UNIVERSITY OF SOUTHAMPTON

Numerical Simulation of Diffusion Controlled Reactions

Stuart Christopher Benedict Abercrombie, B.Sc.

Submitted for the degree of Ph.D.
Chemistry Department, Faculty of Engineering, Science and Mathematics, September 2003

UNIVERSITY OF SOUTHAMPTON

ABSTRACT

FACULTY OF ENGINEERING, SCIENCE AND MATHEMATICS

CHEMISTRY

Doctor of Philosophy

NUMERICAL SIMULATION OF DIFFUSION CONTROLLED REACTIONS

by Stuart Christopher Benedict Abercrombie

The amperometric response of electrodes generally cannot be predicted without taking into account mass transport effects. These are described by partial differential equations that often require numerical solution. In this thesis the adaptive finite element method is examined as a means to this end.

Adaptive finite element, while long used in engineering fields, has not so far been significant in electrochemical simulation. Most simulations have been effected with finite difference or non-adaptive finite element, with *a priori* mesh densities. Neither of these has the advantage of error control that the algorithm presented here has, nor do they allow the same geometric flexibility. An efficient, and in many ways novel, implementation of adaptive finite element is described, which allows a user-defined error bound to be met using an optimised machine-generated mesh. Rather than utilising generic error measures, the mesh is optimised specifically for accuracy in the current using a new error estimation strategy. This yields a widely applicable steady state simulation program whose flexibility is demonstrated with a variety of realistic problems.

Acknowledgements

This work would not have been completed without the help of a number of people.

My supervisor Dr Guy Denuault amiably gave me free rein to pursue my ideas, for which I'm very grateful. John Angus and Jonathan Amphlett both gave me the benefit of their simulation experience, as well as some entertaining times in the pub. Stuart Evans provided not only experimental insight but, crucially, and often abetted by David Cook, the impetus for lengthy and frequently hilarious stints in the tea room.

Outside of chemistry, I must thank my other friends, in particular Matthew Powell and Matthew Adams who managed to plough through enough of the text to provide some useful comments. They, Ian, Malcolm, Simon, Kate and Stuart Wallace have helped make life interesting.

I thank my family for supporting me before and during this work. It wouldn't exist without them.

Finally, I would like to thank Dr David Gavaghan, the external examiner, for technical advice that significantly improved the final thesis, as well as for his generous approach in the viva.

Contents

List of Figures

List of Tables

List of Symbols

Latin

Symbol	Meaning	Dimensions	Section
A	electrode area	cm^2	1.1
A	species A		4.1.1
$\mathbf{a}$	vector of nodal field values		2.1.2
$\mathbf{a}^e$	nodal field values of element e		2.1.3
$\mathbf{a}$	a vector field		1.1.3
a_i	field value at node i		2.1.2
a	electrode radius	cm	1.4
a	PDE diffusive coefficient		1.3.2
B	species B		4.1.1
b	lower order boundary integral		4.1.3
$\hat{b}$	higher order boundary integral		4.1.3
C^n	continuity of order n		2.1.1
c	concentration	$\mathrm{mol\ dm}^{-3}$	1.1
c^*	bulk concentration	$\mathrm{mol\ dm}^{-3}$	1.1.4
c_A	concentration of species A	$\mathrm{mol\ dm}^{-3}$	4.1.1
c_A^*	bulk concentration of species A	$\mathrm{mol\ dm}^{-3}$	4.1.1
c_A^s	surface concentration of species A	$\mathrm{mol\ dm}^{-3}$	4.1.1
c_B	concentration of species B	$\mathrm{mol\ dm}^{-3}$	4.1.1
c_B^*	bulk concentration of species B	$\mathrm{mol\ dm}^{-3}$	4.5.1
c_B^s	surface concentration of species B	$\mathrm{mol\ dm}^{-3}$	4.1.1

Symbol	Meaning	Dimensions	Section
c_O	concentration of oxidised species	mol dm^{-3}	1.1
c_R	concentration of reduced species	mol dm^{-3}	1.1.4
c_Z	concentration of species Z	mol dm^{-3}	4.5.1
D	diffusion coefficient	cm^2 s^{-1}	1.1.1
D_A	diffusion coefficient of species A	cm^2 s^{-1}	4.1.1
D_B	diffusion coefficient of species B	cm^2 s^{-1}	4.1.1
D_O	oxidised species diffusion coefficient	cm^2 s^{-1}	1.1
D_R	reduced species diffusion coefficient	cm^2 s^{-1}	1.1.4
d	lower order domain integral		4.1.3
$\hat{d}$	higher order domain integral		4.1.3
E	electrode potential	V	1.1.4
$E^{0\prime}$	formal potential	V	1.1.4
e^-	an electron		1.1
e	local field error		3.2.3
$\mathbf{e_q}$	local field gradient error		3.2.3
$\mathbf{F}$	global forcing vector		2.1.2
$\mathbf{F}^e$	element forcing vector		2.1.5
F	Faraday's constant	C mol^{-1}	1.1
f	F/RT	V^{-1}	1.1.4
f	PDE source term	mol dm^{-3} s^{-1}	1.3.2
g	microband electrode spacing		4.4.1
h	element size measure		2.2
h	raised/recessed offset of microbands		4.4.1
h	height of channel flow cell	cm	4.6
I_1, I_2, etc.	stiffness matrix integrals		2.1.5
$I(\cdot)$	current functional		3.2.2
i	electrode current	A	1.1.4
i	$\sqrt{-1}$		Appendix A
i_c	cathodic cell current	A	1.1

Symbol	Meaning	Dimensions	Section
i_{norm}	normalised electrode current		1.4
j	flux density	mol cm^{-2} s^{-1}	1.1.1
$\mathbf{j_c}$	convective flux density vector	mol cm^{-2} s^{-1}	1.1.1
j_c	convective flux density	mol cm^{-2} s^{-1}	1.1.1
$\mathbf{j_d}$	diffusive flux density vector	mol cm^{-2} s^{-1}	1.1.1
j_d	diffusive flux density	mol cm^{-2} s^{-1}	1.1.1
K	global stiffness matrix		2.1.2
K^e	element stiffness matrix		2.1.5
K	dimensionless rate constant		3.2.6
$K(\cdot)$	complete elliptic integral of 1st kind		4.4.1
k	1st order homogeneous rate constant	s^{-1}	1.1.3
k_b	heterogeneous backward rate constant	cm s^{-1}	1.1.4
k_f	heterogeneous forward rate constant	cm s^{-1}	1.1.4
k_{ox}	oxidation rate constant	cm s^{-1}	1.1
k_{red}	reduction rate constant	cm s^{-1}	1.1
k^0	intrinsic heterogeneous rate constant	cm s^{-1}	1.1.4
L	element edge length		2.1.5
L	microdisc recess depth		4.2
$\mathcal{L}(\cdot)$	differential operator		1.1.2
l	characteristic domain length	cm	1.4
l_1, l_2, l_3	triangle area coordinates	cm	1.4
m	$g/(g+2)$		4.4.2
$\mathbf{N}$	vector of nodal shape functions		2.1.2
$\mathbf{N}^e$	element e nodal shape functions		2.1.3
N	number of statistical samplings		1.3.4
N_i	shape function at node i		2.1.2
n	number of electrons		1.1.4
n	number of mesh nodes		2.1.2
n	number of mesh elements		3.2.3

Symbol	Meaning	Dimensions	Section
$\hat{\mathbf{n}}$	unit surface normal		1.1.3
$\mathcal{O}(\cdot)$	order of		1.3.3
O	oxidised species		1.1
Pe	element Peclet number		2.2
P_s	shear rate Peclet number		4.6
p	dimensionless pressure		6.3.5
$\mathbf{q}$	field gradient		3.2.3
q	PDE first order reaction coefficient		1.3.2
R	reduced species		1.1
R, r	dimensionless cylindrical r coordinate		1.4
Re	Reynolds number		6.3.5
r	cylindrical radial coordinate	cm	1.1.2
$\mathcal{S}$	the surface of volume $\mathcal{V}$		1.1.3
t	time	s	1.1.3
Δt	small temporal distance	s	1.3.1
u	dimensionless concentration		1.3.2
$\bar{u}$	FE approximation of u		2.1.2
$\bar{u}^e$	FE approximation of u within element e		2.1.3
$\bar{u}$	lower order FE approximation of u		3.2.3
$\tilde{u}$	higher order FE approximation of u		3.2.3
u_x	dimensionless fluid velocity x component		6.3.5
u_y	dimensionless fluid velocity y component		6.3.5
$\bar{v}$	lower order FE approximation of v		3.2.4
$\tilde{v}$	higher order FE approximation of v		3.2.4
$\mathcal{V}$	an arbitrary volume		1.1.3
$\mathbf{v}$	convective velocity field	cm s^{-1}	1.1.1
v_x	convective velocity field x component	cm s^{-1}	1.1.1
v_y	convective velocity field y component	cm s^{-1}	1.1.1
v	weighted dual field		3.2.2

Symbol	Meaning	Dimensions	Section
v'	unweighted dual field		3.2.2
v_0	flow cell inlet velocity	cm s^{-1}	4.6
W_i	weight function at node i		2.1.2
w	weight function		1.3.2
X, x	dimensionless Cartesian x coordinate		1.4
x	Cartesian x coordinate	cm	1.1
Δx	small distance in x direction	cm	1.3.1
x_e	channel flow microband width	cm	4.6
Y, y	dimensionless Cartesian y coordinate		1.4
Y	species Y		4.5.1
y	Cartesian y coordinate	cm	1.1.2
Δy	small distance in y direction	cm	1.3.1
Z, z	dimensionless cylindrical z coordinate	cm	1.4
Z	species Z		4.5.1
z	cylindrical z coordinate	cm	1.1.2

Greek

Symbol	Meaning	Dimensions	Section
α	transfer coefficient		1.1.4
α	Robin BC homogeneous coefficient		1.3.2
α	stabilisation parameter		2.2
α	SECM tapering angle		5.1
$\alpha_1, \alpha_2, \alpha_3$	linear shape function coefficients		2.1.3
β	Robin BC inhomogeneous component		1.3.2
Γ	simulation domain boundary		1.1.2
Γ_D	subset of Γ with Dirichlet boundary		2.1.1
Γ^e	an edge of element e		2.1.5
Γ_R^e	a Robin boundary edge of element e		2.1.5
Γ_e	electrode boundary section		3.2.2
Γ_R	subset of Γ with Robin boundary		2.1.1
Δ	element area		2.1.3
δ	surface concentration ratio		4.1.1
ϵ_i	element i error		3.2.3
ζ	current calculation weighting function		3.2.2
η	relative global error		3.2.3
$\bar{\eta}$	global error tolerance		3.2.3
ξ_1, ξ_2, ξ_3	various current error estimates		3.2.4
ρ	reaction source or sink	mol dm^{-3} s^{-1}	1.1.3
τ	degree of SUPG weighting		2.2
Ω	simulation domain		1.1.2
Ω^e	element e		2.1.5

List of Abbreviations

Abbreviation	Definition	Section
ADI	Alternating Direction Implicit	1.3.1
BE(M)	Boundary Element (Method)	1.3.3
BC	Boundary Condition	1.1.4
BiCG	BiConjugate Gradient	Appendix D
BiCGStab	BiCG Stabilised	Appendix D
CDT	Constrained Delaunay Triangulation	3.1.2
CRS	Compressed Row Storage	2.3.2
DOF	Degrees Of Freedom	2.1
DRM	Dual Reciprocity Method	1.3.3
EFD	Explicit Finite Difference	1.3.1
FD(M)	Finite Difference (Method)	1.3.1
FE(M)	Finite Element (Method)	1.3.2
GMRES	Generalised Minimum Residual	Appendix D
MWR	Minimum Weighted Residual	1.3.2
ODE	Ordinary Differential Equation	1.1.4
(P)CG	(Preconditioned) Conjugate Gradient	Appendix D
PDE	Partial Differential Equation	1.1.3
SECM	Scanning Electrochemical Microscopy	1.5
SPR	Superconvergent Patch Recovery	3.2.3

Chapter 1

Mass Transport Problems

The aim of this work is to model certain electrochemical systems. In the quantitative interpretation of experimental results it is often found that, having formulated the equations describing the problem, one is left with a difficult mathematical challenge. We attempt here to solve some of these problems.

Electrochemical experiments exist in both *galvanostatic* and *potentiostatic* forms, corresponding to controlled current and potential respectively. The ambit of this work extends to the latter only, where typically the current is measured, for instance to determine the concentration of an electroactive species, or a reaction rate constant. In order to relate a measured current to a chemical quantity, mass transport effects must often be modelled, and it is on these that this thesis focuses. More particularly, we aim to model mass transport to the *microelectrodes* now ubiquitous in electrochemistry.

The factors governing mass transport rates are the mass transport mode—diffusion, convection or migration—and the electrode geometry—microdisc, microband, etc. The effect of each mass transport regime is discussed in §1.1.1; it governs the equation we formulate to model the movement of species. The electrode geometry determines the boundary conditions we impose on the aforesaid equation. A variety of electrode geometries are covered in this work, most with diffusive mass transport only, but convection is

also considered.

A determinant of how we approach the mass transport problem is whether or not experimental measurements (typically of the current) are time dependent in the region of interest. If so, the concentration of reactant is generally varying, and we must solve the *transient* problem. In reality, of course, all systems are transient, but frequently to within experimental accuracy the current is unchanging, and we assume this to be owing to a time independent concentration field. This is termed a *steady state* problem. We deal with only steady state problems in this thesis, which is an unfortunate limitation, but an important simplification for our work. Extension to transients is considered in the final chapter.

A final factor that cannot be ignored is the mechanism, which may involve heterogeneous reactions (at the electrode) and homogeneous reactions (in solution). These are intimately coupled to the mass transport, and bear on the boundary conditions and governing equation respectively. The incorporation of arbitrary mechanisms is a difficult problem when considering more complex geometries, with simulations usually only addressing a few cases. We consider only first order reactions, and a limited subset of these in practice, but wider applicability is one of our aims.

