Gamma}. \quad (7.23)$$

As mentioned in Section 7.2.3, a technique is required to extend the velocity field from the domain boundary across the entire domain. In this case, again drawing inspiration from the work of Allaire [9], this can be done by extending the quantities forming (7.22) and (7.23),

that is the elasticity equations, across the entire computational domain through the use of an *ersatz* material approach. What this means is that the empty/void regions of the computational domain, rather than actually being modelled as empty, are instead modelled as being full, but of a material which is much more compliant than the material filling the material part of the computational domain, that is $Y_{\text{void}} = 10^{-3} \times Y_{\text{material}}$. In this way one can extend the shape derivative from the boundary, Γ , to the entire computational domain, Ω , as follows

$$\mathcal{C}'(\Omega, \mathbf{u})(\omega) = \int_{\Omega} (-\boldsymbol{\sigma}(\mathbf{u}) \cdot \boldsymbol{\varepsilon}(\mathbf{u}))\omega \cdot \mathbf{n}_{\phi} \, ds. \quad (7.24)$$

Thus the level set function describing the current domain configuration can be evolved towards a less compliant configuration by choosing the advection velocity as follows

$$b = (\boldsymbol{\sigma}(\mathbf{u}) \cdot \boldsymbol{\varepsilon}(\mathbf{u}))_{\Omega}. \quad (7.25)$$

It should also be noted that the variation in the material properties across the interface using this ersatz material approach are defined using a smooth step function as follows,

$$Y(\mathbf{x}) = Y_{\text{void}} + \left(\frac{\phi(\mathbf{x})}{\sqrt{\phi(\mathbf{x})^2 + 0.1h_{\text{min}}^2}} + 1 \right) \left(\frac{Y_{\text{material}} - Y_{\text{void}}}{2} \right), \quad (7.26)$$

where h_{min} denotes the edge length of the smallest element in the partition.

7.3.3 Volume constraint

In a number of papers subsequent to Allaire's paper [9] on level set based topology optimisation, for example [34, 195], the volume constraint, as stated in (7.19), is incorporated into the solution using the augmented Lagrangian method. That is, the objective functional for the minimum compliance problem (7.18) with its constraints (7.19) can be converted to an unconstrained optimisation problem of the following objective functional

$$R(\Omega^{\phi^+}; \lambda_A^m, \gamma_A^m) = \mathcal{C}(\Omega^{\phi^+}) + \lambda_A^m \left[\frac{|\Omega^{\phi^+}|}{|\Omega|} - A_{\text{req}} \right] + \frac{1}{2\gamma_A^m} \left[\frac{|\Omega^{\phi^+}|}{|\Omega|} - A_{\text{req}} \right]^2, \quad (7.27)$$

where λ_A^m denotes a Lagrange multiplier and γ_A^m denotes a penalty parameter, with the superscript referring in both cases to the m^{th} iteration of the solution to the minimisation problem. The augmented Lagrangian approach begins by attempting to enforce the volume constraint using a penalty method. This can be seen by the quadratic penalty term in (7.27). The issue with using a penalty method in this context is that a given solution, Ω^{ϕ^+} , to the minimisation problem, (7.27), is only guaranteed to be feasible, (i.e. to satisfy the volume constraint), as the penalty parameter, $\gamma_A^m \rightarrow 0$. As the penalty parameter grows small, however, the problem will become poorly conditioned. The purpose of the augmented Lagrangian method is an attempt to alleviate this issue by making the minimisers of $R(\Omega^{\phi^+})$ more feasible for more moderate penalty parameters, which is achieved by including an explicit estimate of the multiplier associated with the standard Lagrangian, λ_A^m . For these reasons then, both the penalty parameter γ_A^m and

the estimated Lagrange multiplier λ_A^m are given initial guesses, γ_A^0 and λ_A^0 , chosen by the user, which are then updated during the optimisation to more appropriately enforce the constraint. To ensure that upon convergence the penalty parameter is sufficiently large to ensure that the solution is feasible, it is updated as follows at each time step

$$\gamma_A^m = z\gamma_A^{m-1}, \quad (7.28)$$

where the parameter z is a positive constant to be chosen by the user such that $z \in (0, 1)$. The estimated Lagrange multiplier, can then be updated such that it reduces the infeasibility in each subsequent minimiser, as follows

$$\lambda_A^m = \lambda_A^{m-1} + \frac{1}{\gamma_A^{m-1}} \left[\frac{|\Omega^{\phi^+}|}{|\Omega|} - A_{\text{req}} \right]. \quad (7.29)$$

The shape derivative associated with the objective functional (7.27), has also been derived in numerous articles, see for example [34, 194, 195], and can be stated as

$$R'(\Omega^{\phi^+}; \lambda_A, \gamma_A)(\omega) = \int_{\Gamma} \left(\lambda_A + \frac{1}{\gamma_A} \left[\frac{|\Omega^{\phi^+}|}{|\Omega|} - A_{\text{req}} \right] - \boldsymbol{\sigma}(\mathbf{u}) \cdot \boldsymbol{\varepsilon}(\mathbf{u}) \right) \omega \cdot \hat{\mathbf{n}} \, d\mathbf{x}, \quad (7.30)$$

The terms in the shape derivative associated with the augmented Lagrangian method are constants and thus extend naturally over the domain, and with the elasticity terms already dealt with via the inclusion of an ersatz material approach, the advection velocity field over a problem domain which will minimise the compliance of a linear elastic structure subject to a volume constraint can be stated as follows

$$b = - \left(\lambda_A + \frac{1}{\gamma_A} \left[\frac{|\Omega^{\phi^+}|}{|\Omega|} - A_{\text{req}} \right] - \boldsymbol{\sigma}(\mathbf{u}) \cdot \boldsymbol{\varepsilon}(\mathbf{u}) \right)_{\Omega}. \quad (7.31)$$

7.4 Discontinuous Galerkin based level set topology optimisation: numerical example

7.4.1 Algorithm

As a number of changes have been made to Algorithm 4 with regards to using the proposed *hp*-adaptive level set methodology when it comes to solving a minimum compliance problem subject to a volume constraint, the specific algorithm to be used for solving the example problem in this chapter is presented in Algorithm 5. Algorithm 5 begins as previously by setting up the problem domain, and initial level set function, which in this case will always be chosen such that the analytical function describing the level set function is a signed distance function, and then computing an initial narrow banded mesh. This information is then passed into the `refineUpdate` function, where the initial mesh is adaptively refined until a threshold on the error is satisfied (or upper limits on refinement are reached). In this case, as the initial level set function

Input: Ω, ϕ^0
Output: $\phi_h(\mathbf{x}, t_e)$
Initialise problem parameters: ϕ_h^0, \mathcal{T} ;
Compute initial Narrow Band, \mathcal{T}_{NB} ;
 $[\phi, \mathcal{T}_{\text{NB}}] = \text{refineUpdate}(\phi_h^0, \mathcal{T}_{\text{NB}}, \mathcal{T}_{\text{NB}}, 1e-2, [0.08 \ 0.75], 4, 3, \text{'L2Error'}, 0.25, \text{'initialise'}, 100)$;
while $|A - A_{\text{req}}| > 10^{-5}$ **AND** $\text{all}(|R^m - R^{m-5:m-1}| > 10^{-3}|R^m|)$ **do**
 Compute stress and strain solutions to elasticity problem, $\boldsymbol{\sigma}(\mathbf{u}), \boldsymbol{\varepsilon}(\mathbf{u})$;
 Update augmented Lagrangian parameters, λ_A^m, γ_A^m ;
 Compute advection velocity vector, \mathbf{b} ;
 Compute time step, Δt_e ;
 Evolve level set interface by solving (5.6), $\phi_h^m(\mathbf{x})$;
 Reinitialise level set function, $\phi_h^m(\mathbf{x})$;
 Update narrow band, \mathcal{T}_{NB} ;
 if *elements move from outside to inside narrow band* **then**
 | Extrapolate the level set function onto those elements, $\phi_h^m(\mathbf{x})$;
 end
 $[\phi, \mathcal{T}_{\text{NB}}] = \text{refineUpdate}(\phi_h^m, \mathcal{T}_{\text{NB}}, \mathcal{T}_{\text{NB}}, 1e-2, [0.08 \ 0.75], 4, 3, \text{'signedDistanceError'}, 0.25, \text{'reinitialise'}, 25)$;
 $m = m + 1$;
end

Algorithm 5: *hp*-adaptive narrow banded level set evolution algorithm for minimum compliance problems.

will always be chosen as a signed distance function, the refinement algorithm can be driven by the L^2 error in the projection until a suitable mesh is computed. Another change made here is that the upper limits on p -refinement have been decreased from a maximum polynomial order of 8 to a maximum polynomial order of 3. The reason for this change is that the computation required to solve the linear elasticity equations (i.e. the construction and factorisation of the stiffness matrix) can rapidly approach upper limits on memory for such high order polynomials, especially given that the linear elasticity equations are of higher dimension, and do not benefit from the narrow band. Similarly the threshold value itself is chosen to be smaller in this case as the problems are expected to rapidly become singular.

Once the initial mesh is generated it can be passed into the evolution loop. The first change to the evolution loop is the stopping criterion. As the aim now is to look for a feasible steady state solution to the minimisation problem, inspired by [194], two criteria must be satisfied to end the simulation; firstly the relative change in the value of the objective functional for all of the previous 5 iterations must be less than a threshold value dependent on the absolute value of the current value of the objective functional, $\text{all}(|R^m - R^{m-5:m-1}| < 10^{-3}|R^m|)$, which ensures a steady state solution has been reached; and secondly the volume constraint must be satisfied to a given threshold, $|A - A_{\text{req}}| < 10^{-5}$, which ensures that the solution is feasible.

Next, in order to compute the advection velocity vector, one needs to compute stress and strain solutions to the current configuration of the linear elasticity equations and then similarly update the estimate for the current Lagrange multiplier, as well as the penalty parameter re-

sponsible for enforcing the volume constraint. The advection velocity can then be computed as stated in Equation (7.31). One issue which was noted in the course of experimenting was that for certain parametrisations, the penalty terms associated with the volume constraint can quickly outgrow the stress solution or vice versa which can lead to large variations in the velocity field across the domain. This can then lead to slow convergence in some regions, as the computed time step (and thus the maximum distance the level set interface can move during a given step) is inversely proportional to the maximum computed velocity in the mesh. The solution to this issue is a careful consideration of the initial choices for these parameters. The evolution loop then continues in much the same way as previously, the level set function is evolved and then reinitialised, after which the narrow band is updated, and if need be the refinement algorithm can be called to generate a mesh which more precisely describes the current position of the level set function.

7.4.2 Numerical example: minimum compliance design of a cantilever beam subject to a volume constraint

This example problem comprises the minimum compliance design of a cantilever beam. For this specific example the domain will be configured as shown in Figure 7.2, that is a 2D cantilever beam which is twice as long as it is tall, and in this case $L = 1\text{m}$. The entire left edge will have a homogeneous Dirichlet boundary condition, and there will be a traction, $\mathbf{g}_N = 0.01\text{N}$, applied uniformly to $0.05L$ either side of the the centre of the right edge. All other edges have a homogeneous Neumann boundary condition.