1.1 The Problem

Consider the elementary single electron electrode reaction (Figure 1.1 on the following page)

$$\mathrm{O} + \mathrm{e}^- \underset{k_{\mathrm{ox}}}{\overset{k_{\mathrm{red}}}{\rightleftarrows}} \mathrm{R}\,, \tag{1.1}$$

where O and R are oxidised and reduced species respectively, and k_{red} and k_{ox} their reduction and oxidation rate constants. Regardless of the nature of O and R, Fick's first law (see [2] and § 1.1.1 on page 4) tells us that the Faradaic current is directly related to the gradient of the concentrations of

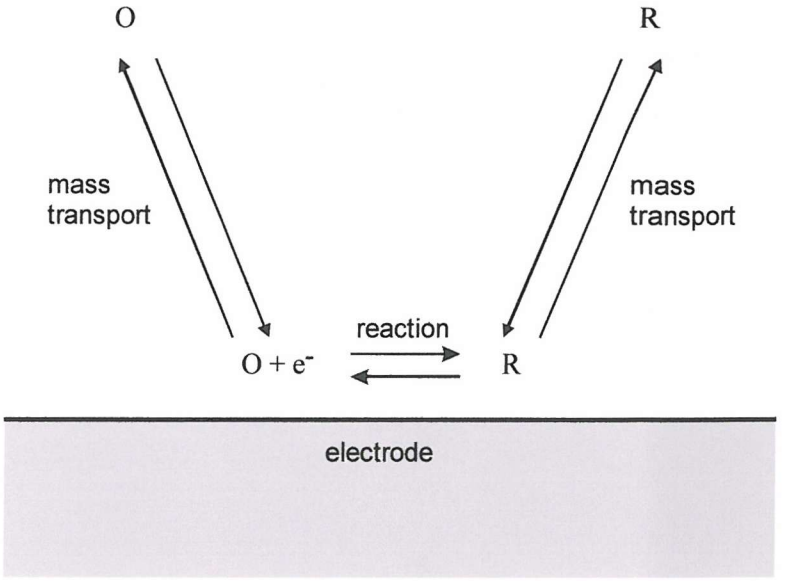

Figure 1.1: Schematic of an electrode reaction.

O and R. For instance, the cathodic current, i_c, is given in a one dimensional system extending along the positive x axis from an electrode at $x = 0$ by

$$i_c = AFD_O \left.\frac{\partial c_O}{\partial x}\right|_{x=0}, \tag{1.2}$$

with A being the electrode area, F Faraday's constant, D_O the diffusion coefficient of species O, and c_O its concentration. This statement is a relation between concentration gradients—and hence flux—and current; it says nothing about reaction kinetics, which we consider later.

In order to predict the current, then, we must generally be able to calculate $\frac{\partial c}{\partial n}$ at the electrode.[1] The obvious means of doing this is to find the concentration as a function of position (termed the concentration field), and differentiate it, which is what we in principle proceed to do. Since the concentration field depends on mass transport, and it is this aspect that concerns us, the next step is to consider the types of mass transport and the equations governing them.

[1] In this work $\frac{\partial}{\partial n}$ denotes differentiation with respect to the outward normal. Here it is equal to $-\frac{\partial}{\partial x}$.

1.1.1 Modes of Mass Transport

There are three types of solution mass transport, any and all of which can appear in electrochemical systems. They are diffusion, convection and migration, and are described below.

Diffusion

Diffusion is the most commonly important mode of mass transport. It is always present, and its effect can only be neglected when that of convection or migration is overwhelmingly stronger. It arises from the random thermal motion of solvent molecules, as hypothesised in the 1905 analyses of Einstein and Smoluchowski (see, for instance, [3]). But this assumption is not necessary for its mathematical analysis, and was not made by Fick in 1855 when constructing his famous laws [4].

Fick made the experimental observation, now termed *Fick's first law*, that

$$\mathbf{j_d} = -D\nabla\mathbf{c}\ ; \tag{1.3}$$

or, in its more familiar one dimensional form,

$$j_d = -D\frac{\partial c}{\partial x}\ . \tag{1.4}$$

That is, the diffusive flux density vector, $\mathbf{j_d}$, is negatively proportional to the concentration gradient, the constant of proportionality being the diffusion coefficient[2] D. In other words, diffusion works to smooth out concentration gradients by equalizing concentrations; it is an entropic, time-irreversible, effect.

[2]In reality D is not always a constant, and can depend on the concentration of solute species [5]; but in many cases this is not a significant factor, and henceforth we shall assume that it is not. We also note later that D need not be a scalar.

Convection

Convection is the net movement of solvent and solute molecules caused by physically imparted momentum. It comes in two forms: natural and forced. The aim is usually to eliminate the former, as it is extremely difficult to predict, deriving from all manner of effects within the reaction solution—gravity, density gradients, temperature gradients, etc.

Forced convection, conversely, is by definition induced by the cell design. Sometimes the resulting convection is deliberately chaotic and unpredictable, as with "turbulence promoters" in industrial cells, and in such cases would almost certainly be analytically unquantifiable. In forced convection experiments, however, the velocity field is designed to be predictable, and if it can be found as a function of spatial position then it can be incorporated into the mass transport equations.

Ignoring fairly small effects such as density gradients, etc. caused by reactant mass transport or reaction, the Navier-Stokes equations (see [6] and the final chapter of this work) governing the velocity field $\mathbf{v}$ are independent of the reactant mass transport equation(s), and can therefore be solved independently. In a tiny number of cases, notably that of the Poiseuille flow (*ibid.*) in a channel flow cell, the velocity field may be determined exactly analytically. In others, approximations are sometimes applicable, for instance with the wall jet cell [7–11]. In all the following work we assume that $\mathbf{v}$ or an approximation thereof is known, since solving the fluid dynamics problem is generally harder than the reactant mass transport one (some of the difficulties are mentioned in the final chapter).

Under the assumption that we know $\mathbf{v}$, its incorporation into the mass transport equation is simple. The convective flux density, $\mathbf{j_c}$, clearly depends on the magnitude of the components of the velocity field and the concentration of the species with the aforesaid velocity. It is then

$$\mathbf{j_c} = c\mathbf{v} \; ; \tag{1.5}$$

or, if only the x component of $\mathbf{v}$ is non-zero,

$$j_c = cv_x \ . \tag{1.6}$$

While a relatively small change to the mass transport equation, the convective term makes both analytical and numerical solution considerably harder, some of the reasons for which are discussed later. This is perhaps not entirely surprising, as the character of convection is entirely different from that of diffusion—for example, it is, unlike diffusion, time-reversible.

Migration

Migration is the effect of ions, as a result of their charge, moving under the force from the electric field created by the potential difference between the cell's electrodes. It is commonly eliminated in experimental electrochemistry by the addition of a relatively large quantity of supporting electrolyte, and so will not be considered further. It could, however, become more important in simulations as new experiments without supporting electrolyte are performed. Apart from contributing a term to the mass transport equation, it generally requires the solution to another coupled PDE governing the electric field.

1.1.2 Electrode Geometries

The choice of electrode geometry is in part tied to the mass transport regime, and is also related to whether or not the experiment is transient or steady state. Since the early 1980s *microelectrodes* have become important to many experiments [12]. These are generally held to be electrodes with a characteristic dimension of less than 50μm, allowing a number of advantages.

Firstly, of course, small electrodes can be used in situations where large ones will not fit. But other advantages include high rates of mass transport for studying fast mechanisms, and reduced capacitative charging and ohmic

drop (*ibid.*). However, while alleviating experimental problems, microelectrodes generally increase theoretical ones. Macroelectrode edge effects can often be ignored, allowing symmetric reduction to one dimension (see below), but with microelectrodes edge effects are crucial to an accurate description of mass transport, and more dimensions must usually be incorporated into the mass transport equation. For example, diffusive transport to planar electrodes and convective-diffusive transport to rotating disc electrodes are well served with simple (semi-)analytical models, but even the simple case of the steady state current to a recessed microdisc lacks such a thing over the range of possible recess depths.

As a result of these considerations, the simulations in this thesis primarily deal with microelectrodes. Two important cases of these are microbands and microdiscs, and both will be studied in a number of guises. For these purposes, either Cartesian or cylindrical coordinate systems will be adopted. This is concealed in the Laplacian in the coordinate system-independent formulation in §1.1.3, so we make the forms explicit here.

For the purposes of microband-type simulations, with translational symmetry in the z direction, we use the simplified Laplacian operator:

$$\nabla^2 = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2} \,. \tag{1.7}$$

In microdisc-type simulations, where the solution is taken as angularly independent,

$$\nabla^2 = \frac{\partial^2}{\partial r^2} + \frac{1}{r}\frac{\partial}{\partial r} + \frac{\partial^2}{\partial z^2} = \frac{1}{r}\frac{\partial}{\partial r}\left(r\frac{\partial(\cdot)}{\partial r}\right) + \frac{\partial^2}{\partial z^2} \,. \tag{1.8}$$

The related grad and div operators, ∇ and $\nabla\cdot$, also clearly vary.

Whatever the precise geometry, the conception of the problem is of solving some transport equation within a domain Ω whose shape is defined by the extent of the solution in the cell. Since the scale of the problem we wish to solve is typically much smaller than the overall extent of solution, it is usual to approximate the simulation domain as infinite in one or more directions. This generally helps in analytical solution efforts, as it increases the symmetry of

the problem, but poses difficulties with most numerical techniques. Thus often a finite domain is simulated, but one much smaller than the true one, and this must be taken into account.

Mathematically we can write the steady state problem as finding c such that

$$\mathcal{L}c = d\,, \tag{1.9}$$

where $\mathcal{L}$ is a differential operator, with appropriate boundary conditions for c on the boundary of the domain, Γ. Where multiple species are involved, we may take c to be a vector quantity. Before solving the problem, then, we must determine the nature of $\mathcal{L}$ and the boundary conditions.

1.1.3 The Reaction-Diffusion-Convection Equation

We now formulate the governing equation that we shall spend the rest of this thesis trying to solve, in one form or another. There is a microscopic, stochastic approach to the derivation, going straight from first principles to the ultimate transport equation. But having given the constitutive laws above, we can immediately use the principle of conservation of mass to establish the *reaction-diffusion-convection equation* without a microscopic model.

Many non-stochastic derivations of the diffusion equation look at infinitesimal blocks, and result in coordinate system-dependent formulae (e.g. see [2]). We shall take a mathematically more general tack, and incorporate convection and reaction as well, but fundamentally the idea is the same.

Firstly, we require *Gauss's Divergence Theorem* [13], which is also used several times later in this work. Essentially it states that, if $\mathbf{a}$ is a suitable vector field over an arbitrary volume $\mathcal{V}$ with a surface $\mathcal{S}$ having an outward unit normal $\hat{\mathbf{n}}$, then

$$\int_{\mathcal{S}} \hat{\mathbf{n}} \cdot \mathbf{a} dS = \int_{\mathcal{V}} \nabla \cdot \mathbf{a} dV \ : \tag{1.10}$$

it relates surface integrals to volume integrals.

The diffusive-convective flux is, from equations (1.3) and (1.5),

$$\mathbf{j} = \mathbf{j_d} + \mathbf{j_c} = c\mathbf{v} - D\nabla\mathbf{c} \; . \tag{1.11}$$

The flux into $\mathcal{V}$ through the surface $\mathcal{S}$ is then

$$\int_{\mathcal{S}} \hat{\mathbf{n}} \cdot (D\nabla\mathbf{c} - c\mathbf{v})\, dS \tag{1.12}$$

(since $\hat{\mathbf{n}}$ is the outward normal we change the sign), giving, with the Divergence Theorem,

$$\int_{\mathcal{V}} \nabla \cdot (D\nabla\mathbf{c} - c\mathbf{v})\, dV \; . \tag{1.13}$$

The rate of change of material within $\mathcal{V}$ is, by definition,

$$\frac{d}{dt}\int_{\mathcal{V}} c dV = \int_{\mathcal{V}} \frac{\partial c}{\partial t} dV \tag{1.14}$$

(the interchange of limits is valid under reasonable assumptions).

Heterogeneous electrode reactions clearly by their nature occur at the boundary of the domain within which mass transport is modelled, and thus do not come into the governing equation. Commonly, though, homogeneous reactions happen too, and manifest themselves as sources or sinks of the species whose concentration we are modelling. We denote a general source or sink of material by $\rho = \rho(x, y, c)$ (its sign determines which—positivity implies a source).

By conservation of mass, the molar rate of change of material within $\mathcal{V}$ must equal the quantity entering through its surface (1.13), plus the amount generated within $\mathcal{V}$ by sources,

$$\int_{\mathcal{V}} \rho dV \; . \tag{1.15}$$

Therefore

$$\int_{\mathcal{V}} \frac{\partial c}{\partial t} dV = \int_{\mathcal{V}} \nabla \cdot (D\nabla\mathbf{c} - c\mathbf{v})\, dV + \int_{\mathcal{V}} \rho dV \; . \tag{1.16}$$

Collapsing the integrals into one, and distributing the scalar product,

$$\int_{\mathcal{V}} \left(\frac{\partial c}{\partial t} - \nabla \cdot (D\nabla c) + \nabla\mathbf{c} \cdot \mathbf{v} - \rho \right) dV = 0 \; . \tag{1.17}$$

Since the volume $\mathcal{V}$ is arbitrary, and has not been specified, (1.17) can only be true if the integrand is zero (again under reasonable mathematical assumptions). We thus arrive at the general reaction-diffusion-convection equation—the basis for all the work presented in this thesis:

$$\frac{\partial c}{\partial t} = \nabla \cdot (D \nabla c) - \nabla \mathbf{c} \cdot \mathbf{v} + \rho \,. \tag{1.18}$$

It is a second order *partial differential equation* (PDE), which will be linear if ρ is (and if both D and $\mathbf{v}$ are independent of c, which we have already assumed). Often D is taken to be independent of spatial position, in which case we have a slightly more familiar form:

$$\frac{\partial c}{\partial t} = D \nabla^2 c - \nabla \mathbf{c} \cdot \mathbf{v} + \rho \,. \tag{1.19}$$

(We shall make this assumption, although it makes no difference to the numerical solution formulation.) In this work, as has already been stated, only steady state problems are considered, which means solving

$$D \nabla^2 c - \nabla \mathbf{c} \cdot \mathbf{v} + \rho = 0 \,. \tag{1.20}$$

A few things should be noted about (1.18) and its related forms. We have assumed that D, the diffusion coefficient, is a scalar. If diffusion were faster in some directions than others—if it were anisotropic—then D would be a *tensor* [14,15]. In fact this usually complicates the theory only slightly,[3] but it can make simulation more difficult in practice. In this work we assume isotropic diffusion.

Secondly, simply by reinterpreting c as a vector of concentrations, we have, *mutatis mutandis*, the reaction-diffusion-convection equation for a system of interdependent species. It should be remembered here that the term ρ can be complicated: it will usually be a function (commonly a polynomial) of c, and could have other dependencies.

[3] If the tensor can be *diagonalised*, as is often possible, tensor analysis can be discarded entirely, and the coefficient viewed simply as having three different values for the three spatial axes [16].

There are a number of special cases of (1.19). It is important to identify them, as certain commonly encountered forms and techniques for their solution are well documented in the literature. Four important examples are given below.

Diffusion Equation

Also known as *Fick's second law*, the case where $\mathbf{v} = \mathbf{0}$, $\rho = 0$ is, in most fields, called the *diffusion equation*:

$$\frac{\partial c}{\partial t} = D\nabla^2 c \,. \tag{1.21}$$

For non-convective systems where homogeneous reactions are absent or negligible, (1.21) is an important model. It is, according to the classification used for partial differential equations [13], a *parabolic* equation. This is important for solving it numerically, as parabolic partial differential equations have entirely different properties from the other common time-dependent type, *hyperbolic*. The distinction becomes important when dealing with mixed diffusion and convection problems.

Because it governs many other phenomena, most notably heat conduction (for this reason it is also sometimes called the *heat equation*), (1.21) has been studied extensively. It forms a model transient problem, and its consideration is important for extension to transients, discussed in Chapter 6.

Laplace Equation

A further specialisation of (1.21) assumes that the concentration field is unchanging—that the problem is steady state. Here $\frac{\partial c}{\partial t} = 0$, so

$$\nabla^2 c = 0 \,. \tag{1.22}$$

The Laplace Equation (1.22) is clearly only applicable where (1.21) is, but with the additional constraint that the concentration, and consequently current, be unchanging. This never truly happens, but depending on the experiment, can be approached quickly if mass transport is fast.

Unfortunately a complication arises when considering the relation with experiment. Experiments usually achieve a steady state, but not always a useful one purely described by diffusion: in practice, natural convection imposes a steady state in most potential step experiments, typically after about thirty seconds, limiting their maximum duration. If a purely diffusive steady state is desired it must establish itself well before this time, and this constrains the combination of mechanism and electrode geometry for which steady state experiments are useful. With a simple E mechanism microdiscs achieve this but microbands do not, for instance. All of the cases simulated in this work are known to exhibit true experimental steady states.

In common with the diffusion equation, Laplace's equation describes a variety of phenomena, obviously including steady state heat conduction, but also electrostatics, for example. Consequently, many analytical and numerical techniques have been applied to it. Lacking the time coordinate, it is of a radically different type—*elliptic* this time. Despite its apparent simplicity, and the wealth of literature on the subject, even this problem can be challenging to solve, the major difficulty being the nature of the *boundary conditions* imposed in electrochemical problems (see § 1.1.4 on the following page and § 1.1.5 on page 18). Much of this thesis is devoted solely to this problem.

Poisson Equation

A slight generalisation of the Laplace Equation entails a known solution-independent source term f, which may be a function of position:

$$D\nabla^2 c = -f \ . \tag{1.23}$$

The obvious interpretation is of a diffusant generated by a homogeneous chemical reaction at a rate f—i.e. with $\rho = f$ in equation (1.19). This *inhomogeneous* form of the Laplace equation arises in modelling mechanisms where species are coupled by homogeneous reactions and one concentration

field can be determined independently of the others (for instance, with the irreversible ECE mechanism [2]), as well as with zero order reactions.

Modified Helmholtz Equation

A less commonly seen generalisation of (1.22) is the modified Helmholtz equation [17]. It has the form

$$D\nabla^2 c - kc = 0 \,, \tag{1.24}$$

where k is a positive constant (if k were negative, (1.24) would become the plain Helmholtz equation, but this is not of interest to electrochemists.). It can be used to model a first order EC' mechanism [2], where the homogeneous rate constant is proportional to k. Beyond this, it serves as a building block in the simulation of more complex mechanisms, and appears in some form wherever there is a first order homogeneous reaction of a diffusing species. For instance, with the irreversible ECE mechanism the chemically reactive species is governed by an equation of the form:

$$D\nabla^2 c - kc = -f \,. \tag{1.25}$$

1.1.4 Boundary Conditions

It is not difficult to find functions that satisfy PDEs like equations (1.21) or (1.22) or (1.23) or (1.24); unlike ordinary differential equations (ODEs), the general solutions contain arbitrary *functions*, not constants, so there exists a multiplicity of candidates. The difficulty is in making such functions obey the imposed boundary conditions, which can be very complex in dimensionalities higher than one.

In electrochemistry, spatial boundary conditions at cell walls are defined by the reaction, or lack of, at the cell boundary/solution interface. Where the solution meets an active electrode, the reaction occurring there can be expected to impose some condition on the concentration that results in a

flux of material. Depending on the rate of reaction, its order, and the mechanism, different conditions need to be imposed. Conversely, at an insulating wall—the glass surrounding a microdisc, for example—no reaction will be occurring, and we expect the flux to be zero. Any number of reaction conditions could be imposed, but here, as in most work on the subject, we restrict ourselves to linear boundary conditions. This limits what we do to first order electrode processes, but these are by far the most common, and it avoids some complexity. Finally, where the simulation domain tends towards the bulk solution, conditions must be imposed there to reproduce this effect.

Linear boundary conditions are usually categorised mathematically into three types: Dirichlet, Neumann, and Robin. However, unlike many physical problems governed by PDEs, electrochemical problems almost always have *mixed* boundary conditions—that is, different types of boundary condition hold on different parts of the boundary.[4] Many of the problems of electrochemical simulation stem from this fact.

For transient problems, which we do not simulate, one must also impose initial conditions. These are usually of obvious form, (e.g. $c|_{t=0} = c^*$, where c^* is the bulk concentration), but do not affect the steady state solution.

The General Boundary Condition

With one exception, all of the boundary conditions that we shall impose—even the one on insulators—can be thought of as special or limiting cases of the general *current-potential characteristic* [2]. Assuming in this section that we are referring to concentrations at the boundary, it can be written as

$$i = nFAk^0 \left[c_O e^{-\alpha nf(E-E^{0\prime})} - c_R e^{(1-\alpha)nf(E-E^{0\prime})}\right] \tag{1.26}$$

$$= nFA\left[k_f c_O - k_b c_R\right] . \tag{1.27}$$

[4]Some authors use "mixed boundary condition" to refer to a Robin condition, on the grounds that it is a "mixture" of Dirichlet and Neumann conditions. The terminology used here seems more modern, and less apt to confuse.

The *standard* or *intrinsic rate constant*, k^0, is a potential-independent constant determined by the electrode system. The forward and backward rate constants, k_f and k_b, do depend on the potential. The other symbols have their usual meanings (*ibid.*).

By Fick's first law, as exemplified in (1.2), the total current, i, is related to the concentration gradients of both species O and R. Thus we must, in general, impose the conditions

$$-D_O \frac{\partial c_O}{\partial n} = k_f c_O - k_b c_R \tag{1.28}$$

and

$$D_R \frac{\partial c_R}{\partial n} = k_f c_O - k_b c_R \tag{1.29}$$

at electrodes. We note that they satisfy the mass balance condition

$$-D_O \frac{\partial c_O}{\partial n} = D_R \frac{\partial c_R}{\partial n} , \tag{1.30}$$

mandated by conservation of mass; the one implies the other in this sense.

These two conditions are essentially *Robin* conditions (although these are usually presented mathematically in the context of a single field) because they prescribe a linear combination of the concentration and its derivative. In the full general case the reactant and product concentrations are clearly inseparable—they must both be modelled and solved for—but often one can be considered in isolation, simplifying matters, if certain approximations are reasonable. It is also found that in the case of no reaction (at an insulator), the conditions reduce to *Neumann* conditions; and conversely at electrodes where there are very fast reactions *Dirichlet* conditions are imposed instead. We now consider these important special cases.

Insulators and Neumann Boundaries

At the insulating surrounds of electrodes there is clearly no reaction and no current. By definition, then, $k^0 = 0$, and we immediately see that

$$\frac{\partial c_O}{\partial n} = \frac{\partial c_R}{\partial n} = 0 . \tag{1.31}$$

These are examples of *Neumann conditions*, and specifically *homogeneous Neumann conditions*, because the prescribed derivative is zero. We note that the mass balance condition is inevitably satisfied by insulators with these conditions: no material travels either way.

Non-zero prescribed derivatives would seem on the face of it to correspond to a galvanostatic experiment, and as such would not be relevant to our work. In a similar manner to the Poisson equation, however, they reappear when modelling species that can be solved for sequentially. Consider the mass balance relation of (1.30). If $\frac{\partial c_O}{\partial n}$, for instance, is known (i.e. the c_O field has been determined independently), then enforcement of mass balance reduces to imposition of an inhomogeneous Neumann condition on c_R.

Reversible Systems and Dirichlet Boundaries

In the first major special case at electrodes the intrinsic rate constant is very high, implying equilibrium of surface concentrations, and the system is termed *reversible* [2].

If we take (1.26), divide by $nFAk^0$, and let $k^0 \to \infty$, we lose the current—and hence the concentration derivative—term, leaving

$$\frac{c_O}{c_R} \to e^{nf(E-E^{0\prime})} . \tag{1.32}$$

The exponential term is constant as far as we are concerned. The crucial point is that in this case we want to impose a ratio of concentrations: this is a *Dirichlet* type boundary condition (although, again, in the mathematical literature these are usually just values imposed on one field). It must be remembered that, as well as enforcing (1.32), we must still satisfy the mass balance condition of equation (1.30), which is not implied by the Dirichlet condition.

An approximation common to many theoretical treatments, where the concentration of reactant tends to zero, applies to the case where, on top of reversibility, we have a negligible backward reaction because we have a high

overpotential. This is a common simplification, attractive because it can mean the product species can be neglected entirely. Here, of course, mass balance is not a concern.

Totally Irreversible Systems and Robin Boundaries

In the second class of special case the kinetics in both directions are slow. If only one of the reactions is significant, one of the terms in equation (1.26) can be dropped. This is termed the *totally irreversible* case [2]. For instance, we might have

$$i = nFAk_f c_O \ , \tag{1.33}$$

meaning

$$-D_O \frac{\partial c_O}{\partial n} = k_f c_O \ . \tag{1.34}$$

This is still a Robin boundary condition, and is simple to enforce with finite element.

In common with many practitioners, for the most part we will be assuming one or the other of these special cases, not least because most of the approximate solutions with which we typically compare results do likewise.

Bulk Boundaries

Bulk boundaries do not fit into the general scheme of (1.1.4). As has already been said, most simulations must deal with the fact that the full expanse of the solution in which the active species exist is far larger than the small region around electrodes of interest. On the scale of the electrodes it has been shown through many experimental validations that the solution may therefore usually be approximated as infinite in extent.

While boundary and finite element can, in fact, deal with infinite domains, it is more usual to see simulations conducted on large but finite spaces. This is done principally for simplicity, and we follow this practice. The question remains, then, of imposing boundary conditions for the bulk at a

finite distance that will mimic the effect of those imposed infinitely far away. This is not a well investigated or documented area.

It is clear that the concentration tends towards the bulk value as one travels away from the electrode(s), but this would imply both the concentration tending to a constant and its derivative tending to zero. Thus one could arguably treat the bulk boundary either as an insulator (see above), or impose a Dirichlet condition for the bulk value. It would appear that the factor determining the best approximation is the effect on the quantity of interest—usually the current. Where the bulk boundary forms a crucial part of the system, for instance where it is the sole source of diffusant, then a Dirichlet condition would seem inescapable. With a generator-collector configuration, however, where other more important sources also exist, the choice is less clear cut, and the difference in results can be significant [18].

For testing purposes it is sometimes useful to be able to rule out bulk boundaries as a source of error, so with some simulations run for validation purposes, where an analytical solution exists, the exact values can be imposed on the bulk boundary. This is of course not a realistic practice, and a better solution in finite element might be *infinite elements* [16]. These are not a panacea, however, and are not widely documented. Some of their possibilities and potential difficulties are discussed in the final chapter.

1.1.5 Boundary Singularities

As already suggested, the importance of modelling edge effects in microelectrode experiments gives rise to simulation domains with mixed boundaries. Often, where an electrode is surrounded by insulating glass, and the electrode reaction is diffusion controlled, a Dirichlet boundary abuts a homogeneous Neumann boundary (see Figure 1.2). In the microdisc case illustrated we can expect there to be a non-zero flux all along the surface of the microdisc; indeed the analytical solution shows that it increases to infinity as the edge is reached. Next to this, however, is the insulating surround at which by

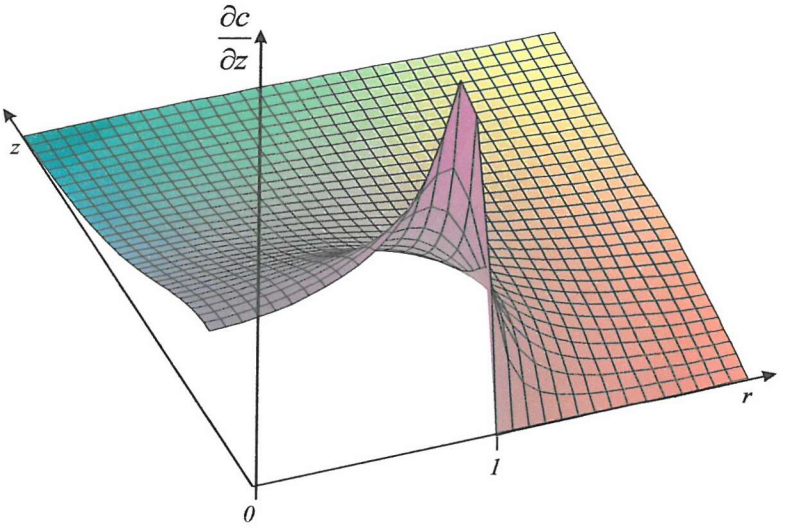

Figure 1.2: The concentration field z derivative near the edge of a microdisc with a radius of unity. The electrode is a Dirichlet boundary ($c = 0$), whereas the insulator is a Neumann boundary: $\frac{\partial c}{\partial n} = -\frac{\partial c}{\partial z} = 0$. The flux is discontinuous where they touch at $z = 0,\ r = 1$.

definition there is no flux. Thus, while the concentration field may be continuous at the point where they join, its normal derivative cannot be. This in turn means the concentration changes sharply, and can make its modelling difficult. This is a central problem of numerical microelectrode simulation, and is discussed below, when the solving methods are presented.

1.2 Analytical Solution Methods

The obvious place to start, as with most mathematical equations, is with analytical solutions. If such a thing can be found then we have the ideal case: a (presumably) quickly evaluable function then supplies the desired numbers. Unfortunately they are rare, and are almost entirely absent in domains with more than one spatial dimension. As stated earlier, this derives from the complexity of imposing boundary conditions where they can follow an arbitrarily complex line or surface in two or more dimensions. Important

to almost any successful attempt is symmetry of one type or another.