The part of the domain that is full of material will consist of a material with Young's modulus, $Y_{\text{material}} = 1\text{Pa}$ and Poisson's ratio, $\nu = 0.3$, and the empty part of the domain will be filled with a material with Young's modulus $Y_{\text{void}} = 10^{-3}\text{Pa}$ and Poisson's ration $\nu = 0.3$. In the vicinity of the level set interface these material parameters will vary smoothly according to the relation described by (7.26). The initial level set function is defined on the domain, $\Omega = (0, 2) \times (0, 1)$, and can be stated as follows

$$\tilde{\phi}^0 = \min(\tilde{\phi}_k^0 - r), \quad k = 1, \dots, 10, \quad (7.32)$$

where $r = 0.1$ and

$$\begin{aligned} \tilde{\phi}_1^0 &= \sqrt{x^2 + (y - 0.25)^2}, & \tilde{\phi}_{10}^0 &= \sqrt{(x - 1)^2 + (y - 1)^2}, \\ \tilde{\phi}_2^0 &= \sqrt{x^2 + (y - 0.75)^2}, & \tilde{\phi}_{11}^0 &= \sqrt{(x - 1.33)^2 + (y - 0.25)^2}, \\ \tilde{\phi}_3^0 &= \sqrt{(x - 0.33)^2 + y^2}, & \tilde{\phi}_{12}^0 &= \sqrt{(x - 1.33)^2 + (y - 0.75)^2}, \\ \tilde{\phi}_4^0 &= \sqrt{(x - 0.33)^2 + (y - 0.5)^2}, & \tilde{\phi}_{13}^0 &= \sqrt{(x - 1.66)^2 + y^2}, \\ \tilde{\phi}_5^0 &= \sqrt{(x - 0.33)^2 + (y - 1)^2}, & \tilde{\phi}_{14}^0 &= \sqrt{(x - 1.66)^2 + (y - 0.5)^2}, \\ \tilde{\phi}_6^0 &= \sqrt{(x - 0.66)^2 + (y - 0.25)^2}, & \tilde{\phi}_{15}^0 &= \sqrt{(x - 1.66)^2 + (y - 1)^2}, \\ \tilde{\phi}_7^0 &= \sqrt{(x - 0.66)^2 + (y - 0.75)^2}, & \tilde{\phi}_{16}^0 &= \sqrt{(x - 2)^2 + (y - 0.25)^2}, \\ \tilde{\phi}_8^0 &= \sqrt{(x - 1)^2 + y^2}, & \tilde{\phi}_{17}^0 &= \sqrt{(x - 2)^2 + (y - 0.75)^2}, \\ \tilde{\phi}_9^0 &= \sqrt{(x - 1)^2 + (y - 0.5)^2}, & & \end{aligned} \quad (7.33)$$

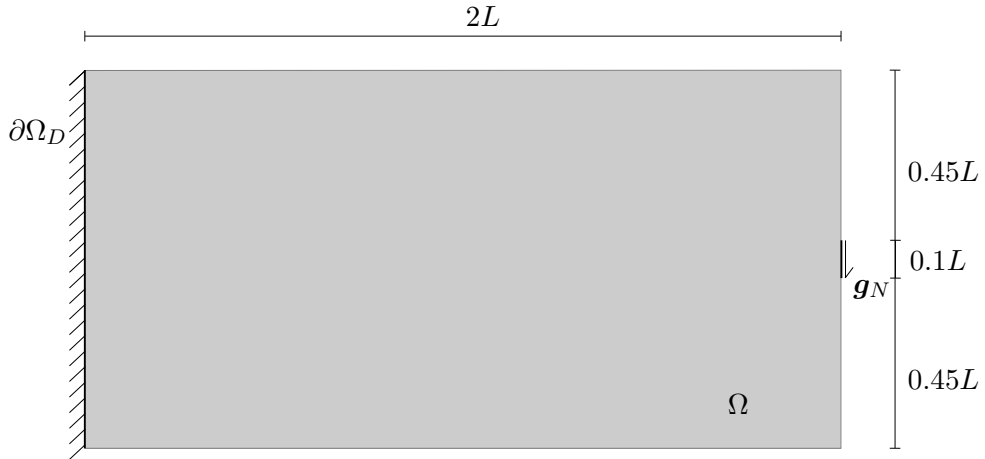


Figure 7.2: Domain configuration for the cantilever beam topology optimisation problem, showing boundary and loading conditions.

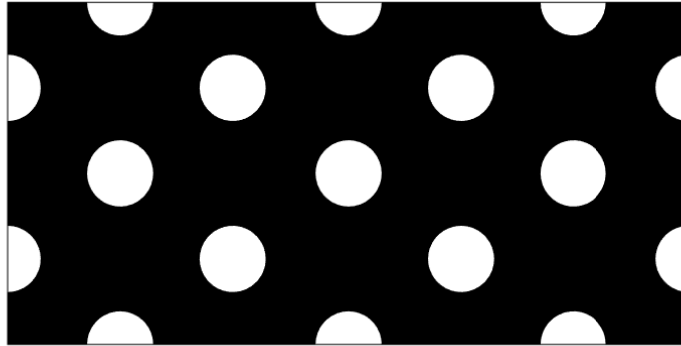


Figure 7.3: Initial topology for the minimum compliance design of a cantilever beam.

This initial distribution of holes can be seen in Figure 7.3.

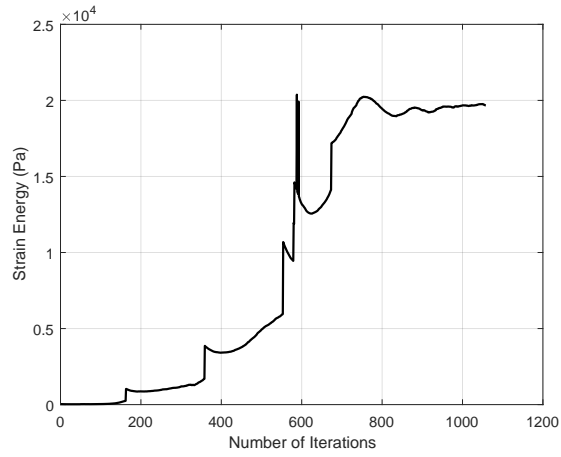
The required proportion of the domain permitted to be filled with material upon convergence is defined as $A_{\text{req}} = 0.3$. As little guidance is provided in the literature explaining how to appropriately choose the parameters associated with augmented Lagrangian method, for the purposes of this numerical example an appropriate strategy was determined by experimentation. It was found that an appropriate choice for the initial estimate for the Lagrange Multiplier, λ_A^1 , is that it should be small in comparison with the maximum elementwise strain energy of the initial configuration, and also therefore it is appropriate to choose the penalty parameter, γ_A^1 , to be large. Similarly it is useful to ensure that the decay of the penalty parameter, and thus by virtue of (7.29), the growth of the estimated Lagrange multiplier happens slowly to ensure stability which implies that the parameter z should be chosen such that it is only slightly smaller than unity. However, care should also be taken to ensure that the combination of these values does not cause the estimated Lagrange multiplier to be too small for many iterations, as

this can cause the compliance terms to dominate the evolution for too long which could lead to premature closure of holes, which without a hole nucleation mechanism would not be able to be recreated. An appropriate balance was found through the following choices, $\lambda_A^1 = 0.001$, $\gamma_A^1 = 10000$ and the decay rate $z = 0.99$.

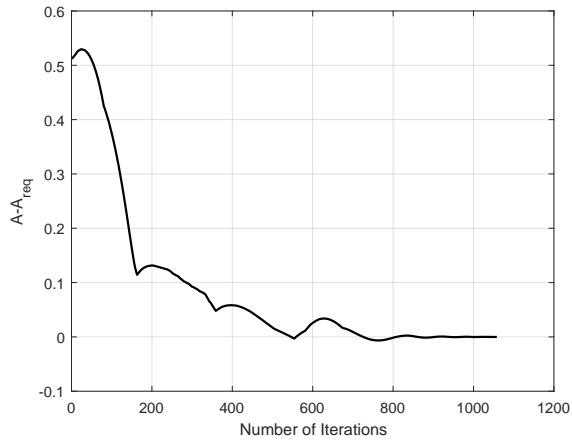
The variation in the compliance and volume constraint during the optimisation process are shown in Figures 7.4(a) and 7.4(b). The first point of note is that there is a large growth in the strain energy across the first ~ 750 iterations. This coincides with moving towards the region of feasibility. It could be noted for example, that the minimum possible compliance solution is the trivial solution where the entire beam is filled with the stiffer material, however, such a solution does not belong to the space of admissible solutions. Thus, whilst there are regions prior to the 750th iteration where the objective functional appears to level off, for example during the first 100 iterations, this doesn't constitute convergence as the solution is yet to be feasible. The first solutions computed which approach the admissible region occur when the Lagrange multiplier hits its peak value, see Figure 7.4(c). Choosing the augmented Lagrange parameters, λ_A^1 , γ_A^1 and z , as they have been chosen, allows for the slow growth of the Lagrange multiplier up to this peak value, and also helps to minimise the overshoot at the point where this peak is reached. By ensuring appropriate values of these parameters it can be seen that during this initial period of optimisation there is a good balance between the shrinking of the material part of the domain as it tends towards the feasible region, and the redistribution of material away from areas of low stress and towards areas of high stress. This can be seen in Figure 7.5, whereby there is much greater variation in the computed shape prior to the Lagrange multiplier reaching its peak value. Once the solution has first approached the feasible region, the remaining iterations consists of smoothing out the Lagrange multiplier estimate, followed by a sharpening of the internal structures which ensures the most efficient distribution of material for the given objective functional.

Another point of note are the non smooth regions in the strain energy and volume constraint which occur throughout the optimisation. These changes occur as a result of the evolving narrow band. As the level set interface evolves, elements which are inside the narrow band, move outside the narrow band and the value of the level set function on these elements is set to a large constant, which causes a sharp change in the material parameters at these locations across a single iteration. This is most significant in areas of high stress. For example, the spikes at iterations 589 and 593 occurs as a result of the elements upon which the tractions are applied moving outside the narrow band.

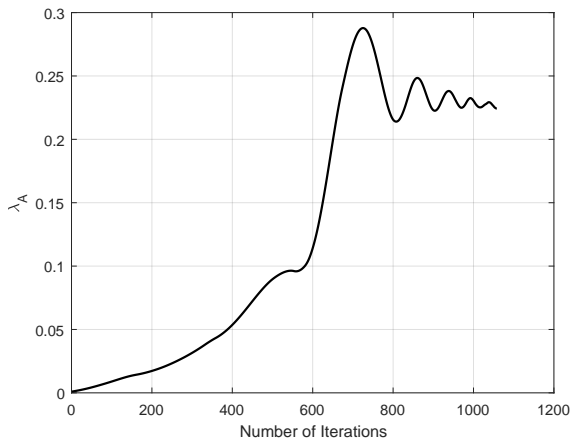
The design upon which the optimisation converges can be seen in Figure 7.6, and the narrow banded mesh configuration describing this solution is shown in Figure 7.8. The solution presented shows good agreement with that found in the literature [198], including the analytical solution for the same problem (which exists for structures consisting of discrete elements) [182]. It can also be noted, as seen in Figures 7.7 and 7.9 which show the topology and mesh at iteration 310, that one of the benefits of using the *hp*-adaptive mesh in this example problem, is that such a mesh allows for sharp resolution of the thin branches of the internal structure, and similarly the precise capture of the sharp corners which develop between these branches.



(a) Strain energy (compliance)

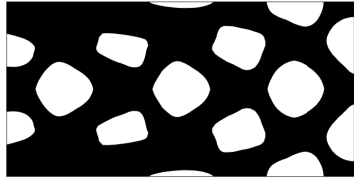


(b) Volume constraint

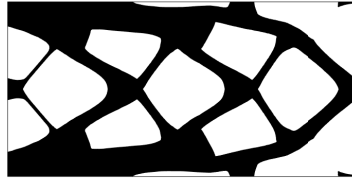


(c) Lagrange multiplier estimate

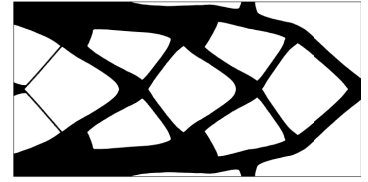
Figure 7.4: Compliance, volume ratio and Lagrange multiplier estimates during the optimisation for the minimum compliance design of a cantilever beam subject to a volume constraint.



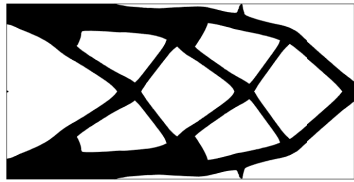
(a) Shape at iteration 100



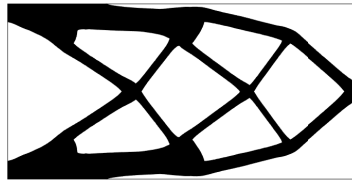
(b) Shape at iteration 200



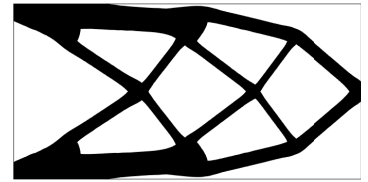
(c) Shape at iteration 300



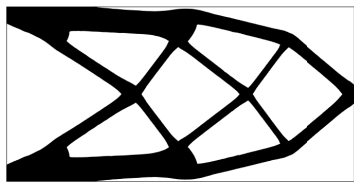
(d) Shape at iteration 400



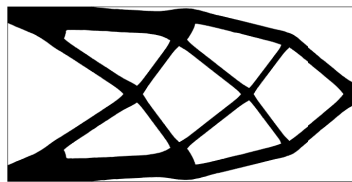
(e) Shape at iteration 500



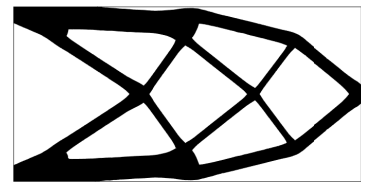
(f) Shape at iteration 600



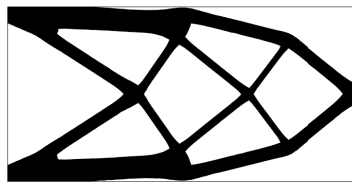
(g) Shape at iteration 700



(h) Shape at iteration 800



(i) Shape at iteration 900



(j) Shape at iteration 1000

Figure 7.5: Varying topology over pseudotime for the optimising cantilever beam.