The most obvious example of helpful symmetry is that of the translational variety, allowing the approximation of certain spatial derivatives in the Cartesian diffusion equation as zero. For instance, where a large enough electrode is employed, edge effects may be neglected, and if the surface normal points in the x axis direction, the Cartesian diffusion equation

$$\frac{\partial c}{\partial t} = D\left(\frac{\partial^2 c}{\partial x^2} + \frac{\partial^2 c}{\partial y^2} + \frac{\partial^2 c}{\partial z^2}\right) \tag{1.35}$$

reduces to

$$\frac{\partial c}{\partial t} = D\frac{\partial^2 c}{\partial x^2}\,. \tag{1.36}$$

There exist a reasonable number of analytical solutions to the one dimensional transient problem posed by (1.36). Since, by a substitution, problems with spherical symmetry can be reduced to Cartesian one dimensional problems, these solutions also extend to hemispherical electrodes. However, less spatially symmetric geometries, for instance SECM tips, do not have this property, and solutions for even the simplest mechanisms do not exist. The only notable case of a truly geometrically flexible method for two dimensional problems is that of conformal mapping [19–21], which is only applicable to the Cartesian Laplace equation. It has yielded, for instance, the solution to the dual microband generator-collector problem [22], but three dimensions, homogeneous terms and transients all defeat it.

The inlaid microdisc does posses enough symmetry to allow some exact analytical solutions. In particular, the steady state E mechanism problem is fairly easily solved if the right coordinate system is used, as shown in Appendix A. The EC' case, however, is rather harder [23, 24], and the resulting solutions are unwieldy. As far as is known, no exact solution has been derived for the transient case, although approximations exist. Generally speaking, these approaches fall down as soon as the geometry of the problem changes, even slightly, as with a recessed microdisc.

Whatever can be achieved with a given problem, a difficulty of the analytical approach remains in that all calculations must be thrown away for a new

problem; it can only ever be problem-specific. The length and specificity of Appendix A perhaps make the most eloquent case against analytical solution techniques for the problems under consideration: adaptation is difficult for the EC' mechanism, and impossible for the recessed microdisc.

Relinquishing the hope of an exact formula for the desired answers, researchers have also sought expressions that approximate in a useful way the system under study. We adopt the description *semi-analytical* for these. Semi-analytical methods approximate at a less fundamental level than the numerical techniques discussed below, often using preexisting exact solutions for related cases. Numerical methods tend to use non-problem specific approximations.

For instance, since an analytical solution to the recessed microdisc E mechanism problem is unavailable, various approximations have been derived. Bond *et al.* [25], for example, assumed a constant concentration at the mouth of the recess, and derived a simple expression valid for relatively deeply recessed microdiscs. This can be a fruitful pursuit, but the specificity remains.

Since our aim is to solve general problems, and as a result of their limited applicability, the only use we shall make of analytical or semi-analytical expressions is in verifying the validity of the more general numerical techniques described next.

1.3 Numerical Solution Methods

Numerical techniques for solving the equations in which we are interested are manifold. They all share the characteristic of approximating to reduce the infinite dimensional problem posed by (1.22), say, to a finite dimensional one. Thus the differential equation problem becomes an algebraic one, and this is solved instead. The key, of course, is to ensure that the essential nature of the former is reflected in the latter. There are many ways of

attempting this, with varying degrees of efficacy and appropriateness, many of which we shall ignore completely: spectral, finite analytic, wavelet, for example. Most of these are simply not general enough for our problems. We shall only address the few techniques that have reasonable currency in the electrochemical world: finite difference, finite element, boundary element and random walk (Monte Carlo).

In electrochemistry the one dimensional numerical simulation problem has essentially already been solved, in that there exist packages that solve the range of problems of interest [26]. This is because one spatial dimension is far easier to deal with than two or three. Unfortunately this limits simulations to only one microelectrode of interest, the hemisphere. In order to simulate a variety of microelectrode geometries, a program capable of incorporating at least two spatial dimensions is necessary, and it is the aim of this work to develop such. Therefore a key requirement of the numerical method used is geometric flexibility. A second requirement is the efficient modelling of the boundary singularities mentioned above. The common simulation methods must therefore be evaluated with these *desiderata* in mind.

1.3.1 Finite Difference

One of the oldest numerical techniques for PDE solution—certainly predating computers (see [27], translated in [28])—is finite difference (FD). Most electrochemical simulations to date have used it, principally on account of its simplicity, at least in its more rudimentary forms.

Finite difference relies on the Taylor series approximation of derivatives in the defining equation. For instance,

$$c(x \pm \Delta x) = c(x) \pm \frac{\partial c}{\partial x}\Delta x + \frac{\partial^2 c}{\partial x^2}\frac{(\Delta x)^2}{2} + \ldots \tag{1.37}$$

(assuming differentiability and convergence).

Using such expansions, all the derivatives in (1.36) or any other PDE can be approximated. For example, using the first two terms of (1.37),

one can produce the approximate equation, known as the forward difference approximation,

$$\frac{\partial c}{\partial x} \simeq \frac{c(x+\Delta x) - c(x)}{\Delta x} \; ; \tag{1.38}$$

or the backward difference approximation

$$\frac{\partial c}{\partial x} \simeq \frac{c(x) - c(x-\Delta x)}{\Delta x} \; . \tag{1.39}$$

Both become exact as $\Delta x \to 0$, so their average, the central difference approximation, must too:

$$\frac{\partial c}{\partial x} \simeq \frac{c(x+\Delta x) - c(x-\Delta x)}{2\Delta x} \; . \tag{1.40}$$

Of course, these expressions' accuracy is contingent on Δx being small enough to render higher terms negligible. Assuming the solution is smooth enough, higher order methods will be more accurate. The forward and backward differences are first order, as they discard second order terms and above. The central difference is second order, as it happens that the second order terms cancel, and only terms of third order and above are neglected.

By writing another Taylor series for $\frac{\partial c}{\partial x}$ and repeating the process, a second order central difference approximation of the right-hand side of (1.36) can be derived:

$$\frac{\partial^2 c}{\partial x^2} \simeq \frac{c(x+\Delta x) - 2c(x) + c(x-\Delta x)}{(\Delta x)^2} \; . \tag{1.41}$$

This is used in many finite difference schemes, the main difference between which is the approximation used for $\frac{\partial c}{\partial t}$.

In the *explicit finite difference* (EFD) scheme [29], the forward difference expression is used for $\frac{\partial c}{\partial t}$, to give

$$\frac{c(x,t+\Delta t) - c(x,t)}{\Delta t} \simeq D\left[\frac{c(x+\Delta x,t) - 2c(x,t) + c(x-\Delta x,t)}{(\Delta x)^2}\right] \; . \tag{1.42}$$

One overlays a grid on the domain, and uses this approximation at each grid point. The "explicit" part of the name is explained when (discarding the approximation sign) we rearrange in terms of $c(x, t+\Delta t)$:

$$c(x,t+\Delta t) = c(x,t) + \frac{D\Delta t}{(\Delta x)^2}[c(x+\Delta x,t) - 2c(x,t) + c(x-\Delta x,t)] \; . \tag{1.43}$$

Assuming c is known at time t, it can be calculated for time $t + \Delta t$. In other words, the algebraic system associated with the approximation is diagonal [30], and trivial to solve. This holds with higher dimensional analogues. Note that, while the approximation of $\frac{\partial^2 c}{\partial x^2}$ is second order, the approximation of $\frac{\partial c}{\partial t}$ is only accurate to first order in time.

It should be apparent that the type of analysis above can be extended to any PDE, including, in principle, (1.19) in its full generality. In general FD can incorporate any convective or homogeneous reaction terms, including non-linear ones, so it is, *prima facie*, attractive for simulation programs designed to handle arbitrary mechanisms.

While EFD is simple, and important for illustrative purposes, it is not generally held to be a useful scheme, its widespread use in electrochemistry notwithstanding. It can be shown [30] that the scheme is only stable (that is, it only produces physically meaningful results) where

$$\frac{2D\Delta t}{(\Delta x)^2} \leq 1, \tag{1.44}$$

and that even more stringent requirements exist for the two and three dimensional versions. For good spatial accuracy Δx must be small, but this limits the size of Δt. Soon enough, the scheme becomes uselessly slow with even moderate problem requirements.

With steady state problems, stability is not an issue. And with transient problems, the stability issues can be addressed by using backward or central difference approximations in time.[5] The common feature here, however, is that the resulting algebraic systems are no longer diagonal, and can require slower and more complicated solvers. In the special case of one spatial dimension, the implicit and semi-implicit methods produce a *tridiagonal* matrix [30], and can therefore be solved relatively easily. In higher dimensions, both transient and steady state approaches produce less tractable band diagonal matrices. One way around this is with the *alternating direction implicit*

[5]These are termed fully implicit and semi-implicit (or Crank-Nicolson) respectively. The Crank-Nicolson scheme also carries the advantage that it is second order in time.

(ADI) algorithm (*ibid.*).[6] This effectively breaks the process into two or more one dimensional problems that are solved in turn. The central difficulty of finite difference remains, however: that of boundaries.

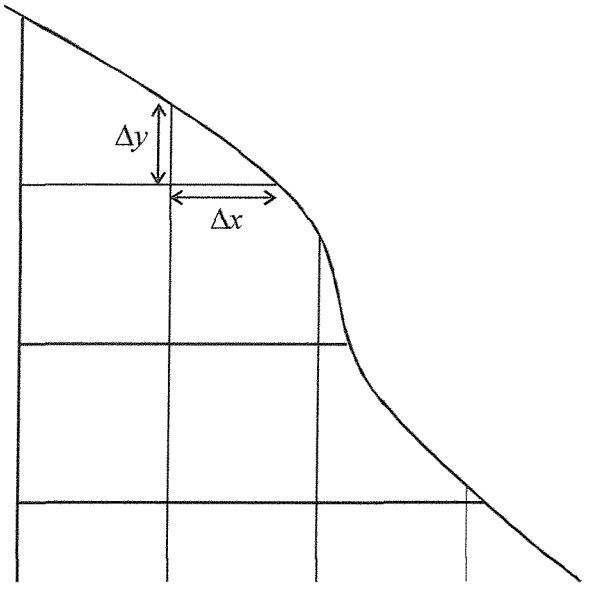

Figure 1.3: A two dimensional finite difference grid in a domain with a complicated boundary. Specific Taylor series approximations are required near the boundary for Neumann or Robin BCs, as well as higher order Dirichlet boundary approximations.

Figure 1.3 illustrates the problem with imposing boundary conditions in finite difference. To a first order approximation a Dirichlet condition can be imposed at a grid point near to the boundary. Lower order boundary conditions can pollute a higher order solution, however, and to be sure of second order spatial accuracy second order boundary conditions are needed, requiring separate Taylor series analyses. The case of Neumann or Robin conditions is even worse: Taylor series are necessary for any approximation at all. A boundary that irregularly intersects the grid therefore poses a difficult

[6] Others exist too: hopscotch, Du-Fort Frankel, fast implicit, strongly implicit, to name some of them—see Britz [29].

challenge. Should one wish to raise the order of the method everything changes again.

For one dimensional problems (that is, one spatial dimension), where only two boundary points can ever exist, finite difference is a reasonable method, usually in its Crank-Nicolson form (although a popular commercial package [26]) uses fully implicit FD). For higher dimensional problems with very simple boundaries ADI could be appropriate, but is probably not worth the effort, as the simulation program would be very specific to the system for which it was designed. The essential difficulty is that the formulation is dependent on geometry, and geometry can vary a great deal. For this reason, despite its initial attractiveness, FD lacks the geometric flexibility to tackle many problems in electrochemistry, and we consequently discard it. We do not consider further, therefore, the vast body of literature on the subject, comprehensively referenced in, for instance, [31].

One rider should be attached to the above conclusions. They hold for spatial discretisation, but for transient simulations that use finite element for spatial discretisation, finite differences *are* usually employed for time discretisation. Some of the same schemes and stability concerns reappear there, but even this type of temporal discretisation can be thought of in a finite element framework, and doing so suggests some potentially superior alternatives [16]. Some of these issues are discussed in Chapter 6.

1.3.2 Finite Element

The finite element method (FEM) is a considerably more modern technique than finite difference, and requires more mathematics to formulate. As a consequence it has been much less used in electrochemistry, but it is gaining popularity. It is possible to place FEM, FDM and BEM (described below) in a single *minimum weighted residual* (MWR) framework, but in the case of FDM this is rather contrived, in the sense that the weight "functions" are not strictly functions at all. It is worth remembering, however, that in

a sense finite difference is a special case of finite element, so we can never expect finite difference to be superior, except in its simplicity.[7]

A full derivation of the finite element method will be given in Chapter 2; here it is enough to see the initial idea without its detailed consequences. Consider the steady state equation for the field u:

$$\nabla \cdot (a\nabla u) + qu = -f \ , \tag{1.45}$$

where a, q and f are known scalar functions of position defined over the domain Ω with the boundary Γ, constrained by the Dirichlet boundary conditions

$$u = \alpha \qquad \text{on} \qquad \Gamma_D \tag{1.46}$$

and the Robin boundary conditions

$$\frac{\partial u}{\partial n} = \alpha u + \beta \qquad \text{on} \qquad \Gamma_R \ . \tag{1.47}$$

If u satisfies (1.45) then it must also satisfy

$$\int_\Omega w[\nabla \cdot (a\nabla u) + qu + f]d\Omega = 0 \tag{1.48}$$

for any function w. (Here w is the *weight function* : the "W" of MWR.) Assuming the Dirichlet and Robin conditions of equations (1.46) and (1.47) are also imposed, this is an equivalent statement of the problem. So instead of approximating the derivatives of (1.45) as in FD, one can approximate the integral in (1.48).

Since numerical differentiation is known as a difficult and inaccurate procedure generally [30], and numerical integration is much easier (*ibid.*), one might intuitively prefer this idea. But the real advantage is apparent in the second big idea of FE: Ω is broken (meshed) into simple shapes (e.g. triangles), called *elements*, and the field in each approximated by a simple (e.g. linear) function. Unlike FD, no mention is made of axes, and the spacing

[7]It should also be born in mind that even with Galerkin weighting, certain uniform finite element meshes yield the same matrix as simple finite difference discretisations [32].

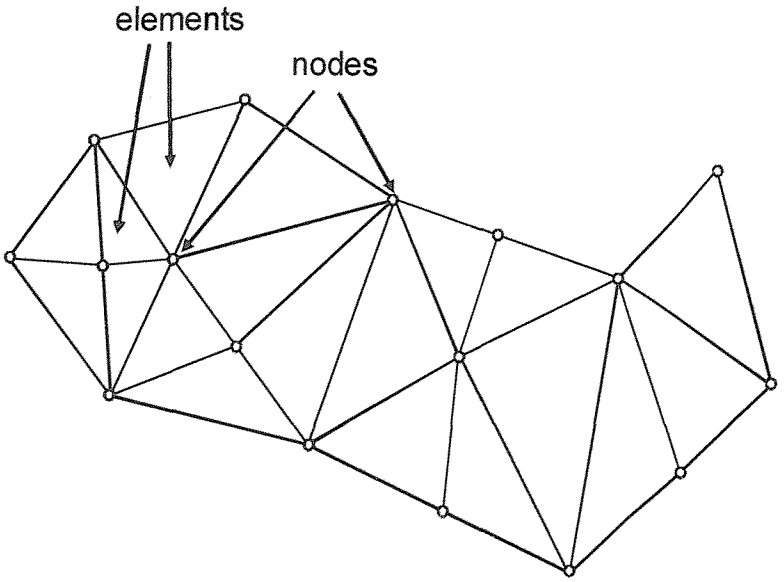

Figure 1.4: A finite element mesh for an irregular two dimensional shape. The elements are linear triangles, which means their nodes are at the corners of elements—higher order elements have additional nodes elsewhere. A few example elements and nodes are labelled. Note that elements can be placed arbitrarily in an unstructured manner.

of adjacent mesh points is not integral to the formulation, so any geometry can in principle be imposed. In Figure 1.4 a possible mesh is shown for an irregular shape difficult or impossible to treat with finite difference.

There are numerous formulations of finite element, principally differing in the weight functions used. Unless otherwise noted, the most standard version, the Galerkin formulation, will be used here. This has the special advantage that with *self-adjoint* problems (see Appendix B on page 263), the resulting matrix is symmetric, which happens to be advantageous. There are also related, more theoretical, advantages.

The idea behind (1.48) is clearly not contingent on any particular form for the PDE. It also happens that later in the formulation (see Chapter 2) no other assumptions about the PDE are made. So, as would be expected

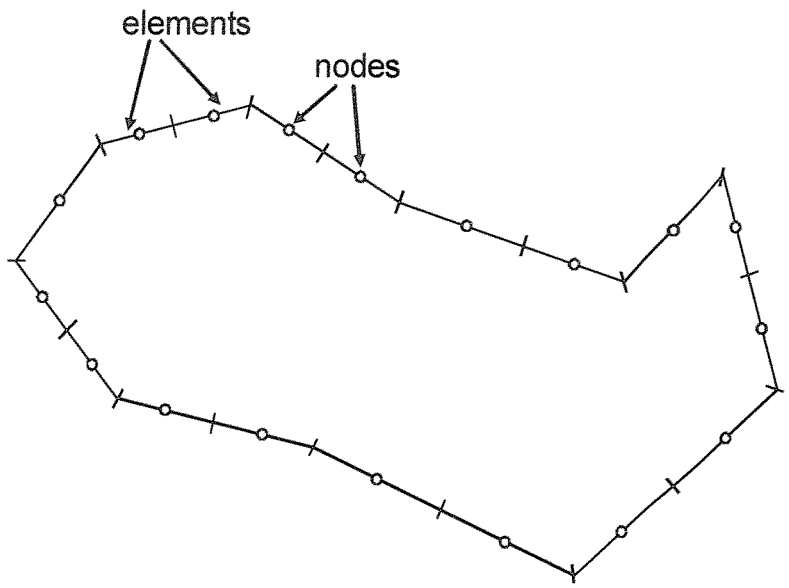

Figure 1.5: A boundary element mesh for the same boundary as Figure 1.4 on the page before. The elements reside entirely on the boundary, and are demarcated by perpendicular lines. For the simplest elements, nodes are at the centres of elements. Again, a few example elements and nodes are highlighted.

from finite element being in a sense a generalised version of finite difference, it can also in principle deal with entirely arbitrary manifestations of (1.19), including convective, reaction and transient terms.[8] The practicalities of this will be discussed later.

A primary advantage of FE is the flexibility of its mesh.[9] Firstly, this means that non-axially aligned and even curved domains can be modelled with relative ease without changing the simulation program. Secondly, the density of mesh nodes can be easily increased near the boundary singularities described in §1.1.5, allowing efficient handling of them. (Expanding grid finite difference has long been used for this purpose, but refinement is usually tied to an axial direction, making it inflexible and as a consequence possibly inefficient.)

Among other advantages, finite element allows the possibility of estimating the error of the approximate answer—a crucial aspect that has so far gone unmentioned in this work, but which will prove essential later. This capability has seemingly not been developed for finite difference. Harriman *et al.* show [33–38] that the error for currents calculated from finite element simulations can be controlled by adaptively adjusting the mesh. This idea is used extensively in Chapter 3. Finally, it is apparently easier to prove theorems in relation to finite element than finite difference. A sense of this can be found in [32], where the relatively stringent assumptions required for simple proof of finite difference's convergence are given. Very roughly, because finite element deals with "weak" solutions (see 2.1.1 on page 45), with lower differentiability requirements, it is more permissive than finite difference, with its Taylor series basis.

[8] In fact, as noted in §1.3.1, transient terms are usually handled differently, with finite differences, but they can be treated in the same way as spatial derivations.

[9] Here we use *mesh* to denote a generalised version of the regular grid used in FD, but the distinction is blurred when expanding grid FD is compared with structured mesh FE. Perhaps the true difference is that unstructured (see later) FE meshes, which have no analogue in FD, are also used.

The disadvantages of FEM lie in its complexity. The most difficult practical aspect of its implementation is indubitably mesh generation. Certainly, if a good mesh is available, finite element will be an efficient method; but generating such a thing, particularly in three dimensions, is not a trivial undertaking. In the work discussed later the two dimensional meshing problem is addressed with reasonable success, but the general problems of meshing are not to be dismissed lightly. It should be remembered, though, that finite difference offers no solution to this either: the issue of how to efficiently mesh an arbitrary domain never arises because it is not possible.

It is fairly clear then, that, provided the slightly more abstract formulation of FE is acceptable, there is no compelling reason to use FD: the geometric flexibility and error estimation capabilities of finite element are better-suited to handling outstanding electrochemical problems. The same clear-cut dismissal cannot be made of the next alternative that we discuss.

1.3.3 Boundary Element

Boundary element uses the same idea as finite element: it solves the PDE integrated over a weight function. But instead of carving up the domain into pieces, and using a different weight function in each, it uses a specially chosen global weight function to eliminate the domain integral in an expression similar to (1.48). In fact, it eliminates the domain integral from the *adjoint* expression.

There is no space to fully describe this, but an example gives the idea. Assume we wish to solve the weighted Laplace problem:

$$\int_{\Omega} w\nabla^2 u d\Omega = 0 \,, \tag{1.49}$$

We can employ a trick, similar to one used later in the finite element formulation, to rewrite the integral as

$$\int_{\Omega} \nabla^2 w u d\Omega + B = 0 \,. \tag{1.50}$$

where B denotes boundary terms (see (2.10) for the sort of expression).

If we wish to eliminate the domain integral in (1.50), one obvious approach is to use a w that identically satisfies the governing equation. As has been noted, functions that satisfy PDEs (but not their boundary conditions) are not hard to find. In the two dimensional Cartesian form of our example we need w to satisfy

$$\nabla^2 w = \frac{\partial^2 w}{\partial x^2} + \frac{\partial^2 w}{\partial y^2} = 0 \; . \tag{1.51}$$

A host of possibilities suggest themselves, some more trivial (and useless) than others:

$$w = 0 \tag{1.52}$$

$$w = x \tag{1.53}$$

$$w = y \tag{1.54}$$

$$w = x^2 - y^2 \; , \text{ etc.} \tag{1.55}$$

A whole variety have been used with BEM—and indeed the best ones are a matter of dispute—but whatever the choice the domain integral essentially disappears, leaving solely boundary terms. This has profound implications for the meshing—see Figure 1.5 on page 29 for a comparison with finite element. In Figure 1.5 it can be seen that two dimensional problems require only the discretisation of a *line* rather than a potentially irregularly shaped *area*.

Although it is relatively unimportant for our discussion, it should be mentioned that most BE work in electrochemistry—and most BE work generally—has been done with rather less arbitrary weight functions. Instead of weight functions that satisfy the governing equation everywhere, *fundamental solutions* (also called *singular weight functions* [16]) are used that obey it *almost everywhere*. In our example,

$$\nabla^2 w = \delta(x - \xi_1)\delta(x - \xi_2) \; . \tag{1.56}$$

(δ here is the Dirac delta functional [13].) The reasons for the different

approaches are not important here; instead we note the obvious advantages and disadvantages over FEM.

The greatest advantage of BEM is that the disappearance of the domain integral means that no domain meshing need be done: only the boundary need be discretised, meaning the meshing challenge in three dimensions, particularly, becomes far easier due to a reduction in dimensionality by one. Another is that high accuracy can often be achieved with few elements, and the algorithm can be more efficient than finite element. This is not altogether surprising, since some of the nature of the analytical solution has been incorporated into the method.

The benefit of a reduced number of degrees of freedom is not as clear cut as the size of the matrix suggests, however, as with BEM it is not sparse or symmetric or positive definite, necessitating computationally expensive general matrix solvers. If dense matrix LU factorisation is used to solve the linear system arising from discretisation, as it usually is, the computational effort scales with the cube of the number of elements (see Appendix C). This imposes a practical limit on normal computers of a few thousand degrees of freedom. Nor is there, to the author's knowledge, any equivalent of multigrid, which allows in finite difference and element, in some cases, solution in $\mathcal{O}(n)$ (where n is the number of nodes) time. Finally, an important phenomenon used in finite element error analysis, superconvergence, is also apparently absent in boundary element.

Whatever the net advantages of BEM may be, they come at a price in flexibility: the obvious disadvantage—not shared by FEM—is that the weight function depends on the governing equation. If singular weight functions are used, as they usually are in electrochemistry, then a fundamental solution must be found whenever the formulation, and hence the mechanism or transport regime, changes. This is a major limitation of the basic boundary element method. Further difficulties become apparent even when a fundamental solution is available, or non-singular weight functions are used. The neat elimination of domain integrals in the derivation of the adjoint

expression (1.50) commonly falls down when extra terms—time derivatives, convective terms, constant sources/sinks, etc.—appear. The BEM practitioner must constantly walk a tight-rope to preserve its main advantage: the lack of a domain integral.

In some cases, the problems are surmountable. If the convective velocity vector is constant, for instance, there does exist a usable fundamental solution. But fundamentally, as it stands, the method is too inflexible to be generally useful in electrochemistry. This is where the Dual Reciprocity Method (DRM) and its variants come in.

Apparently the only hope for BEM as a generally useful electrochemical technique lies in modifications like DRM. Essentially they separate out non-Laplacian terms and approximate them, to allow solution of an equation of the form (in two dimensions):

$$\nabla^2 u = f(u, x, y, t) \ , \tag{1.57}$$

where f is an arbitrary function, allowing time and spatial derivatives. So convective, transient, and mechanistically more complicated problems can be solved. Unfortunately DRM is under-researched, and some recent findings in electrochemistry [18] suggest that its additional complications and approximations destroy convergence in many important cases. While the transient case has met with some success [39], convection appears difficult to handle [18]. There always remains the difficulty of putting nodes in the right place on the boundary; and similarly with the internal nodes that DRM demands.

The last point illustrates a recurring problem with numerical methods. In all techniques where a grid or mesh is used to discretise the domain, the effect of the location of nodes on the accuracy of the result is known to be pronounced. Near boundary singularities it is usually necessary to have a high density of approximating nodes to capture the nature of the rapidly varying concentration field. Much effort in finite difference and non-adaptive finite element has been poured into finding functions to describe, *a*

priori, the locations of nodes. Recent BEM work persists with this. Research in electrochemistry and other areas has shown that FEM meshes can be effectively tailored to the problem using *error estimators* that guide mesh refinement. This is something so far largely undeveloped with BEM. For this and the other reasons, we conclude that FEM remains the most attractive option.

1.3.4 Random Walk

Unlike the numerical methods sketched above, random walk (RW) is not a standard means of solving partial differential equations. Nonetheless, a surprising amount of electrochemical literature exists on using RW to solve diffusion problems. On the surface, it appears to offer an entirely non-mathematical route to modelling numerical mass transport. Virtually none of the papers on the subject refers to the diffusion equation, and they certainly do not use Taylor series, or weighted integral formulations. Instead, random walk is justified solely with the familiar physical picture of diffusive mass transport: random molecular motion. Purely from the perspective of simplicity, then, one might wonder whether random walk offers something that finite difference and finite element and boundary element do not.

It is not possible to give a standard mathematical formulation of the random walk simulation technique, as various algorithms are used—at least not without delving into stochastic differential equation theory [40]. Instead we give below a brief sketch of the previous work in this area.

Previous Work

An early paper [41] describes the simple random walk algorithm used in most subsequent papers. In a one dimensional problem the motion of a diffusing molecule is described by random, equally probable steps to the left or right, curtailed by reflecting or absorbing barriers corresponding intuitively to "zero flux" or "zero concentration" boundary conditions respectively. The article

considers simulation particles as direct representations of physical molecules, which explains its mooting of a "more realistic picture" of reality, where particles' step lengths are variable, being drawn from a Gaussian distribution—an approach also adopted by a recent paper [42]. In all other electrochemical papers mentioned, steps from a distribution with a fixed length are taken.

Much early electrochemical use of random walk simulations was in fractal electrode investigations, a useful introduction to which is [43]. Witten and Meakin [44] used a lattice-based random walk to study the Hausdorff dimensionality of deposits in diffusion-limited aggregation. In one model, particles diffused simultaneously; in the second particles diffused and deposited one at a time. Both yielded qualitative data of an unknowable accuracy.

Nyikos and Pajkossy and co-workers published a number of papers [45–48] with similar random walk simulations yielding semi-quantitative results (arbitrary constants remained).

Voss and Tomkiewicz [49] describe a random walk model of diffusion-limited aggregation with a "sticking coefficient" incorporating the effect of potential. They demonstrate a qualitative relation between simulation and experimental parameters. Their model was adopted by Fanelli *et al.* to study growth of mono- and multi-layers [50], stripping voltammetry [51], and diffusion to fractal electrodes, in all cases yielding qualitative insight.

Further fractal-related work has appeared from Trigueros *et al.* [52] as well as Sapoval *et al.* [53, 54].

All of these mainly fractal-devoted papers claim to give insight into the relation between various parameters and the morphology of the resulting deposit or shape of the current transient, but do not yield numerical predictions of experimental results. Partly, of course, this is because computers were too slow to run full three dimensional simulations. Initially, nucleation and growth were treated in the same way [55], but Nagy *et al.* [56, 57], followed by others [42, 58], applied random walk to nucleation and growth in three dimensions, producing numerical answers limited only by model simplifications and statistical variance.