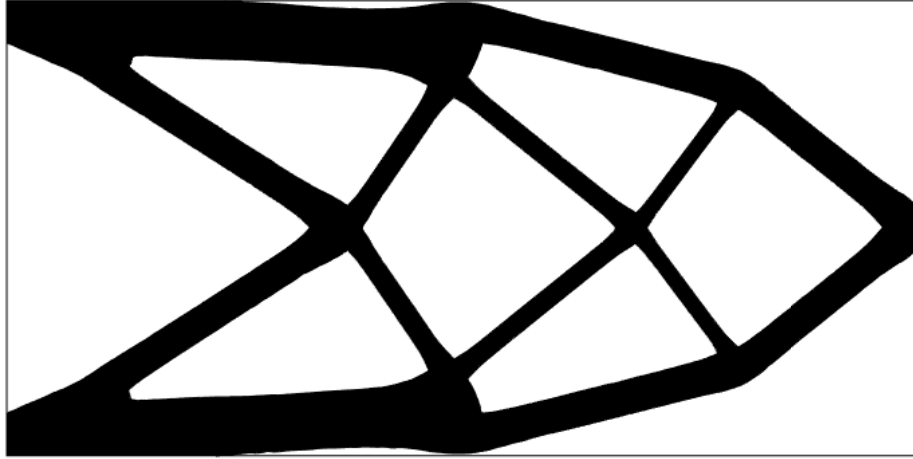


Figure 7.6: Converged solution to the minimum compliance design of a cantilever beam subject a volume constraint.

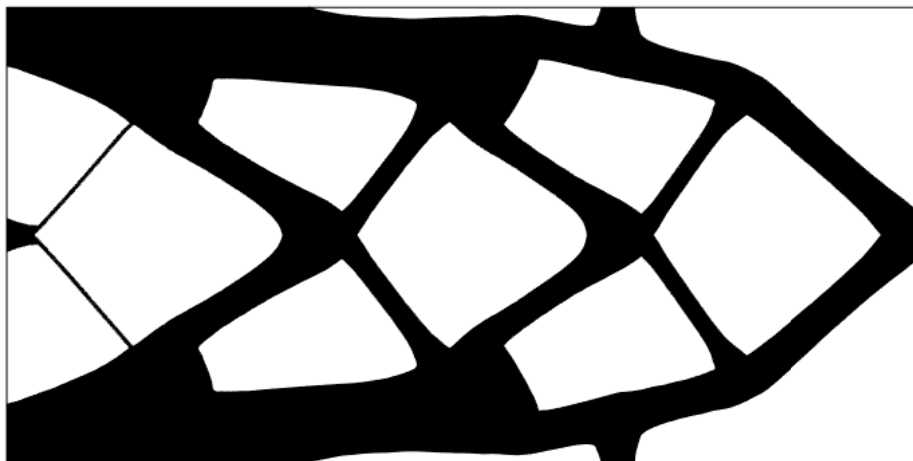


Figure 7.7: Solution to the minimum compliance design of a cantilever beam subject to a volume constraint after 310 iterations.

7.5 Summary

This chapter takes the proposed hp -adaptive DG discretised narrow banded level set methodology and uses it in combination with shape sensitivity analysis and an augmented Lagrangian method to form a level set based topology optimisation method. The proposed topology optimisation method is then used to solve the standard benchmark problem of the minimum compliance design of a linear elastic cantilever beam with a volume constraint. Whilst the results presented are only preliminary, it is found that the solution aligns well with that presented elsewhere in the literature, however, in comparison with the results found in the literature, the proposed method is capable of resolving well both very thin branches of the internal structure within the beam, as well as sharp corners between these branches of the internal structure.

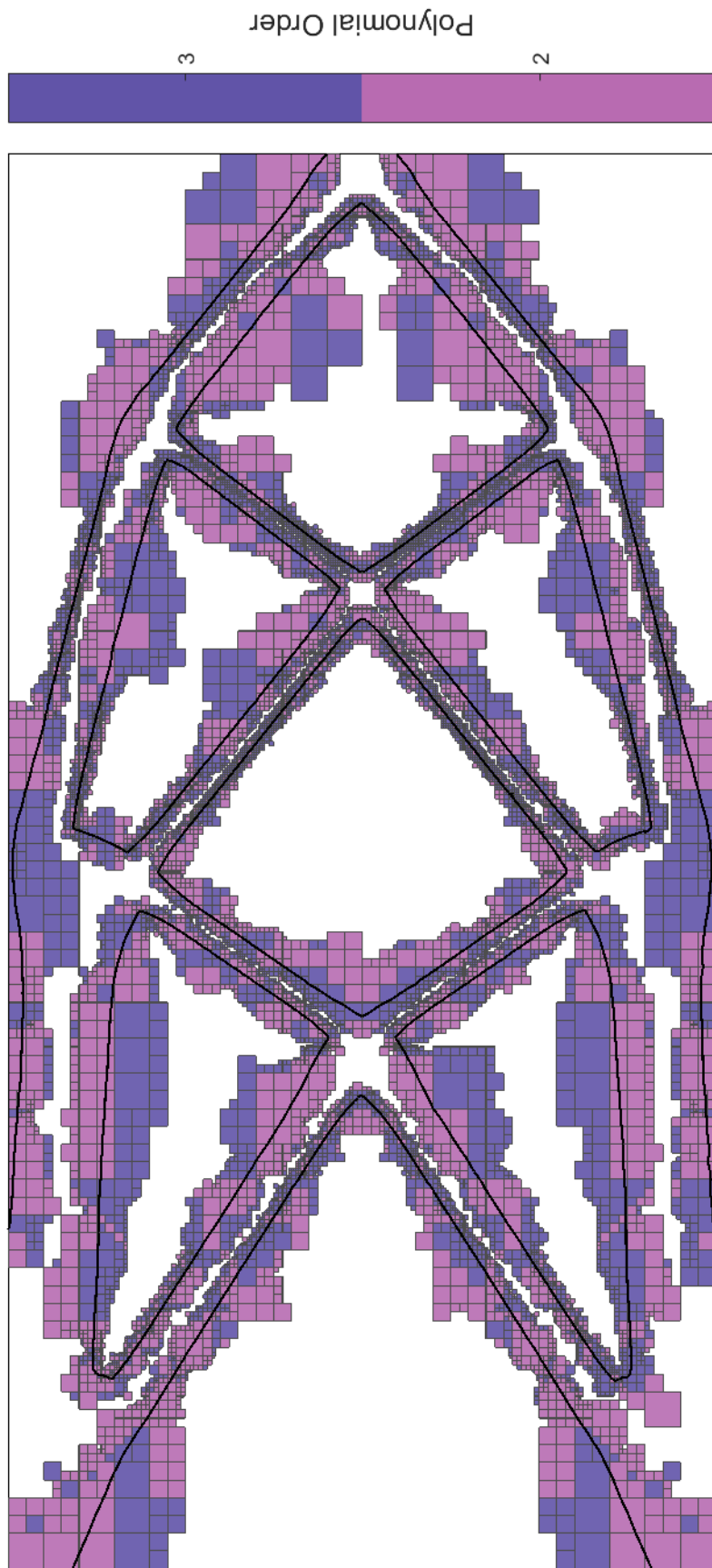


Figure 7.8: Narrow banded mesh at convergence for the minimum compliance design of a cantilever beam subject to a volume constraint. The thick black line denotes the position of the level set interface.

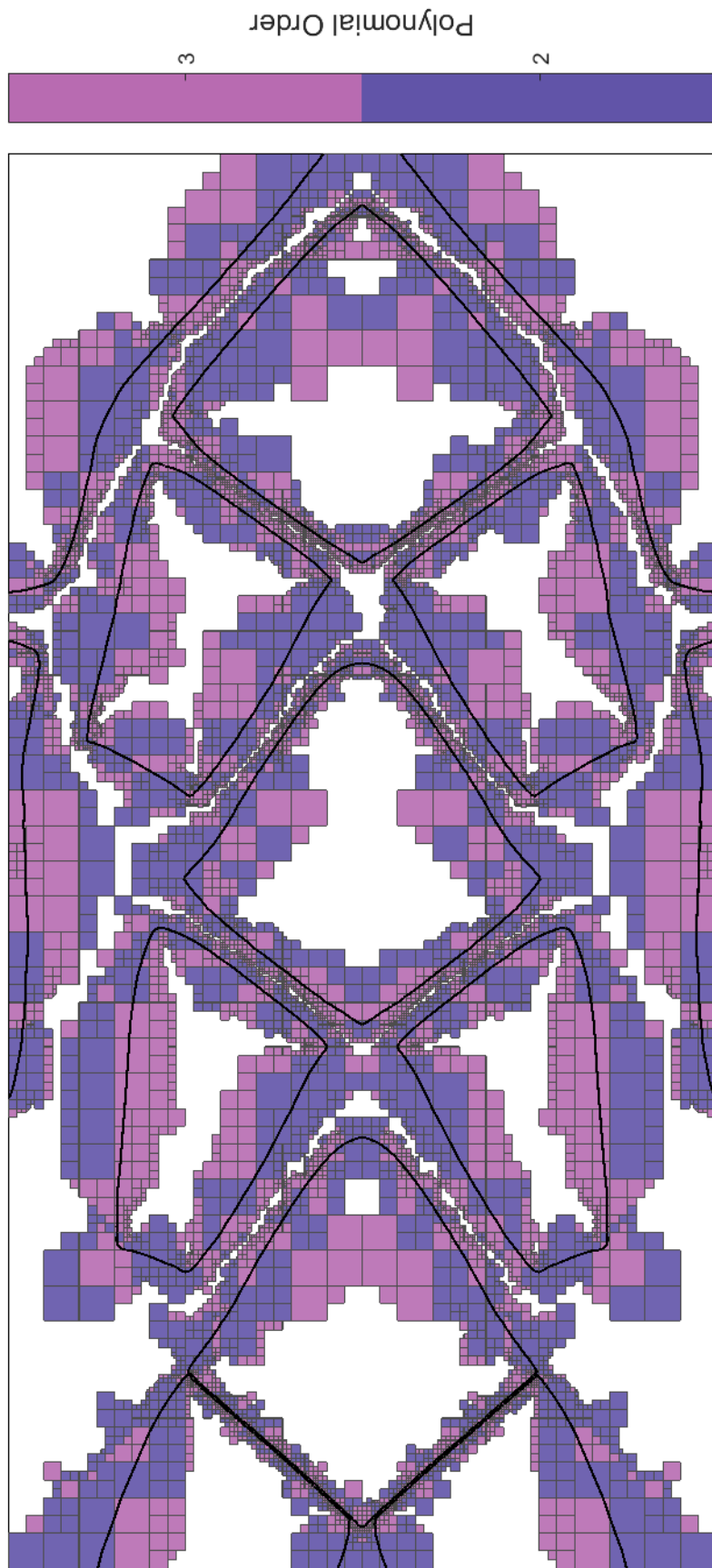


Figure 7.9: Narrow banded mesh after 310 iterations whilst computing the minimum compliance design of a cantilever beam subject to a volume constraint. The thick black line denotes the position of the level set interface.