Difficulties with Random Walk

It is impossible to go into detail here about the various algorithms. A number of simple points can be made, however.

Firstly, while claims are made in various papers that the random walk method is "grid free", it is usually not. Where particles move a fixed step length in one of four or six directions (depending on dimensionality), they are of course moving on an implicit grid, much the same as that used in finite difference. In fact, by considering the difference equations governing movement probabilities one can show that many simulations are in fact *exactly equivalent*, in the limit of an infinite number of particles, to the inefficient and inflexible explicit finite difference scheme described above. This was apparently not realised by the cited authors. (The relation is fairly easy to prove from equations given by Cox and Miller [59].)

The difference, of course, is that there never is an infinite number of particles, which raises the next point. The rate of convergence of all Monte Carlo methods is $\mathcal{O}(1/\sqrt{N})$, where N is the number of samples (typically in this case, the number of particles).[10]. Since N is at best linearly proportional to computational effort, this means that the accuracy scales very poorly with simulation time. From the previous paragraph, however, a crucial point should be made: the "accuracy" of which we speak is of the approximation to the *finite difference approximation* of the true solution. At best, with an infinitely fast computer, we would achieve an explicit finite difference approximation of limited accuracy.

It should be clear, then, that grid-based random walk method is largely useless, unless the accuracy of deterministic methods is simply unattainable due to memory constraints (an unlikely scenario with an efficient mesh). Monte Carlo methods as a whole, however, cannot be dismissed so easily. Numerous variants exist outside of electrochemistry, notably so-called

[10]We do not have convergence in the usual sense, but a reasonable substitute definition is the standard deviation of results. This decreases with $\sqrt{N}$, by the definition of variance.

floating random walk as described by Muller [60] and Haji-Sheikh and Sparrow [61], and various other versions described by Sabelfeld [62]. It may be that these could prove useful for some special cases, but they certainly bring new problems lessening their attractiveness for many simulations. Perhaps their ultimate purpose will be in simulations with random parameters (e.g. with a randomly perturbed potential). It should also be added that, among other techniques, "quasi-random" sequences have been used to increase the convergence rate, but these bring their own theoretical difficulties (see, e.g., Ogawa and Lécot [63]).

For now, in the absence of any compelling demonstration of their usefulness, and without adequate documentation, random walk methods remain a curiosity. The best option, then, remains finite element.

1.4 Dimensionless Quantities

To avoid numerical simulations being tied to one real-world situation, they are usually conducted with dimensionless quantities: various real-world dimensional parameters can be separated out beforehand from the problem statement, the simulation can be completed, and the dimensional quantities re-incorporated to give results applicable to a range of experiments.

In the steady state simulations conducted hereafter, a dimensionless concentration, u, normalised with respect to the bulk concentration c^*, or some related quantity, will be used. For instance, an obvious choice is

$$u = \frac{c}{c^*} \,. \tag{1.58}$$

Typically, then, the concentration in the domain under simulation will vary from zero to unity. If, for example, material is consumed under pure diffusion control at an electrode, $u = 0$ there, while far away, towards the bulk, $u \to 1$. Where more than one species is being modelled it may be preferable to use some combined definition of concentrations. The details vary according to the mechanism.

Dimensionless spatial coordinates can also be of use, but the exact normalisation will vary. It is often useful to define the spatial units in terms of some characteristic length of the simulation domain, l. For a microdisc this characteristic length is the radius of the microdisc; for a microband, the microband width. Using this idea one can define the dimensionless coordinates for Cartesian domains:

$$X = \frac{x}{l} \qquad Y = \frac{y}{l} \; ; \tag{1.59}$$

and for cylindrical domains:

$$R = \frac{r}{l} \qquad Z = \frac{z}{l} \; . \tag{1.60}$$

Since there is little chance of confusion, lower case letters will signify dimensionless coordinates when used with dimensionless concentrations in the following chapters.

The results of a microelectrode simulation thus conducted apply to all sizes of microelectrode, provided any other features of the domain scale with it (which is the case with an isolated electrode in an infinite domain). Where other domain features also appear, such as with SECM simulations, we still normalise with respect to the electrode size, but the relative dimensions of other features limit the applicability of the results.

Finally, we also work in terms of a dimensionless current, normalised with respect to a related analytical result. Here we take the example of the microdisc. The analytical, steady state, n electron diffusion controlled current to a microdisc of radius a is given by the familiar formula [12]

$$i = 4nFDc^{*}a \; . \tag{1.61}$$

Using Fick's first law[11] we define the normalised simulation current for a

[11] See equation 1.3 on page 4.

microdisc-type simulation as

$$i_{norm} = \frac{nFDc^* 2\pi \int_{r=0}^{r=a} \left.\frac{\partial c}{\partial z}\right|_{z=0} r dr}{4nFDc^* a} \tag{1.62}$$

$$= \frac{\pi}{2} \int_{R=0}^{R=1} \left.\frac{\partial u}{\partial Z}\right|_{Z=0} R dR \tag{1.63}$$

A comparable expression is used for microband simulations.

1.5 Summary

In §1.1.3 the problem that we set out to solve has been formulated. This boils down, in our case, to solving (1.20), in some form or another, in a domain reflecting the geometry of the cell in which we are interested.

It is well known that seeking analytical solutions to (1.19) is generally a fruitless task, principally owing to the lack of symmetry of many important domains. When turning to numerical solution methods, one finds that handling the boundary singularities (§1.1.5) intrinsic to microelectrode geometries is one of the major difficulties faced, whichever discretisation method is used.

Examining in turn four of the most common simulation algorithms in electrochemistry (finite difference in §1.3.1, finite element in §1.3.2, boundary element in §1.3.3 and random walk in §1.3.4) it is possible to make some clear decisions. Finite difference, despite its deep entrenchment, appears to have been applied to most electrochemical problems within its range—while suitable for one dimensional domains and other simple problems, it lacks the geometric flexibility to tackle outstanding problems; even the simulation of a tapered SECM [64] tip,[12] for example, is practically beyond it. Conversely, the basic form of boundary element, while exhibiting a level of geometric flexibility greatest of all deterministic methods, cannot incorporate many important governing equation terms; in their generality, convective and homogeneously reactive and transient problems defeat it. On the face of it,

[12] We do this with adaptive finite element in Chapter 5.

dual reciprocity BEM solves these difficulties—and retains the geometric flexibility—but the results of its application have not met expectations, and with the current amount of research it is difficult to know if they will. To this objection may be added two other criticisms: BEM, and DRM BEM in particular, has had little work done on error adaptivity, as frequently practised with FEM; and secondly the method in its entirety is very poorly documented compared with FEM.

Finite element may be viewed as a middle way between finite difference and boundary element. Outside of electrochemistry, it is probably documented as well as, if not better than, finite difference, and it shares FD's equation flexibility; but it also holds some of the promise for complete geometric flexibility that BEM clearly possesses (albeit with a more difficult meshing challenge). It comes in innumerable versions, some of which we might expect to be applicable to electrochemical problems. One of these, as documented in the important work of Harriman *et al.* [33–38], is adaptive finite element where, crucially, the *current* error can be controlled, rather than some other arbitrary error measure. It is hard to see a better hope of flexible deterministic electrochemical problem solver, and we pursue this goal in Chapters 2 and 3.

The relation of random walk to other simulation methods is rather more difficult to explain with reference to preexisting literature, whether or not in electrochemistry. On the surface it offers promise of solving problems in a different way, and of solving problems intractable with other methods. But its inefficiency, and the paucity of documentation, make it difficult to recommend at this stage. Certainly the method as used so far in electrochemistry is difficult to justify. It may be that some other algorithm, perhaps a non-grid based one, possibly using quasi-random sequences, may prove useful in the future. Certainly if randomly perturbed systems become of interest, it will be worth considering. For now, however, for the bulk of systems of practical interest, finite element is clearly more attractive.

Finally, in §1.4 the advantageous changes wrought by use of dimensionless

quantities are seen. Clearly there is no reason to perform simulations without them, so we shall adopt them exclusively.

The next step, then, is to describe the adaptive finite element method in detail. First, in Chapter 2, the basic finite element formulation is described, followed by the specifics of a novel adaptive algorithm in Chapter 3.

Chapter 2

Finite Element Theory

Since the fundamental FE theory is more complicated than that of finite difference, and because finite element is less common in electrochemistry, a presentation of the basic formulation is given in this chapter. This also allows the specifics of our implementation to be made clear. The variational derivation [16, 65] is not used as it is precludes convection, and requires use of the calculus of variations;[1] instead the more general Galerkin weighting approach is used. While this version of finite element for purely diffusive, self-adjoint, problems appears in numerous basic texts (e.g. [65, 66]), the discretisation applied to convective problems is less well documented.

The Galerkin theory is in principle applicable to almost all electrochemical problems, steady state or transient, two or three dimensional, purely diffusive or diffusive-convective, linear or non-linear. In practice, however, certain cases may require special care or alterations. Specifically, where convection dominates mass transport, stabilisation schemes are often used to eliminate the unphysical oscillations that arise when the Galerkin scheme is applied. An attempt is made to explain these in this chapter, although the area is complicated. Essentially these amount to using modified weight

[1]Note that where a variational derivation exists, it gives the same result as Galerkin minimum weighted residual (see later). Note also that, as exemplified in Chapter 3 and elsewhere, the variational approach cannot be entirely ignored, if only as a theoretical tool.

functions, so much of the theory remains the same, and the modifications may be considered separately as an addition to the basic scheme.

Where the shape of elements and other specifics enter the theory, the formulation is specialised to two dimensional steady state problems, which of course limits its applicability. Nonetheless, it covers some important cases that we wish to study in detail.

Substituting element-specific expressions into the general formulation produces an algebraic system that must be solved in order to recover the concentration field. If, as here, the problem is linear, this system is most conveniently written as a matrix equation. The efficient assembly of the matrix requires some particular strategies, and some of the less obvious (or at least less well documented) issues relating to this are raised. After assembly, the linear system must be solved, taking into account the special matrix properties that allow efficient practical solvers. Since this is a standard problem, only a brief sketch of the theory is given in Appendices C and D, but some computational aspects are related at the end of this chapter.

2.1 The Galerkin FE Formulation

Some of what follows in this section is not specific to the Galerkin formulation. Indeed much of it is generally applicable to any minimum weighted residual method. This is useful, as it happens that other formulations can be superior in certain situations. However, the Galerkin method is probably the best for the case of self-adjoint problems, which for our purposes means problems without a convective component to mass transport.

Leaving aside radical alterations of finite element such as the *element free* methods surveyed in the book by Zienkiewicz and Taylor [16], finite element methods always require a mesh of broadly the same nature—as exemplified in Figure 1.4 on page 28. In § 3.1 on page 82 our approach to this mesh generation question is presented in detail. Until then we assume we have a

suitable mesh available.

2.1.1 The Weak Formulation

As a starting point we set out to find $\bar{u} \approx u$, the approximate solution of the partial differential equation

$$\nabla \cdot (a\nabla u) - \mathbf{v} \cdot \nabla u + qu = f \ , \tag{2.1}$$

where a, q and f are known scalar functions of position defined over the domain Ω (which has the boundary Γ), and $\mathbf{v}$ is a known vector field. In addition, we impose the Dirichlet boundary conditions

$$u = \alpha \qquad \text{on} \qquad \Gamma_D \ , \tag{2.2}$$

and the Robin boundary conditions

$$\frac{\partial u}{\partial n} = \alpha u + \beta \qquad \text{on} \qquad \Gamma_R \ . \tag{2.3}$$

Although, as is shown in the previous chapter, the Dirichlet condition can be thought of as a limiting case of the Robin condition, it is implemented in an entirely different manner in finite element, so the two are separated. Neumann conditions, on the other hand, are treated as particular cases of Robin conditions with $\alpha = 0$.

Equation(2.1) is a form of (1.20) specialised to first order homogeneous reactions. For problems involving multiple species, several such equations would need to be solved. It incorporates diffusion (in the Laplacian-type term—a is essentially the diffusion coefficient, although this appears to be given sometimes in differentiated form, outside the divergence operator), convection of velocity $\mathbf{v}$, a first order homogeneous reaction (in the qu term), and a homogeneous reaction whose rate is independent of u (in the f term). Thus it can model several mechanisms, including E ($q = f = 0$), CE ($q = 0, f > 0$), and EC' ($q < 0, f = 0$). Alone it represents the most

general first order steady state system that can be modelled with a single concentration field.

As mentioned in the previous chapter, if (2.1) is true, then it must also be so that

$$\int_\Omega w[\nabla \cdot (a\nabla u) - \mathbf{v} \cdot \nabla u + qu - f]d\Omega = 0 \tag{2.4}$$

for any weight function w. It is here that the term "minimum weighted residual" is explained. If we substitute an approximate solution $\bar{u}$ into the version of the governing equation (2.1) rearranged to equate to zero,

$$\nabla \cdot (a\nabla u) - \mathbf{v} \cdot \nabla u + qu - f = 0\,, \tag{2.5}$$

we can expect that it will not be completely satisfied, giving a *residual* R:

$$\nabla \cdot (a\nabla \bar{u}) - \mathbf{v} \cdot \nabla u + q\bar{u} - f = R\,. \tag{2.6}$$

We cannot ensure that R is zero at the same time as satisfying the boundary conditions, because that would require knowing the solution u, so instead we attempt to minimise R by multiplying by a number of weight functions w, and insisting that the results equal zero. This will become more apparent as the derivation proceeds.

Multiplying through by the w factor, we can break the integral into several pieces:

$$\int_\Omega w\nabla \cdot (a\nabla u)d\Omega - \int_\Omega w\mathbf{v} \cdot \nabla u d\Omega + \int_\Omega wqud\Omega = \int_\Omega wfd\Omega\,. \tag{2.7}$$

So far this is still a general weighted rewrite of (2.1). The first concession to approximation made is in reducing the order of differentiation in the integrand of the first of the integrals on the right-hand side. The result is termed the *weak form* of the problem [17], since the differentiability requirements on u are reduced. It is noted, however, that the weak form can produce more physically realistic results (*ibid.*), and fundamentally there is no reason to accord primacy to the PDE formulation.