Chapter 8

Conclusions

8.1 Key developments

This thesis has presented research into level set methods discretised using DG methods in the context of topology optimisation. To this end a methodology is proposed for solving evolving interface problems using an *hp*-adaptive, narrow banded, level set evolution method, which depends on a high-order accurate level set reinitialisation and extrapolation technique, where all of the underlying PDEs are discretised using an appropriate discretisation technique from the family of DG methods. A number of novelties have been developed in order to achieve this end. The first of these novelties was the development of two PDE based level set reinitialisation methods. Specifically an elliptic and parabolic formulation of the Eikonal minimising reinitialisation problem were developed, which when discretised using an SIPG method demonstrated not only high-order accuracy but a significant improvement over other reinitialisation methods in regards to the amount of movement at the level set interface during reinitialisation. This was achieved by modifying the underlying objective functional to more adeptly deal with small gradients, by employing an appropriate narrow banding strategy, and by employing appropriate means for dealing with a Dirichlet boundary condition over an immersed implicit surface. Secondly, with the importance of frequent reinitialisation when the desired level of accuracy is high, it becomes possible to simplify the evolution equation. This allowed for a novel DG spatial discretisation of the simplified evolution equation, which when combined with a high-order RK temporal discretisation was also capable of demonstrating the desired high-order accuracy for smooth problems. Next, given that narrow banding is an important part of the proposed methodology, some attention was paid to how to do this effectively given the proposed paradigm, which led to the development of a novel technique for extrapolating the value of the level set function to elements outside of the narrow band. After this, a novel refinement strategy is proposed which is driven by a criterion based on a measure of the degree to which the level set function satisfies the Eikonal equation locally. The proposed refinement strategy allowed for the demonstration of exponential convergence for a number of ‘static’ reinitialisation problems, as well as allowing one to maintain control over the gradient of the level set function (the norm of which is to remain equal to unity for all time) throughout an evolution, given sufficient memory, which contributes positively to accuracy in the L^2 norm of the solution. Finally, these constituent parts, as discussed above,

have been combined to form a full level set methodology which has been applied preliminarily to the application of topology optimisation.

8.2 Suggestions for future work

The first areas of future work which should be completed, should one wish to continue the research presented in this thesis, would be finding appropriate or improved solutions to a number of key issues encountered with the work presented so far. Firstly, whilst there are definite advantages to using the proposed reinitialisation methods presented in Chapter 4, it became apparent towards the end of the research period that the linearisation of the Elliptic reinitialisation method (and thus the related extrapolation method) by Picard's method is unsatisfactory, especially as the shapes described by the level set interface become more complex and therefore difficult to solve. Whilst the reinitialisation method as presented utilising the Picard iterative method does converge monotonically towards the solution, the convergence rate of the method can and does become prohibitively slow which means one has to decide between a less accurate solution computed in a reasonable number of iterations or a more accurate solution which may take much longer to compute. Furthermore, this is an issue which can compound with the proposed adaptive mesh refinement strategy as failure to satisfy a tolerance on the error, which may be due to stagnation of the method as opposed to an insufficiently resolved mesh, will trigger further mesh refinement, leading to possible over-refinement and thus a further decrease in efficiency. The solution to this issue is to linearise the reinitialisation problem using a method other than Picard. Attempts were made at this during the research period. For example, some brief experimentation was conducted with Newton type schemes (Newton, quasi-Newton and secant methods) for solving the nonlinear elliptic reinitialisation problem. The expense per iteration did obviously increase, particularly when computing the Jacobian numerically, however, the Newton type schemes were able to demonstrate an increased convergence rate which was promising. The conclusion of this brief experimentation however was that the improved convergence rate would invariably lead to overshoots and therefore instability. Furthermore, it was found around this time that slight improvements could be made for sufficiently smooth problems with the incorporation of an Anderson acceleration algorithm, as discussed in Chapter 5, and as such given the time constraints of the project it was decided to move on with a method that if not fast was at least stable, and given enough time accurate too. There is, however, an abundance of research into nonlinear solvers, even for nonlinear problems discretised using DG methods, and thus there are improvements to be made here. For example, it is likely that there exists a suitable relaxation technique which would allow a Newton type scheme to become a real viable alternative. Similarly, there might be improvements which could be made to the parabolic form of the reinitialisation problem, for example, the convergence of the proposed parabolic method could likely be improved using adaptive pseudo time stepping which may render the method preferable as a constituent of the proposed level set methodology. Ultimately, leveraging the advantageous accuracy of the proposed reinitialisation method by improving the convergence rate would be the premier recommendation for someone looking to continue the research presented in this thesis.

The strange behaviour of the chosen hp -steering criterion is another area which it would have been interesting to investigate. It was shown in a number of example problems that the chosen method would struggle to identify a singularity particularly in the case that the singularity existed at a vertex or between two elements, which would then lead to additional refinement in p which may not have been necessary. This was a particularly difficult issue to diagnose, as for other almost equivalent examples no such problem would occur. As this was the case, again the method was deemed to be sufficient in its current form for the purposes of this thesis. However, as mentioned in Section 6.3.2, there does exist in the literature a number of alternative hp -steering criteria which were not experimented with during the research period, and as such it could be the case that simply adopting a different approach might suffice to solve this issue. In any case, where comparative studies have been conducted into the efficiency of various hp -steering criteria, [170] for example, which method performs best seems to be problem specific and thus it would likely be advantageous to conduct such an investigation to ensure an appropriate choice is being made.

Beyond these issues moving from MATLAB to a compiled language and incorporating high performance computing techniques would not only be a massive quality of life improvement for the types of smaller problems presented in this thesis but would absolutely be necessary for moving beyond benchmark type topology optimisation problems, to real applications. In its current form, the proposed level set methodology is not as efficient as is often implied by the generic statement that implicit methods are more efficient than explicit methods. In fact, given that the required resolution for a given level of accuracy is often so high, and that for stability reasons modifications have to be made, as was done here with the inclusion of reinitialisation, to ensure stability and accuracy, it would be interesting to see how the efficiency of the method truly compares with competitive methods for interface tracking generally, but topology optimisation methods specifically, after the inclusion of any such auxiliary methods. In any case, one of the primary motivations of using DG methods in the first place was the high levels of parallelisation of which the method is capable, which, to some degree regardless of application, would be a time saver, but perhaps specifically here as it is known that topology optimisation is always going to be a time expensive problem to solve accurately. It was unfortunately the case that the work completed during the research period had to guide the direction of the research. Since it is somewhat a given that parallelisation would decrease the time taken to solve any of the given problems, and as there was work which needed to be done on the method itself, parallelisation had to be lowered as a priority. Once the issues mentioned above are ironed out however, the next recommended step would be to update the codebase using an appropriate compiled language and to parallelise that which can be parallelised, in particular first ports of call should be the physics solver and the reinitialisation methods which are the two most expensive parts of the methodology.

Another area of work omitted from the thesis would be the extension of the methodology to the solution of 3D problems. Extending the method to 3D was never an explicit goal of the research to be presented here, as of course the development of a 2D version of the method was considered a sufficiently difficult problem to tackle for the time being. However, with the

eventual aim of tackling bigger, more interesting, real optimal design problems, the extension to 3D would likewise eventually become a necessity. Whilst all of the underlying techniques are demonstrably capable of dealing with 3D problems, it is often the case that translating any given method from 2D to 3D is not as straightforward as one might imagine. For example, for the 2D examples in this thesis Müller’s method is used to perform the integral over the level set interface which is requisite when enforcing the Dirichlet boundary condition during level set reinitialisation. Whilst Muller’s method is shown to extend to 3D problems in [144], as discussed in Section 4.3.2, in order to ensure accuracy it was found to be imperative that a set of integrals over the parts of the domain where the value of level set function is positive, are computed to a high degree of accuracy. Whilst not a particularly difficult task in 2D (requiring finding intersection points between the level set interface and the skeleton of the mesh), the proposed solution doesn’t necessarily trivially translate to 3D. How difficult such an issue would be to overcome is debatable, however, it suffices to show by a simple example that the details can quickly become difficult to deal with when attempting the task of moving from 2D to 3D, and this would therefore likely be a sizeable area of potential future work.

Another omission which should form an area of future work would be the incorporation of a hole nucleation mechanism into the methodology. Hole nucleation methods allow for the generation of new holes during level set evolution which is not possible with the methodology as it currently stands. This is important because a method which cannot generate new holes has the maximum number of holes defined by the initial topology chosen by the user, which could be insufficient to satisfy the global minimum of the optimisation problem. Whilst this was never an aim of this specific research, in order to avoid the development of methodology trapped by local minima, an investigation into hole nucleation mechanisms is another recommended area of future work.

Beyond that stated above to take the developed methodology and use it to not only solve benchmark topology optimisation problems in 2D and 3D (which of course should be computed for verification purposes), but to extend the use of the level set based topology optimisation method for a set of interesting optimal design problems would be a final recommendation from the author for possible areas of future work. Given the DG nature of the methodology, the method could be particularly well suited to model fluid problems and therefore the optimal design of structures undergoing fluid-structure interactions may be one such application, where such a methodology might prove advantageous.

Appendix A

Divergence Free Basis Functions for Müller's Method

The p^{th} order orthonormalised divergence free bases, β' , in equation (4.27) can be generated as follows

$$\beta'_{p,0} = \begin{Bmatrix} y^p \\ 0 \end{Bmatrix}, \quad \beta'_{p,k} = \begin{Bmatrix} 0 \\ x^p \end{Bmatrix},$$

where $k = p + 1$. If $p > 0$ then

$$\beta'_{p,i} = \begin{Bmatrix} x^i y^{p-i} \\ -\frac{i}{1+p-i} x^{i-1} y^{p+1-i} \end{Bmatrix}, \quad \text{for } i = 1, 2, \dots, p.$$

The total number of basis functions for a given order, p , thus is given by

$$N_I = \frac{(p+1)(p+4)}{2}. \quad (\text{A.1})$$

For example, for a second order discretisation, there would be 9 basis functions which would be ordered as follows

$$\begin{aligned} \beta'_1 &= \beta_{0,0}, & \beta'_2 &= \beta_{0,1}, \\ \beta'_3 &= \beta_{1,0}, & \beta'_4 &= \beta_{1,1}, & \beta'_5 &= \beta_{1,2}, \\ \beta'_6 &= \beta_{2,0}, & \beta'_7 &= \beta_{2,1}, & \beta'_8 &= \beta_{2,2}, & \beta'_9 &= \beta_{2,3}, \end{aligned}$$

and which could therefore be stated as

$$\begin{aligned} \beta'_1 &= \begin{Bmatrix} 1 \\ 0 \end{Bmatrix}, & \beta'_2 &= \begin{Bmatrix} 0 \\ 1 \end{Bmatrix}, \\ \beta'_3 &= \begin{Bmatrix} y \\ 0 \end{Bmatrix}, & \beta'_4 &= \begin{Bmatrix} x \\ -y \end{Bmatrix}, & \beta'_5 &= \begin{Bmatrix} 0 \\ x \end{Bmatrix}, \\ \beta'_6 &= \begin{Bmatrix} y^2 \\ 0 \end{Bmatrix}, & \beta'_7 &= \begin{Bmatrix} xy \\ -\frac{1}{2}y^2 \end{Bmatrix}, & \beta'_8 &= \begin{Bmatrix} x^2 \\ 2xy \end{Bmatrix}, & \beta'_9 &= \begin{Bmatrix} 0 \\ x^2 \end{Bmatrix}. \end{aligned}$$

Appendix B

Butcher Tableaus

As it is possible to construct a variety of Runge-Kutta methods for a given order, and there is often not a standard choice as these varieties can differ in terms of stability and accuracy in a given context, this appendix specifies the exact set of coefficients used for each order of Runge-Kutta method used in this thesis. As is explained in the main body of this document (see Section 5.1.2), as a general rule, for a spatial discretisation of maximum polynomial order p_{max} , a Runge-Kutta temporal discretisation of order $p_{max} + 1$ is adopted. Over the range of example problems presented in this thesis, the maximum polynomial order is in the range $1 \leq p_{max} \leq 8$, and as such presented below are the coefficients for Runge-Kutta methods of order $2 \leq p_{RK} \leq 8$. For discretisations of order $p_{max} = 8$, a 10th Runge-Kutta method is used with coefficients which have been computed numerically and which therefore cannot be stated as succinctly, as such the reader is referred to the paper from which these coefficients come, [199], for details. How to use these tables is described Equations (5.7) and (5.6), in Section 5.1.2.

$$\begin{array}{c|cc} 0 & & \\ 1 & 1 & \\ \hline & \frac{1}{2} & \frac{1}{2} \end{array}$$

Table B.1: Butcher Tableau for Heun's Method (RK2)

$$\begin{array}{c|ccc} 0 & & & \\ \frac{1}{3} & \frac{1}{3} & & \\ \frac{2}{3} & 0 & \frac{2}{3} & \\ \hline & \frac{1}{4} & 0 & \frac{3}{4} \end{array}$$

Table B.2: Butcher Tableau for Heun's 3rd Order Method (RK3)

0			
$\frac{1}{2}$	$\frac{1}{2}$		
$\frac{1}{2}$	0	$\frac{1}{2}$	
1	0	0	1
	$\frac{1}{6}$	$\frac{1}{3}$	$\frac{1}{3}$
	$\frac{1}{6}$		

Table B.3: Butcher Tableau for the classical 4th Order Method (RK4)

0					
$\frac{1}{4}$	$\frac{1}{4}$				
$\frac{1}{4}$	$\frac{1}{8}$	$\frac{1}{8}$			
$\frac{1}{2}$	0	0	$\frac{1}{2}$		
$\frac{3}{4}$	$\frac{3}{16}$	$-\frac{3}{8}$	$\frac{3}{8}$	$\frac{9}{16}$	
1	$-\frac{3}{7}$	$\frac{8}{7}$	$\frac{6}{7}$	$-\frac{12}{7}$	$\frac{8}{7}$
	$\frac{7}{90}$	0	$\frac{16}{45}$	$\frac{2}{15}$	$\frac{16}{45}$
	$\frac{7}{90}$				$\frac{7}{90}$

Table B.4: Butcher Tableau for a 5th order RK method (RK5) [200]

0						
$\frac{1}{3}$	$\frac{1}{3}$					
$\frac{2}{3}$	0	$\frac{2}{3}$				
$\frac{1}{3}$	$\frac{1}{12}$	$\frac{1}{3}$	$-\frac{1}{12}$			
$\frac{5}{6}$	$\frac{25}{48}$	$-\frac{55}{24}$	$\frac{35}{48}$	$\frac{15}{8}$		
$\frac{1}{6}$	$\frac{3}{20}$	$-\frac{11}{24}$	$-\frac{1}{8}$	$\frac{1}{2}$	$\frac{1}{10}$	
1	$-\frac{261}{260}$	$\frac{33}{13}$	$\frac{43}{156}$	$-\frac{118}{39}$	$\frac{32}{195}$	$\frac{80}{39}$
	$\frac{13}{200}$	0	$\frac{11}{40}$	$\frac{11}{40}$	$\frac{4}{25}$	$\frac{4}{25}$
	$\frac{13}{200}$					$\frac{13}{200}$

Table B.5: Butcher Tableau for a 7 stage 6th Order Runge-Kutta Method (RK6) [200]

0							
$\frac{1}{6}$	$\frac{1}{6}$						
$\frac{1}{3}$	0	$\frac{1}{3}$					
$\frac{1}{2}$	$\frac{1}{8}$	0	$\frac{3}{8}$				
$\frac{2}{11}$	$\frac{148}{1331}$	0	$\frac{8}{71}$	$-\frac{4}{95}$			
$\frac{2}{3}$	$-\frac{133}{80}$	0	$-\frac{170}{27}$	$\frac{317}{134}$	$\frac{795}{127}$		
$\frac{6}{7}$	$\frac{38}{37}$	0	$\frac{449}{124}$	$-\frac{170}{149}$	$-\frac{349}{113}$	$\frac{81}{157}$	
0	$\frac{5}{154}$	0	0	$\frac{13}{73}$	$-\frac{13}{146}$	$-\frac{12}{73}$	$\frac{3}{70}$
1	$-\frac{113}{32}$	0	$-\frac{195}{22}$	$\frac{32}{7}$	$\frac{1247}{152}$	$-\frac{215}{151}$	$\frac{19}{26}$
	0	0	0	$\frac{32}{105}$	$\frac{198}{703}$	$\frac{43}{453}$	$\frac{123}{548}$
							$\frac{77}{1440}$
							$\frac{11}{270}$

Table B.6: Butcher Tableau for a 9 stage 7th Order Runge-Kutta Method (RK7) [200]

References

- [1] Y. M. Xie and G. P. Steven, “A simple evolutionary procedure for structural optimization,” *Computers & Structures*, vol. 49, pp. 885–896, 1993.
- [2] M. P. Bendsøe and N. Kikuchi, “Generating optimal topologies in structural design using a homogenization method,” *Computer Methods in Applied Mechanics and Engineering*, vol. 71, pp. 197–224, 1988.
- [3] M. P. Bendsøe, “Optimal shape design as a material distribution problem,” *Structural optimization*, vol. 1, pp. 193–202, 1989.
- [4] C. Fleury, “Shape Optimal Design by the Convex Linearization Method,” in *The Optimum Shape: Automated Structural Design* (J. A. Bennett and M. E. Botkin, eds.), General Motors Research Laboratories Symposia Series, pp. 297–326, Boston, MA: Springer US, 1986.
- [5] M. H. Imam, “Three-dimensional shape optimization,” *International Journal for Numerical Methods in Engineering*, vol. 18, no. 5, pp. 661–673, 1982.
- [6] M. E. Botkin, “Shape Optimization of Plate and Shell Structures,” *AIAA Journal*, vol. 20, no. 2, pp. 268–273, 1982.
- [7] S. Osher and J. A. Sethian, “Fronts propagating with curvature-dependent speed: Algorithms based on Hamilton-Jacobi formulations,” *Journal of Computational Physics*, vol. 79, pp. 12–49, 1988.
- [8] S. Osher, and F. Santosa, “Level Set Methods for Optimization Problems Involving Geometry and Constraints: I. Frequencies of a Two-Density Inhomogeneous Drum,” *Journal of Computational Physics*, vol. 171, no. 1, pp. 272–288, 2001.
- [9] G. Allaire, F. Jouve, and A.-M. Toader, “Structural Optimization Using Sensitivity Analysis and a Level-Set Method,” *Journal of Computational Physics* vol. 194, no. 1, pp. 363–393, 2004.
- [10] F. Gibou, R. Fedkiw, and S. Osher, “A review of level-set methods and some recent applications,” *Journal of Computational Physics*, vol. 353, pp. 82–109, 2018.
- [11] B. Bourdin, and A. Chambolle, “Design-Dependent Loads in Topology Optimization,” *ESAIM: Control, Optimisation and Calculus of Variations*, vol. 9, pp. 19–48, 2003.

- [12] A. Myslinski, and K. Koniarski, “Hybrid level set phase field method for topology optimization of contact problems,” *Mathematica Bohemica*, vol. 140, no. 4, pp. 419–435, 2015.
- [13] O. Sigmund, and K. Maute, “Topology Optimization Approaches: A Comparative Review,” *Structural and Multidisciplinary Optimization*, vol. 48, no. 6, pp. 1031–1055, 2013.
- [14] E. Tyflopoulos, F. Tollnes, M. Steinert, and A. Olsen, “State of the Art of Generative Design and Topology Optimization and Potential Research Needs,” DS 91: Proceedings of NordDesign 2018, Linkping, Sweden, 2018.
- [15] P. Šolin. *Partial differential equations and the finite element method*. Vol. 73. John Wiley & Sons, 2005.
- [16] C. H. Villanueva and K. Maute, “Density and level set-XFEM schemes for topology optimization of 3-D structures,” *Computational Mechanics*, vol. 54, pp. 133–150, 2014.
- [17] N. Jenkins and K. Maute, “Level set topology optimization of stationary fluid-structure interaction problems,” *Structural and Multidisciplinary Optimization*, vol. 52, pp. 179–195, 2015.
- [18] P. Liu, Y. Luo, and Z. Kang, “Multi-material topology optimization considering interface behavior via XFEM and level set method,” *Computer Methods in Applied Mechanics and Engineering*, vol. 308, pp. 113–133, 2016.
- [19] S. Yamasaki, T. Yamada, and T. Matsumoto, “An immersed boundary element method for level-set based topology optimization,” *International Journal for Numerical Methods in Engineering*, vol. 93, no. 9, pp. 960–988, 2013.
- [20] B. Ullah, *Structural Topology Optimisation Based on the Boundary Element and Level Set Methods*. Doctoral Dissertation, Durham University, 2014.
- [21] M. Otomori, T. Yamada, K. Izui, and S. Nishiwaki, “Level set-based topology optimisation of a compliant mechanism design using mathematical programming,” *Mechanical Sciences*, vol. 2, pp. 91–98, 2011.
- [22] D. L. Chopp, “Computing Minimal Surfaces via Level Set Curvature Flow,” *Journal of Computational Physics*, vol. 106, pp. 77–91, 1993.
- [23] S. Osher, and R. P. Fedkiw, *Level Set Methods and Dynamic Implicit Surfaces*, Applied Mathematical Sciences, v. 153. New York: Springer, 2003.
- [24] S. Osher and J. Yan, “Discontinuous Galerkin Level Set Method for Interface Capturing,” *UCLA Report*, (2005).
- [25] R. Guo and F. Filbet, “A p-Adaptive Local Discontinuous Galerkin Level Set Method for Willmore Flow,” *Journal of Scientific Computing*, vol. 76, pp. 1148–1167, 2018.

- [26] J. Strain, “Tree Methods for Moving Interfaces,” *Journal of Computational Physics*, vol. 151, pp. 616–648, 1999.
- [27] F. Losasso, R. Fedkiw, and S. Osher, “Spatially adaptive techniques for level set methods and incompressible flow,” *Computers & Fluids*, vol. 35, pp. 995–1010, 2006.
- [28] V. Ducrot and P. Frey, “Anisotropic Level Set Adaptation for Accurate Interface Capturing,” *Proceedings of the 17th International Meshing Roundtable* (R. V. Garimella, ed.), pp. 159–176, Berlin, Heidelberg: Springer Berlin Heidelberg, 2008.
- [29] N. R. Morgan and J. I. Waltz, “3D level set methods for evolving fronts on tetrahedral meshes with adaptive mesh refinement,” *Journal of Computational Physics*, vol. 336, pp. 492–512, 2017.
- [30] L. C. Ngo and H. G. Choi, “A multi-level adaptive mesh refinement method for level set simulations of multiphase flow on unstructured meshes,” *International Journal for Numerical Methods in Engineering*, vol. 110, no. 10, pp. 947–971, 2017.
- [31] T. Wang, H. Li, Y. Feng, and D. Shi, “A coupled volume-of-fluid and level set (VOSET) method on dynamically adaptive quadtree grids,” *International Journal of Heat and Mass Transfer*, vol. 67, pp. 70–73, 2013.
- [32] J. Baiges, J. Martínez-Frutos, D. Herrero-Pérez, F. Otero, and A. Ferrer, “Large-scale stochastic topology optimization using adaptive mesh refinement and coarsening through a two-level parallelization scheme,” *Computer Methods in Applied Mechanics and Engineering*, vol. 343, pp. 186–206, 2019.
- [33] E. Bezchlebová, V. Dolejší, and M. Feistauer, “Discontinuous Galerkin method for the solution of a transport level-set problem,” *Computers & Mathematics with Applications*, vol. 72, pp. 455–480, 2016.
- [34] J. Luo, Z. Luo, L. Chen, L. Tong, and M. Y. Wang, “A semi-implicit level set method for structural shape and topology optimization,” *Journal of Computational Physics*, vol. 227, pp. 5561–5581, 2008.
- [35] W. H. Reed and T. R. Hill, “Triangular mesh methods for the neutron transport equation,” No. LA-UR-73-479; CONF-730414-2, Los Alamos Scientific Lab, 1973.
- [36] P. LeSaint and P. Raviart, “On a Finite Element Method for Solving the Neutron Transport Problem,” *Mathematical aspects of finite elements in partial differential equations*, pp. 89–123, 1974.
- [37] J. Nitsche, “Über ein Variationsprinzip zur Lösung von Dirichlet-Problemen bei Verwendung von Teilräumen, die keinen Randbedingungen unterworfen sind,” *Abhandlungen aus dem Mathematischen Seminar der Universität Hamburg*, vol. 36, pp. 9–15, 1971.
- [38] I. Babuška, “The Finite Element Method with Penalty,” *Mathematics of Computation*, vol. 27, no. 122, pp. 221–228, 1973.