To achieve the reduction in the order of derivatives a generalisation of the product rule is used:

$$w\nabla \cdot (a\nabla u) = \nabla \cdot (wa\nabla u) - \nabla w \cdot a\nabla u \ . \tag{2.8}$$

In combination with the divergence theorem, this yields a standard result sometimes known as *Green's Theorem* [16] or *Green's first formula* [67].[2] Applying (2.8) to (2.7), produces five integrals:

$$\int_\Omega \nabla \cdot (wa\nabla u)d\Omega - \int_\Omega \nabla w \cdot a\nabla u d\Omega - \int_\Omega w\mathbf{v} \cdot \nabla u d\Omega + \int_\Omega wqud\Omega = \int_\Omega wfd\Omega \tag{2.9}$$

The problem is now in weak form.

There are at least three reasons for removing second order derivatives from the formulation. The first is that doing so allows element shape functions (see later) of lower order. Without this change the approximate solution $\bar{u}$ would have to be C^1 continuous.[3] Rewriting in this way has the advantage that the solution need only be C^0 continuous, removing the need for inter-element derivative continuity. The second reason is that doing so in combination with Galerkin weight functions introduces symmetry that manifests itself advantageously in the system matrix. Thirdly, it allows Robin boundary conditions to be built into the formulation. This is an important change in view of the difficulties of doing the same with finite difference (see §1.3.1).

The next step is to convert the leftmost integral of (2.9)—an integral over the domain—into a boundary integral. (This would have been combined with the previous step had the pre-packaged Green's Theorem been used.) Recognising this as of the form of the left-hand side of divergence theorem statement—see Equation 1.10 on page 8—we can write

$$\int_\Gamma wa\frac{\partial u}{\partial n}d\Gamma - \int_\Omega \nabla w \cdot a\nabla u d\Omega - \int_\Omega w\mathbf{v} \cdot \nabla u d\Omega + \int_\Omega wqud\Omega = \int_\Omega wfd\Omega \ . \tag{2.10}$$

[2]Sources appear to differ on terminology: alternative nomenclature includes *Green's first identity*; and *Green's Theorem* is also used to denote another result.

[3]C^1 continuity means that all the function's first order derivatives exist and are continuous. See, for instance, [13].

(Remember here that the differentiation $\frac{\partial}{\partial n}$ is along the outward normal.)

This change elegantly incorporates the Robin boundary conditions. It happens, as is seen later, that it is relatively easy to restrict $\bar{u}$, the approximate solution, so that it satisfies (2.2). At this stage we solely seek to impose (2.3), so we set $w = 0$ on Γ_D, whence the slightly modified form:

$$\int_{\Gamma_R} wa\frac{\partial u}{\partial n}d\Gamma - \int_\Omega \nabla w \cdot a\nabla u d\Omega - \int_\Omega w\mathbf{v}\cdot\nabla u d\Omega + \int_\Omega wqud\Omega = \int_\Omega wfd\Omega\ . \tag{2.11}$$

Since there is a Robin condition on Γ_R, $\frac{\partial u}{\partial n}$ is known on that boundary, and can be substituted from (2.3) to give a total of six integrals:

$$\int_{\Gamma_R} aw\alpha u d\Gamma + \int_{\Gamma_R} aw\beta d\Gamma - \int_\Omega \nabla w \cdot a\nabla u d\Omega - \int_\Omega w\mathbf{v}\cdot\nabla u d\Omega + \int_\Omega wqud\Omega = \int_\Omega wfd\Omega\ . \tag{2.12}$$

No real approximations have yet been made, as signalled by (2.12) referring to u rather than $\bar{u}$; but a lower order of differentiability of u has been conceded by adopting the weak formulation. The difference is subtle: all "strong" solutions—solutions to the PDE form of the problem—are admissible by the weak formulation; it is just that (2.12) allows other, weak, solutions that do not formally satisfy (2.1). This distinction need not concern us.

To proceed further we must adopt a particular type of weight function. With the Galerkin formulation this weight function is intimately tied to the shape function used for representing the solution, as described in the next section.

2.1.2 Galerkin Weighting

We define the approximate solution to be

$$\bar{u} = \sum_{j=1}^{n} N_j a_j\ , \tag{2.13}$$

with a_j the (as yet undetermined) value of $\bar{u}$ at node j, N_j the *shape function* associated with that node, and n being the number of nodes. If $\mathbf{N}$ is viewed as a row vector of nodal shape functions, and $\mathbf{a}$ as a column vector of unknowns, (2.13) may be written more compactly as

$$\bar{u} = \mathbf{Na} \, . \tag{2.14}$$

In this we follow Zienkiewicz and Taylor [16].

Specific types of shape functions will be presented below. For the moment we state that N_i is only non-zero in the locality of node i, dropping away to zero by the time any neighbouring nodes are reached. This is true of the linear shape functions of the next section.

Substituting (2.14) into (2.12),

$$\int_{\Gamma_R} aw\alpha \mathbf{N}\mathbf{a} d\Gamma - \int_\Omega \nabla w \cdot a \nabla \mathbf{N}\mathbf{a} d\Omega - \int_\Omega w\mathbf{v} \cdot \nabla \mathbf{N}\mathbf{a} d\Omega + \int_\Omega wq\mathbf{N}\mathbf{a} d\Omega = \int_\Omega wf d\Omega - \int_{\Gamma_R} aw\beta d\Gamma \, . \tag{2.15}$$

This restriction on $\bar{u}$ confines the nature of the solution significantly—it has moved from an infinite to a finite dimensional space.

In the Galerkin formulation one chooses to make $\bar{u}$ satisfy (2.15) with n different weight functions W_i, equalling each of the element shape functions N_i, except when the node i lies on Γ_D. On the Dirichlet boundary the value of u is predetermined, so a degree of freedom is lost, and the weight function at such nodes is set to zero (but see below). Thus

$$\int_{\Gamma_R} aN_i\alpha \mathbf{N}\mathbf{a} d\Gamma - \int_\Omega \nabla N_i \cdot a \nabla \mathbf{N}\mathbf{a} d\Omega - \int_\Omega N_i\mathbf{v} \cdot \nabla \mathbf{N}\mathbf{a} + \int_\Omega N_i q\mathbf{N}\mathbf{a} d\Omega = \int_\Omega N_i f d\Omega - \int_{\Gamma_R} aN_i\beta d\Gamma \tag{2.16}$$

defines n equations for $i = 1 \ldots n$.

The integrands may be expanded in order to see the form of the integrals.

For instance, the first integral of (2.16) is

$$\int_{\Gamma_R} aN_i\alpha\mathbf{N}\mathbf{a}d\Gamma = \int_{\Gamma_R} aN_i\alpha\left(\sum_{j=1}^{n} N_j a_j\right)d\Gamma \tag{2.17}$$

$$= \int_{\Gamma_R} aN_i\alpha(N_1a_1 + N_2a_2 + \ldots + N_na_n)d\Gamma \tag{2.18}$$

$$= \int_{\Gamma_R} aN_i\alpha N_1a_1d\Gamma + \int_{\Gamma_R} aN_i\alpha N_2a_2d\Gamma + \ldots + \int_{\Gamma_R} aN_i\alpha N_na_nd\Gamma \tag{2.19}$$

$$= a_1\int_{\Gamma_R} aN_i\alpha N_1d\Gamma + a_2\int_{\Gamma_R} aN_i\alpha N_2d\Gamma + \ldots + a_n\int_{\Gamma_R} aN_i\alpha N_nd\Gamma \tag{2.20}$$

Owing to the locality of the shape functions, most of these integrals, and the other ones deriving from (2.16), will be zero. Consider equation k of the set $1 \ldots n$. The various integrands would entail multiplication of shape function N_k, or a derivative thereof, by each of the other shape functions $1 \ldots n$, or, again, a derivative thereof. Only if both shape functions were non-zero over the same domain would the result be non-zero. This is in turn defined by the nodal connectivity in the mesh: if the nodes share an edge, their shape functions will overlap, and the respective integral will be non-zero.

For instance, if node k were connected to nodes r, s, t, then only the nodal shape functions k, r, s, t would be non-zero in the same range as N_k, so the integrals would disappear in all other cases. The four series of integrals of the type shown in (2.17) therefore collapse to four terms each, giving, for the

equation defined by the weight function N_k,

$$a_k \int_{\Gamma_R} a\alpha N_k^2 d\Gamma - a_k \int_\Omega a\|\nabla N_k\|^2 d\Omega - a_k \int_\Omega N_k \mathbf{v}\cdot\nabla N_k d\Omega + a_k \int_\Omega q N_k^2 d\Omega +$$
$$a_r \int_{\Gamma_R} a N_k \alpha N_r d\Gamma - a_r \int_\Omega a\nabla N_k\cdot\nabla N_r d\Omega - a_r \int_\Omega N_k \mathbf{v}\cdot\nabla N_r d\Omega + a_r \int_\Omega N_k q N_r d\Omega +$$
$$a_s \int_{\Gamma_R} a N_k \alpha N_s d\Gamma - a_s \int_\Omega a\nabla N_k\cdot\nabla N_s d\Omega - a_s \int_\Omega N_k \mathbf{v}\cdot\nabla N_s d\Omega + a_s \int_\Omega N_k q N_s d\Omega +$$
$$a_t \int_{\Gamma_R} a N_k \alpha N_t d\Gamma - a_t \int_\Omega a\nabla N_k\cdot\nabla N_t d\Omega - a_t \int_\Omega N_k \mathbf{v}\cdot\nabla N_t d\Omega + a_t \int_\Omega N_k q N_t d\Omega =$$
$$\int_\Omega N_k f d\Omega - \int_{\Gamma_R} a N_k \beta d\Gamma \,. \quad (2.21)$$

In spite of its length, the form of (2.21) is fairly simple. Since the only unknowns are a_k, a_r, a_s, a_t, it is a linear relation between them. In general it can be written as

$$\sum_{j=0}^{n} a_j \left(\int_\Omega a\nabla N_i\cdot\nabla N_j d\Omega + \int_\Omega N_i \mathbf{v}\cdot\nabla N_j - \int_{\Gamma_R} a N_i \alpha N_j d\Gamma - \int_\Omega N_i q N_j d\Omega \right)$$
$$= \int_{\Gamma_R} a N_i \beta d\Gamma - \int_\Omega N_i f d\Omega \quad (2.22)$$

(here we have multiplied through by -1 in order to make the diffusive term positive). With n such equations, and n unknowns, there results an n dimensional linear system. As (2.21) suggests, most of the coefficients of a_i will be zero.

One final point remains to be made regarding Dirichlet boundary conditions. Their imposition is discussed below, but we have already said that the weight function is set to zero at such nodes. Therefore (2.22) does not apply, and instead there is a tautologous equation of the form $0 = 0$ for each case where node i lies on a Dirichlet boundary. The rest of the system of equations, however, would still refer to the node's value. This is of course resolved by substituting the known nodal value from the boundary condition, as is detailed presently, but in the interim, during the construction of the linear system, our program in fact ignores Dirichlet node weight function

differences entirely. Thus, the notion of setting $W_i = 0$ at Dirichlet nodes is purely a formal one; in practice one need not be so fastidious.

One of the reasons for choosing the weight functions to be the same as the shape functions might be apparent from (2.22)—the symmetry for self-adjoint problems described below (note that the Dirichlet node zero weight functions do not damage this symmetry). But there is another rationale for the Galerkin weight function choice. If the problem is self-adjoint (see Appendix B), then the $\bar{u}$ so derived will be the closest to the true solution, u, in a certain least-squares sense—the same result as would be derived from the variational formulation [16].

The linear system defined by the equations (2.22) for $i = 1 \ldots N$ may be written more compactly in matrix-vector notation as

$$\mathsf{K}\mathbf{a} = \mathbf{F} \,, \tag{2.23}$$

where

$$K_{ij} = \int_\Omega a \nabla N_i \cdot \nabla N_j d\Omega + \int_\Omega N_i \mathbf{v} \cdot \nabla N_j d\Omega - \int_{\Gamma_R} a N_i \alpha N_j d\Gamma - \int_\Omega N_i q N_j d\Omega \tag{2.24}$$

and

$$F_i = \int_{\Gamma_R} a N_i \beta d\Gamma - \int_\Omega N_i f d\Omega \,. \tag{2.25}$$

K is known as the "stiffness matrix" because of its origin in mechanical applications; and $\mathbf{F}$ as the "force vector" for similar reasons.

Note the dependence of symmetry of K on convection: all terms in the definition of K_{ij} are invariant under the interchange of i and j except that involving $\mathbf{v}$. Since symmetric matrices are generally easier to solve (see Appendices C and D), this is important. The high number of zero coefficients is responsible for the *sparsity* of matrix K, which must be taken advantage of in any efficient solver (see Appendix C). Finally, where the matrix is symmetric it is more easily solved if it is also positive definite. It can be shown, for instance by considering the variational formulation [16, 32], that for the problems under study this does occur, which essentially means that where

convection is absent a conjugate gradient solver may be used to solve the linear system.

2.1.3 The Linear Triangular Element

Until now, the derivation has been independent of dimensionality, and the order of the approximation (in the Taylor series sense) has not been made explicit; the general ideas apply to all possibilities. We have seen that the end result of the mathematics is an algebraic system $\mathsf{K}\mathbf{a} = \mathbf{F}$, and we have the form of the integrals that define K_{ij} and F_j. It now remains to determine the integrands to construct the linear system. To allow this, the domain's dimensionality, an element family, and a particular element order must be selected.

We wish to solve, at a minimum, two dimensional problems if we are to tackle microelectrode systems. Since mesh construction (discussed in the next chapter) is more difficult in three dimensions, we opt for two dimensions here, and ultimately for the problems that we solve later on. Two basic shapes of elements are normally used in two dimensions: triangles and quadrilaterals, the latter sometimes in the specialised form of rectangles. Since most two dimensional automatic mesh generation has been conducted using triangles, we adopt them exclusively. In particular we shall use for the purposes of explanation the linear variety, which is the simplest. With this type of element the overall solution field, $\bar{u}$, will be piecewise linear. Later, we actually employ quadratic elements for most of our simulations.