- [39] I. Babuška and M. Zlamal, “Nonconforming Elements in the Finite Element Method with Penalty,” *SIAM Journal on Numerical Analysis*, vol. 10, no. 5, pp. 863–875, 1973.
- [40] G. A. Baker, “Finite Element Methods for Elliptic Equations Using Nonconforming Elements,” *Mathematics of Computation*, vol. 31, no. 137, pp. 45–49, 1977.
- [41] J. Douglas and T. Dupont, “Interior Penalty Procedures for Elliptic and Parabolic Galerkin Methods,” in *Computing Methods in Applied Sciences* (R. Glowinski and J. L. Lions, eds.), Lecture Notes in Physics, pp. 207–216, Springer Berlin Heidelberg, 1976.
- [42] M. F. Wheeler, “An Elliptic Collocation-Finite Element Method with Interior Penalties,” *SIAM Journal on Numerical Analysis*, vol. 15, no. 1, pp. 152–161, 1978.
- [43] D. N. Arnold, *An Interior Penalty Finite Element Method with Discontinuous Elements*, Doctoral Dissertation, University of Chicago, 1979.
- [44] G. Chavent and G. Salzano, “A finite-element method for the 1-D water flooding problem with gravity,” *Journal of Computational Physics*, vol. 45, pp. 307–344, 1982.
- [45] D. N. Arnold, F. Brezzi, B. Cockburn, and L. D. Marini, “Unified Analysis of Discontinuous Galerkin Methods for Elliptic Problems,” *SIAM Journal on Numerical Analysis*, vol. 39, no. 5, pp. 1749–1779, 2002.
- [46] S. K. Godunov and I. Bohachevsky, “Finite difference method for numerical computation of discontinuous solutions of the equations of fluid dynamics,” *Matematicheskii Sbornik*, vol. 47(89), no. 3, pp. 271–306, 1959.
- [47] R. Courant, E. Isaacson, and M. Rees, “On the solution of nonlinear hyperbolic differential equations by finite differences,” *Communications on Pure and Applied Mathematics*, vol. 5, no. 3, pp. 243–255, 1952.
- [48] P. Lax and B. Wendroff, “Systems of conservation laws,” *Communications on Pure and Applied Mathematics*, vol. 13, no. 2, pp. 217–237, 1960.
- [49] B. van Leer, “Towards the ultimate conservative difference scheme I. The quest of monotonicity,” in *Proceedings of the Third International Conference on Numerical Methods in Fluid Mechanics* (H. Cabannes and R. Temam, eds.), Lecture Notes in Physics, pp. 163–168, Springer Berlin Heidelberg, 1973.
- [50] B. van Leer, “Towards the ultimate conservative difference scheme. II. Monotonicity and conservation combined in a second-order scheme,” *Journal of Computational Physics*, vol. 14, pp. 361–370, 1974.
- [51] B. Van Leer, “Towards the ultimate conservative difference scheme III. Upstream-centered finite-difference schemes for ideal compressible flow,” *Journal of Computational Physics*, vol. 23, pp. 263–275, 1977.

- [52] B. Van Leer, “Towards the ultimate conservative difference scheme. IV. A new approach to numerical convection,” *Journal of Computational Physics*, vol. 23, pp. 276–299, 1977.
- [53] B. van Leer, “Towards the ultimate conservative difference scheme. V. A second-order sequel to Godunov’s method,” *Journal of Computational Physics*, vol. 32, pp. 101–136, 1979.
- [54] P. Colella and P. R. Woodward, “The Piecewise Parabolic Method (PPM) for gas-dynamical simulations,” *Journal of Computational Physics*, vol. 54, pp. 174–201, 1984.
- [55] P. K. Sweby, “High Resolution Schemes Using Flux Limiters for Hyperbolic Conservation Laws,” *SIAM journal on numerical analysis*, vol. 21, no. 5, pp. 995–1011, 1984.
- [56] J. B. Bell, P. Colella, and J. A. Trangenstein, “Higher order Godunov methods for general systems of hyperbolic conservation laws,” *Journal of Computational Physics*, vol. 82, pp. 362–397, 1989.
- [57] P. L. Roe, “Approximate Riemann solvers, parameter vectors, and difference schemes,” *Journal of Computational Physics*, vol. 43, pp. 357–372, 1981.
- [58] B. Engquist and S. Osher, “One-Sided Difference Approximations for Nonlinear Conservation Laws,” *Mathematics of Computation*, vol. 36, no. 154, pp. 321–351, 1981.
- [59] A. Harten, P. D. Lax, and B. van Leer, “On Upstream Differencing and Godunov-Type Schemes for Hyperbolic Conservation Laws,” *SIAM Review*, vol. 25, pp. 35–61, 1983.
- [60] A. Harten and S. Osher, “Uniformly High-Order Accurate Nonoscillatory Schemes. I,” *SIAM Journal on Numerical Analysis*, vol. 24, no. 2, pp. 279–309, 1987.
- [61] A. Harten, B. Engquist, S. Osher, and S. Chakravarthy, “Uniformly High Order Accurate Essentially Non-Oscillatory Schemes, III,” *Journal of Computational Physics*, vol. 71, no. 2, pp. 231–303, 1987.
- [62] C. W. Shu, “TVB Uniformly High-Order Schemes for Conservation Laws,” *Mathematics of Computation*, vol. 49, no. 179, pp.105–121, 1987.
- [63] R. J. LeVeque, *Finite Volume Methods for Hyperbolic Problems*. Cambridge University Press, 2002.
- [64] B. Cockburn and C.-W. Shu, “The Runge-Kutta local projection P1-discontinuous-Galerkin finite element method for scalar conservation laws,” *ESAIM: Mathematical Modelling and Numerical Analysis*, vol. 25, no. 3, pp. 337–361, 1991.
- [65] B. Cockburn and C.-W. Shu, “TVB Runge-Kutta local projection discontinuous Galerkin finite element method for conservation laws. II. General framework,” *Mathematics of Computation*, vol. 52, no. 186, pp. 411–435, 1989.

- [66] B. Cockburn, S.-Y. Lin, and C.-W. Shu, “TVB Runge-Kutta Local Projection Discontinuous Galerkin Finite Element Method for Conservation Laws III: One-Dimensional Systems,” *Journal of Computational Physics*, vol. 84, no. 1, pp. 90–133, 1989.
- [67] B. Cockburn, S. Hou, and C.-W. Shu, “The Runge-Kutta Local Projection Discontinuous Galerkin Finite Element Method for Conservation Laws. IV: The Multidimensional Case,” *Mathematics of Computation*, vol. 54, no. 190, pp. 545–581, 1990.
- [68] B. Cockburn and C.-W. Shu, “The Runge-Kutta Discontinuous Galerkin Method for Conservation Laws V: Multidimensional Systems,” *Journal of Computational Physics*, vol. 141, pp. 199–224, 1998.
- [69] J. Oden, I. Babuška, and C. E. Baumann, “A Discontinuous hp Finite Element Method for Diffusion Problems,” *Journal of Computational Physics*, vol. 146, pp. 491–519, 1998.
- [70] F. Bassi and S. Rebay, “A High-Order Accurate Discontinuous Finite Element Method for the Numerical Solution of the Compressible Navier-Stokes Equations,” *Journal of Computational Physics*, vol. 131, pp. 267–279, 1997.
- [71] D. Di Pietro, and A. Ern, *Mathematical Aspects of Discontinuous Galerkin Methods*. Mathematiques et Applications. Berlin Heidelberg: Springer-Verlag, 2012.
- [72] P. Šolin, K. Segeth, and I. Doležel, *Higher-Order Finite Element Methods*. Studies in Advanced Mathematics, Boca Raton, FL: Chapman & Hall/CRC, 2004.
- [73] X. Feng, O. Karakashian, and Y. Xing, *Recent Developments in Discontinuous Galerkin Finite Element Methods for Partial Differential Equations: 2012 John H Barrett Memorial Lectures*. Springer Science & Business Media, 2013.
- [74] P. Houston, J. Robson, and E. Süli, “Discontinuous Galerkin finite element approximation of quasilinear elliptic boundary value problems I: The scalar case,” *IMA Journal of Numerical Analysis*, vol. 25, pp. 726–749, 2005.
- [75] D. N. Arnold, “An Interior Penalty Finite Element Method with Discontinuous Elements,” *SIAM Journal on Numerical Analysis*, vol. 19, no. 4, pp. 742–760, 1982.
- [76] B. Rivière, M. F. Wheeler, and V. Girault, “Improved energy estimates for interior penalty, constrained and discontinuous Galerkin methods for elliptic problems. Part I,” *Computational Geosciences*, vol. 3, pp. 337–360, 1999.
- [77] B. Rivière and M. F. Wheeler, “A Discontinuous Galerkin Method Applied to Nonlinear Parabolic Equations,” in *Discontinuous Galerkin Methods* (M. Griebel, D. E. Keyes, R. M. Nieminen, D. Roose, T. Schlick, B. Cockburn, G. E. Karniadakis, and C.-W. Shu, eds.), vol. 11, pp. 231–244, Berlin, Heidelberg: Springer Berlin Heidelberg, 2000.
- [78] T. Gudi and A. K. Pani, “Discontinuous Galerkin Methods for Quasi-Linear Elliptic Problems of Nonmonotone Type,” *SIAM Journal on Numerical Analysis*, vol. 45, no. 1, pp. 163–192, 2007.

- [79] L. Song, “Fully discrete interior penalty discontinuous Galerkin methods for nonlinear parabolic equations,” *Numerical Methods for Partial Differential Equations*, vol. 28, pp. 288–311, 2012.
- [80] L. Song, G.-M. Gie, and M.-C. Shiue, “Interior penalty discontinuous Galerkin methods with implicit time-integration techniques for nonlinear parabolic equations,” *Numerical Methods for Partial Differential Equations*, vol. 29, pp. 1341–1366, 2013.
- [81] V. Dolejší, “Analysis and application of the IIPG method to quasilinear nonstationary convection–diffusion problems,” *Journal of Computational and Applied Mathematics*, vol. 222, pp. 251–273, 2008.
- [82] P. Castillo, “Performance of discontinuous Galerkin methods for elliptic PDEs,” *SIAM Journal on Scientific Computing*, vol. 24, no. 2, pp. 524–547, 2002.
- [83] B. Cockburn and C.-W. Shu, “The local discontinuous Galerkin method for time-dependent convection-diffusion systems,” *SIAM Journal on Numerical Analysis*, vol. 35, no. 6, pp. 2440–2463, 1998.
- [84] E. H. Georgoulis, “Discontinuous Galerkin Methods for Linear Problems: An Introduction,” in *Approximation Algorithms for Complex Systems*, edited by E.H. Georgoulis, A. Iske, and J. Levesley, vol. 3, pp. 91–126. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011.
- [85] T. Warburton, and J. S. Hesthaven, “On the Constants in Hp–Finite Element Trace Inverse Inequalities,” *Computer Methods in Applied Mechanics and Engineering*, vol. 192, no. 25, pp. 2765–2773, 2003.
- [86] K. Shahbazi, “An Explicit Expression for the Penalty Parameter of the Interior Penalty Method,” *Journal of Computational Physics*, vol. 205, no. 2, pp. 401–407, 2005.
- [87] Y. Epshteyn, and B. Rivière. “Estimation of Penalty Parameters for Symmetric Interior Penalty Galerkin Methods,” *Journal of Computational and Applied Mathematics*, vol. 206, no. 2, pp. 843–872, 2007.
- [88] E. Burman, and A. Ern, “Continuous Interior Penalty Hp–Finite Element Methods for Advection and Advection–Diffusion Equations,” *Mathematics of Computation*, vol. 76, no. 259, pp. 1119–1140, 2007.
- [89] K. Hillewaert, *Development of the Discontinuous Galerkin Method for High-Resolution, Large Scale CFD and Acoustics in Industrial Geometries*. Presses Universitaires de Louvain, 2013.
- [90] I. Mozolevski, and P. R. Bösing. “Sharp Expressions for the Stabilization Parameters in Symmetric Interior–Penalty Discontinuous Galerkin Finite Element Approximations of Fourth–Order Elliptic Problems,” *Computational Methods in Applied Mathematics*, vol. 7, no. 4, pp. 365–375, 2007.

- [91] D. Sármany, F. Izsák, and J. J. W. van der Vegt, “Optimal Penalty Parameters for Symmetric Discontinuous Galerkin Discretisations of the Time–Harmonic Maxwell Equations,” *Journal of Scientific Computing*, vol. 44, no. 3, pp. 219–254, 2010.
- [92] A. Richter, E. Brussies, and J. Stiller, “Influence of Penalization and Boundary Treatment on the Stability and Accuracy of High–Order Discontinuous Galerkin Schemes for the Compressible Navier–Stokes Equations,” *Journal of Computational Acoustics*, vol. 21, no. 1, 2012.
- [93] A. R. Owens, J. Kópházi, and M. D. Eaton, “Optimal Trace Inequality Constants for Interior Penalty Discontinuous Galerkin Discretisations of Elliptic Operators Using Arbitrary Elements with Non–Constant Jacobians,” *Journal of Computational Physics*, vol. 350, pp. 847–870, 2017.
- [94] C. Johnson and J. Pitkäranta, “An Analysis of the Discontinuous Galerkin Method for a Scalar Hyperbolic Equation,” *Computer methods in applied mechanics and engineering*, vol. 45, no. 1–3, pp. 285–312, 1984.
- [95] T. Peterson, “A Note on the Convergence of the Discontinuous Galerkin Method for a Scalar Hyperbolic Equation,” *SIAM Journal on Numerical Analysis*, vol. 28, pp. 133–140, 1991.
- [96] G. R. Richter, “An Optimal-Order Error Estimate for the Discontinuous Galerkin Method,” *Mathematics of Computation*, vol. 50, no. 181, pp. 75–88, 1988.
- [97] B. Cockburn, B. Dong, and J. Guzmán, “Optimal Convergence of the Original DG Method for the Transport-Reaction Equation on Special Meshes,” *SIAM Journal on Numerical Analysis* vol. 46, no. 3, pp. 1250-1265, 2008.
- [98] E. Deriaz, and P. Haldenwang, “Von Neumann Stability Analysis of the Upwind Finite Difference Schemes,” 2013.
- [99] G. Chavent and B. Cockburn, “The local projection P^0 - P^1 -discontinuous-Galerkin finite element method for scalar conservation laws,” *ESAIM: Mathematical Modelling and Numerical Analysis*, vol. 23, no. 4, pp. 565–592, 1989.
- [100] G. S. Jiang and C.-W. Shu, “On a cell entropy inequality for discontinuous Galerkin methods,” *Mathematics of Computation*, vol. 62, no. 206, pp. 531–538, 1994.
- [101] Q. Zhang and C.-W. Shu, “Error Estimates to Smooth Solutions of Runge-Kutta Discontinuous Galerkin Methods for Scalar Conservation Laws,” *SIAM Journal on Numerical Analysis*, vol. 42, no. 2, pp. 641–666, 2005.
- [102] Q. Zhang and C.-W. Shu, “Stability Analysis and A Priori Error Estimates of the Third Order Explicit Runge–Kutta Discontinuous Galerkin Method for Scalar Conservation Laws,” *SIAM Journal on Numerical Analysis*, vol. 48, pp. 1038–1063, 2010.

- [103] B. Cockburn, “Discontinuous Galerkin Methods,” *Journal of Applied Mathematics and Mechanics* vol. 83, no. 11, pp. 731-754, 2003.
- [104] R. Hartmann, and T. Leicht, *Higher Order and Adaptive DG Methods for Compressible Flows*, Von Karman Institute for Fluid Dynamics, 2014.
- [105] B. Cockburn and C.-W. Shu, “Runge–Kutta Discontinuous Galerkin Methods for Convection-Dominated Problems,” *Journal of Scientific Computing*, vol. 16, pp. 173–261, 2001.
- [106] V. Wheatley, H. Kumar, and P. Huguenot, “On the Role of Riemann Solvers in Discontinuous Galerkin Methods for Magnetohydrodynamics,” *Journal of Computational Physics*, vol. 229, no. 3, pp.660–680, 2010.
- [107] J. A. Sethian, and A. Wiegmann, “Structural Boundary Design via Level Set and Immersed Interface Methods,” *Journal of Computational Physics*, vol. 163, no. 2, pp. 489–528, 2000.
- [108] W. Mulder, S. Osher, and J. A. Sethian, “Computing interface motion in compressible gas dynamics,” *Journal of Computational Physics*, vol. 100, pp. 209–228, 1992.
- [109] J. A. Sethian, *Level Set Methods and Fast Marching Methods: Evolving Interfaces in Computational Geometry, Fluid Mechanics, Computer Vision, and Materials Science*. No. 3 in Cambridge Monographs on Applied and Computational Mathematics, Cambridge University Press, 2nd ed., 1999.
- [110] S. Giani and E. J. C. Hall, “An a posteriori error estimator for hp-adaptive discontinuous galerkin methods for elliptic eigenvalue problems,” *Mathematical Models and Methods in Applied Sciences*, vol. 22, p. 1250030, 2012.
- [111] C. Hu and C. Shu, “A Discontinuous Galerkin Finite Element Method for Hamilton–Jacobi Equations,” *SIAM Journal on Scientific Computing*, vol. 21, pp. 666–690, 1999.
- [112] M. Sussman and M. Y. Hussaini, “A Discontinuous Spectral Element Method for the Level Set Equation,” *Journal of Scientific Computing*, vol. 19, pp. 479–500, 2003.
- [113] E. Marchandise, J.-F. Remacle, and N. Chevaugeon, “A quadrature-free discontinuous Galerkin method for the level set equation,” *Journal of Computational Physics*, vol. 212, pp. 338–357, 2006.
- [114] J. Naber and B. Koren, “A Runge-Kutta Discontinuous-Galerkin Level-Set Method for Unsteady Compressible Two-Fluid Flow,” *ECCOMAS CFD 2006: Proceedings of the European Conference on Computational Fluid Dynamics, Egmond aan Zee, The Netherlands, September 5-8, 2006*.
- [115] Z. Jibben and M. Herrmann, “A Runge-Kutta discontinuous Galerkin conservative level set method,” *Center for Turbulence Research, Proceedings of the Summer Program*, pp. 305–314, 2012.

- [116] M. Owkes and O. Desjardins, “A discontinuous Galerkin conservative level set scheme for interface capturing in multiphase flows,” *Journal of Computational Physics*, vol. 249, pp. 275–302, 2013.
- [117] F. Heimann, C. Engwer, O. Ippisch, and P. Bastian, “An unfitted interior penalty discontinuous Galerkin method for incompressible Navier-Stokes two-phase flow,” *International Journal for Numerical Methods in Fluids*, vol. 71, no. 3, pp. 269–293, 2013.
- [118] A. Karakus, T. Warburton, M. H. Aksel, and C. Sert, “An adaptive fully discontinuous Galerkin level set method for incompressible multiphase flows,” *International Journal of Numerical Methods for Heat & Fluid Flow*, vol. 28, pp. 1256–1278, 2018.
- [119] R. Saye, “Implicit mesh discontinuous Galerkin methods and interfacial gauge methods for high-order accurate interface dynamics, with applications to surface tension dynamics, rigid body fluid-structure interaction, and free surface flow: Part I,” *Journal of Computational Physics*, vol. 344, pp. 647–682, 2017.
- [120] R. Mousavi, F. Kummer, M. Oberlack, and P. F. Pelz, “Level set method for simulating the interface kinematics: application of a discontinuous Galerkin method,” *Proceedings of ECCOMAS congress*, 2016.
- [121] J. Zhang and P. Yue, “A high-order and interface-preserving discontinuous Galerkin method for level-set reinitialization,” *Journal of Computational Physics*, vol. 378, pp. 634–664, 2019.
- [122] F. Pochet, K. Hillewaert, P. Geuzaine, J.-F. Remacle, and É. Marchandise, “A 3D strongly coupled implicit discontinuous Galerkin level set-based method for modeling two-phase flows,” *Computers & Fluids*, vol. 87, pp. 144–155, 2013.
- [123] T. Utz, *Level set methods for high-order unfitted discontinuous Galerkin schemes*, Doctoral Dissertation, Technische Universität Darmstadt, 2018.
- [124] E. Loch, *The Level Set Method for Capturing Interfaces with Applications in Two-Phase Flow Problems*, Doctoral Dissertation, RWTH Aachen University, 2013.
- [125] M. Ta, F. Pigeonneau, and P. Saramito, “An implicit high order discontinuous Galerkin level set method for two-phase flow problems,” *ICMF-2016: 9th International Conference on Multiphase Flow*, 2016.
- [126] W. Sollie, O. Bokhove, and J. van der Vegt, “Space-time discontinuous Galerkin finite element method for two-fluid flows,” *Journal of Computational Physics*, vol. 230, no. 3, pp. 789–817, 2011.
- [127] T. Adams, S. Giani, and W. M. Coombs, “A high-order elliptic PDE based level set reinitialisation method using a discontinuous Galerkin discretisation,” *Journal of Computational Physics*, vol. 379, pp. 373–391, 2019.

- [128] T. Adams, N. McLeish, S. Giani, and W. M. Coombs, “A parabolic level set reinitialisation method using a discontinuous Galerkin discretisation,” *Computers & Mathematics with Applications*, 2019.
- [129] M. Sussman, P. Smereka, and S. Osher, “A Level Set Approach for Computing Solutions to Incompressible Two-Phase Flow,” *Journal of Computational Physics*, vol. 114, pp. 146–159, 1994.
- [130] M. Sussman and E. Fatemi, “An Efficient, Interface-Preserving Level Set Redistancing Algorithm and Its Application to Interfacial Incompressible Fluid Flow,” *SIAM Journal on Scientific Computing*, vol. 20, pp. 1165–1191, 1999.
- [131] R. Mousavi, *Level Set Method for Simulating the Dynamics of the Fluid-Fluid Interfaces: Application of a Discontinuous Galerkin Method*, Doctoral Dissertation, Technische Universität Darmstadt, 2014.
- [132] A. Karakus, T. Warburton, M. H. Aksel, and C. Sert, “A GPU accelerated level set reinitialization for an adaptive discontinuous Galerkin method,” *Computers & Mathematics with Applications*, vol. 72, pp. 755–767, 2016.
- [133] J. Gomes and O. Faugeras, “Reconciling Distance Functions and Level Sets,” *Journal of Visual Communication and Image Representation*, vol. 11, pp. 209–223, 2000.
- [134] M. Weber, A. Blake, and R. Cipolla, “Sparse finite elements for geodesic contours with level-sets,” *Computer Vision-ECCV 2004*, pp. 391–404, 2004.
- [135] C. Li, C. Xu, C. Gui, and M. D. Fox, “Level set evolution without re-initialization: A new variational formulation,” in *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference On*, vol. 1, pp. 430–436, IEEE, 2005.
- [136] C. Li, C. Xu, C. Gui, and M. D. Fox, “Distance Regularized Level Set Evolution and Its Application to Image Segmentation,” *IEEE Transactions on Image Processing*, vol. 19, pp. 3243–3254, 2010.
- [137] C. Basting and D. Kuzmin, “A minimization-based finite element formulation for interface-preserving level set reinitialization,” *Computing*, vol. 95, pp. 13–25, 2013.
- [138] N. Parolini, *Computational fluid dynamics for naval engineering problems*, Doctoral Dissertation, École Polytechnique Fédérale de Lausanne, 2004.
- [139] T. Utz, F. Kummer, and M. Oberlack, “Interface-preserving level-set reinitialization for DG-FEM,” *International Journal for Numerical Methods in Fluids*, vol. 84, pp. 183–198, 2017.
- [140] F. Santambrogio, “Euclidean, Metric, and Wasserstein Gradient Flows: An overview,” *Bulletin of Mathematical Sciences*, vol. 7, no. 1, pp.87–154, 2017.

- [141] P. O. Persson, *Mesh Generation for Implicit Geometries*. Doctoral Dissertation, Massachusetts Institute of Technology, 2005.
- [142] C. S. Peskin, “Flow patterns around heart valves: A numerical method,” *Journal of Computational Physics*, vol. 10, pp. 252–271, 1972.
- [143] R. Saye, “High-Order Quadrature Methods for Implicitly Defined Surfaces and Volumes in Hyperrectangles,” *SIAM Journal on Scientific Computing*, vol. 37, pp. 993–1019, 2015.
- [144] B. Müller, F. Kummer, and M. Oberlack, “Highly accurate surface and volume integration on implicit domains by means of moment-fitting,” *International Journal for Numerical Methods in Engineering*, vol. 96, pp. 512–528, 2013.
- [145] B. Engquist, A. K. Tornberg, and R. Tsai, “Discretization of Dirac delta functions in level set methods,” *Journal of Computational Physics*, vol. 207, pp. 28–51, 2005.
- [146] J. Bremer, Z. Gimbutas, and V. Rokhlin, “A Nonlinear Optimization Procedure for Generalized Gaussian Quadratures,” *SIAM Journal on Scientific Computing*, vol. 32, pp. 1761–1788, 2010.
- [147] I. Babuška, “The finite element method with Lagrangian multipliers,” *Numerische Mathematik*, vol. 20, no. 3, pp. 179–192, 1973.
- [148] G. Brandstetter and S. Govindjee, “A high-order immersed boundary discontinuous-Galerkin method for Poisson’s equation with discontinuous coefficients and singular sources: A high-order immersed boundary DG method,” *International Journal for Numerical Methods in Engineering*, vol. 101, pp. 847–869, 2015.
- [149] I. Babuška, U. Banerjee, and J. E. Osborn, “Survey of meshless and generalized finite element methods: A unified approach,” *Acta Numerica*, vol. 12, pp. 1–125, 2003.
- [150] A. J. Lew and G. C. Buscaglia, “A discontinuous-Galerkin-based immersed boundary method,” *International Journal for Numerical Methods in Engineering*, vol. 76, pp. 427–454, 2008.
- [151] H. J. Barbosa and T. J. Hughes, “Boundary Lagrange multipliers in finite element methods: Error analysis in natural norms,” *Numerische Mathematik*, vol. 62, no. 1, pp. 1–15, 1992.
- [152] Z. Chen and H. Chen, “Pointwise Error Estimates of Discontinuous Galerkin Methods with Penalty for Second-Order Elliptic Problems,” *SIAM Journal on Numerical Analysis*, vol. 42, no. 3, pp. 1146–1166, 2005.
- [153] V. Dolejší, “Semi-implicit Interior Penalty Discontinuous Galerkin Methods for Viscous Compressible Flows,” *Communications in Computational Physics*, vol. 4, no. 2, pp. 231–274, 2008.

- [154] M. G. Crandall and P. L. Lions, “Two Approximations of Solutions of Hamilton-Jacobi Equations,” *Mathematics of computation*, vol. 43, no. 167, pp. 1–19, 1984.
- [155] B. van Leer, J. Thomas, P. Roe, and R. Newsome, “A comparison of numerical flux formulas for the Euler and Navier-Stokes equations,” *Proceedings of the 8th Computational Fluid Dynamics Conference*, 1987.
- [156] H. F. Walker and P. Ni, “Anderson Acceleration for Fixed-Point Iterations,” *SIAM Journal on Numerical Analysis*, vol. 49, pp. 1715–1735, 2011.
- [157] H. F. Walker, “Anderson Acceleration: Algorithms and Implementations”.
- [158] A. Ovsyannikov, V. Sabel’nikov, and M. Gorokhovski, “A new level set equation and its numerical assessments,” *Center for Turbulence Research, Proceedings of the Summer Program*, 2012.
- [159] H. Wei, L. Chen, Y. Huang, and B. Zheng, “Adaptive Mesh Refinement and Superconvergence for Two-Dimensional Interface Problems,” *SIAM Journal on Scientific Computing*, vol. 36, no. 4, pp. 1478–1499, 2014.
- [160] X. Feng and H. J. Wu, “A Posteriori Error Estimates and an Adaptive Finite Element Method for the Allen-Cahn Equation and the Mean Curvature Flow,” *Journal of Scientific Computing*, vol. 24, pp. 121–146, 2005.
- [161] Z. Chen, W. U. Zedong, and Y. Xiao, “An adaptive immersed finite element method with arbitrary Lagrangian-Eulerian scheme for parabolic equations in time variable domains,” *International Journal of Numerical Analysis and Modeling*, vol. 12, no. 3, pp. 567–591, 2015.
- [162] D. Kolomenskiy, J. C. Nave, and K. Schneider, “Adaptive Gradient-Augmented Level Set Method with Multiresolution Error Estimation,” *Journal of Scientific Computing*, vol. 66, pp. 116–140, 2016.
- [163] R. F. Warming and R. M. Beam, “Discrete Multiresolution Analysis Using Hermite Interpolation: Biorthogonal Multiwavelets,” *SIAM Journal on Scientific Computing*, vol. 22, pp. 1269–1317, 2000.
- [164] P. Houston, B. Senior, and E. Suli, “Sobolev Regularity Estimation for hp-Adaptive Finite Element Methods,” *Numerical mathematics and advanced applications*, pp. 631–656, 2003.
- [165] R. Bird, W. M. Coombs, and S. Giani, “A Posteriori Discontinuous Galerkin Error Estimator for Linear Elasticity,” *Applied Mathematics and Computation*, vol. 344–345, pp. 78–96, 2019.
- [166] N. Heuer, M. E. Mellado, and E. P. Stephan, “Hp-Adaptive Two-Level Methods for Boundary Integral Equations on Curves,” *Computing*, vol. 67, no. 4, pp. 305–334, 2001.

- [167] M. Ainsworth, and B. Senior, “Hp–Finite Element Procedures on Non–Uniform Geometric Meshes: Adaptivity and Constrained Approximation,” in *Grid Generation and Adaptive Algorithms*, edited by Marshall W. Bern, Joseph E. Flaherty, and Mitchell Luskin, 1–27. The IMA Volumes in Mathematics and Its Applications. New York, NY: Springer, 1999.
- [168] W. Gui, and I. Babuška, “The h, p and h–p Versions of the Finite Element Methods in 1 Dimension, Part III. The Adaptive h–p Version,” *Numerische Mathematik*, vol. 49, no. 6, pp. 659–683, 1986.
- [169] T. Wihler, “An Hp–Adaptive Strategy Based on Continuous Sobolev Embeddings,” *Journal of Computational and Applied Mathematics* vol. 235, no. 8, pp. 2731–2739, 2011.
- [170] W. Mitchell, and M. A. McClain “A Comparison of Hp–Adaptive Strategies for Elliptic Partial Differential Equations.” *ACM Transactions on Mathematical Software*, vol. 41, no. 1, pp. 1–39, 2014.
- [171] S. Giani, “A Hp-Adaptive Discontinuous Galerkin Method for Plasmonic Waveguides,” *Journal of Computational and Applied Mathematics*, Fourth International Conference on Finite Element Methods in Engineering and Sciences (FEMTEC 2013), 270, pp. 1220, 2014.
- [172] S. Giani, “High-Order/hp-Adaptive Discontinuous Galerkin Finite Element Methods for Acoustic Problems,” *Computing*, vol. 95, no. 1, pp. 215–234, 2013.
- [173] S. Giani, “A hp-adaptive discontinuous Galerkin method for plasmonic waveguides,” *Journal of Computational and Applied Mathematics*, vol. 270, pp. 12–20, 2014.
- [174] H. A. Eschenauer and N. Olhoff, “Topology optimization of continuum structures: A review,” *Applied Mechanics Reviews*, vol. 54, no. 4, pp. 331–390, 2001.
- [175] L. Krog, A. Tucker, and G. Rollema, “Application of Topology, Sizing and Shape Optimization Methods to Optimal Design of Aircraft Components,” Proc. 3rd Altair UK HyperWorks users conference, 2002.
- [176] D. Mylett, L. Butler, and S. Gardner, “Composite Optimisation of a Formula One Front Wing,” Proc. 6th Altair CAE Technology Conference 2009.
- [177] Y. He, D. Burkharter, D. Durocher, and J. M. Gilbert, “Solid–Lattice Hip Prosthesis Design: Applying Topology and Lattice Optimization to Reduce Stress Shielding From Hip Implants.” In 2018 Design of Medical Devices Conference. Minneapolis, Minnesota, USA: American Society of Mechanical Engineers, 2018.
- [178] A. G. M. Michell, “LVIII. The limits of economy of material in frame-structures.” *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, vol. 8, no. 47, pp. 589–597, 1904.
- [179] W. Prager, “A note on discretized michell structures,” *Computer Methods in Applied Mechanics and Engineering*, vol. 3, pp. 349–355, 1974.

- [180] G. I. N. Rozvany, “Optimal load transmission by flexure,” *Computer Methods in Applied Mechanics and Engineering*, vol. 1, pp. 253–263, 1972.
- [181] G. I. N. Rozvany, “Grillages of maximum strength and maximum stiffness,” *International Journal of Mechanical Sciences*, vol. 14, pp. 651–666, 1972.
- [182] G. I. N. Rozvany, “Exact analytical solutions for some popular benchmark problems in topology optimization,” *Structural optimization*, vol. 15, pp. 42–48, 1998.
- [183] T. Lewiński and G. I. N. Rozvany, “Exact analytical solutions for some popular benchmark problems in topology optimization II: Three-sided polygonal supports,” *Structural and Multidisciplinary Optimization*, vol. 33, pp. 337–349, 2007.
- [184] T. Lewiński and G. I. N. Rozvany, “Exact analytical solutions for some popular benchmark problems in topology optimization III: L-shaped domains,” *Structural and Multidisciplinary Optimization*, vol. 35, pp. 165–174, 2008.
- [185] M. P. Rossow and J. E. Taylor. “A finite element method for the optimal design of variable thickness sheets.” em AIAA journal, vol. 11, no. 11, pp. 1566–1569, 1973.
- [186] O. M. Querin, G. P. Steven, and Y. M. Xie, “Evolutionary structural optimisation using an additive algorithm,” *Finite Elements in Analysis and Design*, vol. 34, pp. 291–308, 2000.
- [187] O.M. Querin, G.P. Steven, and Y.M. Xie, “Evolutionary structural optimisation (ESO) using a bidirectional algorithm,” *Engineering Computations*, vol. 15, pp. 1031–1048, 1998.
- [188] M. Zhou and G. Rozvany, “On the validity of ESO type methods in topology optimization,” *Structural and Multidisciplinary Optimization*, vol. 21, pp. 80–83, 2001.
- [189] G. I. N. Rozvany, “A critical review of established methods of structural topology optimization,” *Structural and Multidisciplinary Optimization*, vol. 37, pp. 217–237, 2009.
- [190] N. P. van Dijk, K. Maute, M. Langelaar, and F. van Keulen, “Level-set methods for structural topology optimization: A review,” *Structural and Multidisciplinary Optimization*, vol. 48, pp. 437–472, 2013.
- [191] G. Allaire, F. Jouve, and A.-M. Toader, “A level-set method for shape optimization,” *Comptes Rendus Mathématique*, vol. 334, pp. 1125–1130, 2002.
- [192] M. Y. Wang, X. Wang, and D. Guo, “A level set method for structural topology optimization,” *Computer Methods in Applied Mechanics and Engineering*, vol. 192, pp. 227–246, 2003.
- [193] J. L. Barrera, M. J. Geiss, and K. Maute. “Hole Seeding in Level Set Topology Optimization via Density Fields.” arXiv preprint arXiv:1909.10703 (2019).

- [194] V. J. Challis, “A discrete level-set topology optimization code written in Matlab,” *Structural and Multidisciplinary Optimization*, vol. 41, pp. 453–464, 2010.
- [195] S. Shojaee and M. Mohammadian, “Structural topology optimization using an enhanced level set method,” *Scientia Iranica*, vol. 19, pp. 1157–1167, 2012.
- [196] F. Murat, and J. Simon. “Etude de problmes d’optimal design.” *IFIP Technical Conference on Optimization Techniques*. Springer, Berlin, Heidelberg, 1975.
- [197] J. Nocedal and S. J. Wright, *Numerical Optimization*. Springer Series in Operations Research, New York: Springer, 2nd ed. 2006.
- [198] S. I. Valdez, S. Botello, M. A. Ochoa, J. L. Marroqun, and V. Cardoso. “Topology Optimization Benchmarks in 2D: Results for Minimum Compliance and Minimum Volume in Planar Stress Problems.” *Archives of Computational Methods in Engineering*, vol. 24, no. 4, pp. 803–839, 2017.
- [199] E. Hairer, “A Runge-Kutta Method of Order 10,” *IMA Journal of Applied Mathematics*, vol. 21, no. 1, pp. 47–59, 1978.
- [200] J. C. Butcher, *Numerical Methods for Ordinary Differential Equations, Third Edition*. John Wiley & Sons, 2016.