If the shape functions are linear in x and y, within each element we have

$$\bar{u}^e = \alpha_1 + \alpha_2 x + \alpha_3 y \; . \tag{2.26}$$

This interpolation within an element may be viewed as a plane intersecting the values of $\bar{u}$ at the triangle's vertices—a_i, a_j and a_k (the subscripts in this section are local to the element). Figure 2.1 shows this idea—note that the vertex subscripts are in clockwise order, in contrast to some texts. Since

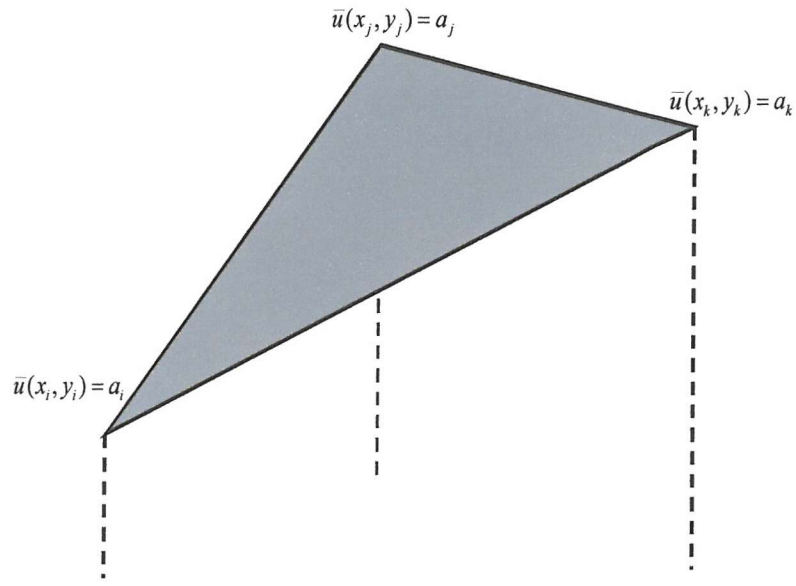

Figure 2.1: A linear triangular element. The nodal values (a_i, a_j and a_k) are interpolated linearly.

the vertex field values are the quantities of interest, we must eliminate the arbitrary α_m constants. This problem boils down to a simple set of three equations:

$$a_i = \alpha_1 + \alpha_2 x_i + \alpha_3 y_i \tag{2.27}$$

$$a_j = \alpha_1 + \alpha_2 x_j + \alpha_3 y_j \tag{2.28}$$

$$a_k = \alpha_1 + \alpha_2 x_k + \alpha_3 y_k \ . \tag{2.29}$$

These can be solved to yield the formula

$$\bar{u}^e = \frac{1}{2\Delta}[(r_i + s_i x + t_i y)a_i + (r_j + s_j x + t_j y)a_j + (r_k + s_k x + t_k y)a_k] \tag{2.30}$$

where

$$r_i = x_k y_j - x_j y_k \tag{2.31}$$

$$s_i = y_k - y_j \tag{2.32}$$

$$t_i = x_j - x_k \ , \tag{2.33}$$

cyclic permutation in the order k, j, i providing the other coefficients, and Δ being the area of the element.

If the shape functions within an element are written as

$$\mathbf{N}^e = \{N_i \; N_j \; N_k\} \tag{2.34}$$

then from (2.30),

$$N_i = \frac{r_i + s_i x + t_i y}{2\Delta}, \text{ etc.} \tag{2.35}$$

Similarly writing the three vertex values of $\bar{u}$ within the element as

$$\mathbf{a}^e = \left\{ \begin{array}{c} a_i \\ a_j \\ a_k \end{array} \right\}, \tag{2.36}$$

allows us to write (2.30) as

$$\bar{u}^e = \mathbf{N}^e \mathbf{a}^e \,, \tag{2.37}$$

in an element-specific imitation of (2.14).

Knowing the functional form of $\mathbf{N}$ allows us to, at least in principle, evaluate the integrals in (2.24) and (2.25). In the particular case where the coefficients a, q, $\mathbf{v}$ and f are constant, exact formulae exist for the integrals, but generally some numerical approximation to the integration must be made (see below).

The length of equations (2.30) to (2.33) might give cause to wonder if a more elegant procedure is available to deal with the relation of element shape functions and their coefficients to real world coordinates. There is, and it involves an element-specific coordinate system, called *area* or *triangular* or *barycentric coordinates*. Since they can simplify and generalise the approach above to higher order elements, and because exact element integration formulae are given in terms of them, we describe area coordinates next.

2.1.4 Area Coordinates

The aim is to find a coordinate system for use within triangular elements, independent of where the element is lodged in Cartesian space. Since we are dealing with triangles, it seems intuitively appealing to have three coordinates, each associated with a different vertex. The difficulty with this idea is simply that a triangle is two dimensional, so there can only be two linearly independent coordinates for a point within it. This is solved by using three coordinates, but making them linearly dependent, reducing the number of degrees of freedom to two.

Consider the system

$$l_1 x_1 + l_2 x_2 + l_3 x_3 = x \ , \tag{2.38}$$

$$l_1 y_1 + l_2 y_2 + l_3 y_3 = y \ , \tag{2.39}$$

$$l_1 + l_2 + l_3 = 1 \ . \tag{2.40}$$

From (2.40), it is clear that having chosen any two of l_1, l_2 and l_3 one cannot choose the third; there are only two degrees of freedom.

Solving for l_i results in the familiar expressions

$$l_1 = \frac{r_1 + s_1 x + t_1 y}{2\Delta} \tag{2.41}$$

$$l_2 = \frac{r_2 + s_2 x + t_2 y}{2\Delta} \tag{2.42}$$

$$l_3 = \frac{r_3 + s_3 x + t_3 y}{2\Delta} \ , \tag{2.43}$$

where r_i, etc. are defined as above. This similarity is not a coincidence: in fact the linear triangular shape expressions are equal to the area coordinates for their corresponding nodes. Something has been achieved, however, since the coordinates (l_1, l_2, l_3) can be used for formulating higher order (e.g. quadratic or cubic) and curved elements. Further, we can use general theory to transform from element coordinate space to Cartesian space, avoiding tedious element-specific algebra. Although we will not use this fact, everything in this section generalises to three dimensions and tetrahedral elements.

As we have already said, we shall use quadratic elements for the bulk of our simulations, partly for reasons of greater accuracy, but also because of error adaptivity considerations explained later in this chapter. While linear elements almost always take the form described here, quadratic triangular elements come in several forms. A fairly minor technical difference, between hierarchical and non-hierarchical elements, is described by Zienkiewicz and Taylor [16] (we use non-hierarchical elements, but see Chapter 6 for mention of their possible advantages). More importantly, we use so-called *subparametric* elements. The more common *isoparametric* variety uses quadratic functions for the *shape* of the element, making it potentially curved. Since curved elements were not felt to be warranted, their additional complications were avoided by the use of simpler subparametric elements. By extension superparametric elements also exist. The details can be found in a standard reference (*ibid.*).

2.1.5 The Element Stiffness Matrix

Having talked in terms of nodal shape functions in (2.13) it might seem natural to construct the linear system (2.23) node by node, filling in K systematically, a row/column at a time. If we attempted to generate the entries for node i, however, we would have to find all nodes connected to node i; and similarly with each node we considered. Depending on the structure of the mesh, and the mesh data structures, this could be very inefficient. Considering a diagram of the nodal shape function for linear triangular elements (shown in Figure 2.2), its piecewise linear nature is apparent, and it is equally reasonable to work on an element basis. This proves more efficient.

The domain integrals in the definition of K_{ij}, whose integrands are non-zero only over the neighbourhood of node i, can be broken into m pieces, where m is the number of elements sharing node i. Thus, if we iterated over these m elements, adding the contribution from each to K (initially zeroed), we would arrive at the same result. The Robin boundary integral

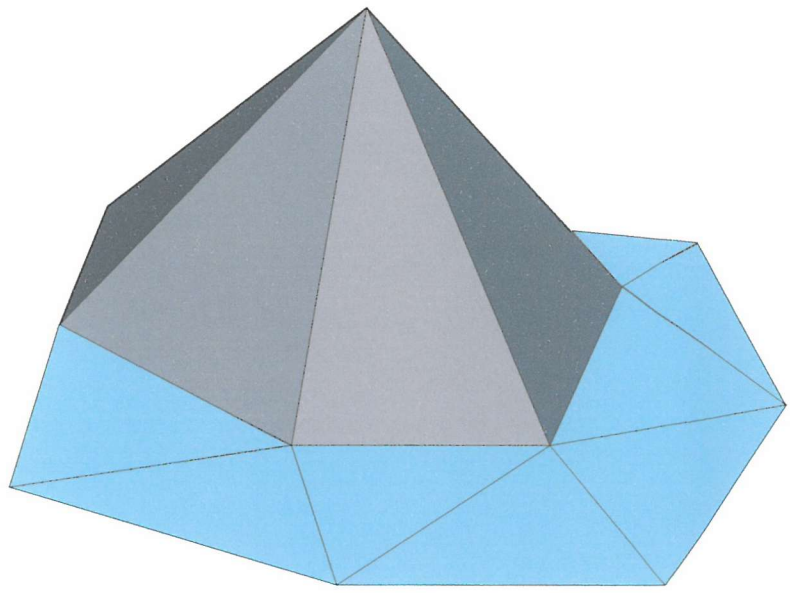

Figure 2.2: A linear triangular element nodal shape function, with some surrounding elements. Note that the shape function is only C^0 at element edges: it is piecewise linear.

can be incorporated on an element-oriented basis too, since each segment of it affects only one element (assuming Robin boundaries are external, which they are in this work). A similar logic applies to the force vector $\mathbf{F}$. It is common to assemble the various integral contributions of an element into a sub-matrix, termed the *element stiffness matrix*, denoted here by K^e; an identical approach also brings us the *element force vector*, $\mathbf{F}^e$.

With linear triangular elements there are three nodes per element, so K^e will be a 3×3 matrix, and $\mathbf{F}^e$ will be a three dimensional vector. The first domain integral in (2.24) is, if a is approximated as constant over the element,[4] given by

$$I_1 = \int_{\Omega^e} a \nabla N_i \cdot \nabla N_j d\Omega \simeq a \int_{\Omega^e} \frac{\partial N_i}{\partial x}\frac{\partial N_j}{\partial x} + \frac{\partial N_i}{\partial y}\frac{\partial N_j}{\partial y} d\Omega \,, \tag{2.44}$$

where Ω^e is the portion of the domain Ω coincident with the element e. The

[4]The implications of this approximation are discussed later in this section.

derivatives are available from (2.35), and are seen to be constant, so

$$I_1 \simeq \frac{a(s_i s_j + t_i t_j)}{4\Delta^2} \int_{\Omega^e} d\Omega = \frac{a(s_i s_j + t_i t_j)}{4\Delta} . \tag{2.45}$$

The next domain integral in (2.24) can be simplified similarly:

$$I_2 = \int_{\Omega^e} N_i \mathbf{v} \cdot \nabla N_j d\Omega \simeq \mathbf{v} \cdot \nabla N_j \int_{\Omega^e} N_i d\Omega . \tag{2.46}$$

The integrand is not constant, but we can use a well-known formula [16, 32, 65, 68] to evaluate it. It can be shown that

$$\int_{\Omega^e} l_1^\alpha l_2^\beta l_3^\gamma d\Omega = \frac{\alpha!\beta!\gamma!}{(\alpha+\beta+\gamma+2)!} 2\Delta , \tag{2.47}$$

where l_1, etc. are the area coordinates introduced above. As noted there, the shape functions for the linear triangular element are equal to the corresponding area coordinates. Therefore

$$I_2 \simeq \frac{v_x s_j + v_y t_j}{3} . \tag{2.48}$$

The other domain integral in (2.24) is approximated as

$$I_3 = \int_{\Omega^e} N_i q N_j d\Omega \simeq q \int_{\Omega^e} N_i N_j d\Omega , \tag{2.49}$$

and by the integration formula is

$$I_3 \simeq \begin{cases} \frac{q\Delta}{12} & i \neq j, \\ \frac{q\Delta}{6} & i = j. \end{cases} \tag{2.50}$$

The same approach gives the domain integral in (2.25):

$$I_4 = \int_{\Omega^e} N_i f d\Omega \simeq \frac{f\Delta}{3} . \tag{2.51}$$

Another formula exists for integrals in area coordinates along the edges of triangles [65]:

$$\int_{\Gamma^e} l_1^\alpha l_2^\beta d\Gamma = \frac{\alpha!\beta!\gamma!}{(\alpha+\beta+1)!} L , \tag{2.52}$$

where Γ^e is the edge over which the integral is taken, and L is its length. The remaining integrals may therefore be approximated as

$$\int_{\Gamma_R^e} aN_i\alpha N_j d\Gamma \simeq a\alpha \int_{\Gamma_R^e} N_i N_j d\Gamma = \begin{cases} \frac{a\alpha L}{6} & i \neq j\ , \\ \frac{a\alpha L}{3} & i = j\ . \end{cases} \tag{2.53}$$

and

$$\int_{\Gamma_R^e} aN_i\beta d\Gamma \simeq a\beta \int_{\Gamma_R e} N_i d\Gamma = \frac{a\beta L}{2}\ . \tag{2.54}$$

(The integral domain Γ_R^e denotes an edge of element e with a Robin boundary—i.e. a subsection of Γ_R belonging to a particular element edge.)

Finally, with a sigh of relief, we write the explicit form of the equations defining the elemental linear system elements (dropping the approximation signs):

$$\mathsf{K}_{ij}^e = \begin{cases} \frac{a(s_i s_j + t_i t_j)}{4\Delta} + \frac{v_x s_j + v_y t_j}{3} - \frac{a\alpha L}{6} - \frac{q\Delta}{12} & i \neq j\ , \\ \frac{a(s_i^2 + t_i^2)}{4\Delta} + \frac{v_x s_i + v_y t_i}{3} - \frac{a\alpha L}{3} - \frac{q\Delta}{6} & i = j\ , \end{cases} \tag{2.55}$$

and

$$\mathbf{F}_j^e = \frac{a\beta L}{2} - \frac{f\Delta}{3}\ . \tag{2.56}$$

Unfortunately, where the magnitude of the convective velocity field, $\mathbf{v}$, is large relative to the diffusion coefficient, a, a different weighting scheme can sometimes be preferable, and these expressions change. This is discussed below in §2.2.

Integral approximation

It is seen later, when convergence is discussed, that finite element with linear elements is second order. It can be shown [16] that, in order to retain convergence, the same order of integration is needed for the element matrix integrals. The approximations made above are of the right order for linear elements. With quadratic elements more accurate integration is required, and this is almost always done numerically, which is actually simpler. In

finite element, Gaussian integration is generally used for this purpose,[5] and is well documented (*ibid.*).

2.1.6 Axi-symmetric Problems

So far no special account has been taken of axi-symmetric problems. It is common to see "axi-symmetric elements" treated separately from Cartesian elements (e.g. [65,66]). New formulae for linear system components can then be derived. But provided one is prepared to sacrifice the seemingly minimal performance increase so gained, both Cartesian and cylindrical coordinate systems (and indeed other ones) can be incorporated into one framework. The Cartesian form of, for the sake of argument, Laplace's equation is

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = 0 \,. \tag{2.57}$$

The cylindrical coordinate form is, following 1.1.2 on page 6,

$$\frac{1}{r}\frac{\partial}{\partial r}\left(r\frac{\partial u}{\partial r}\right) + \frac{\partial^2 u}{\partial z^2} = 0 \tag{2.58}$$

$$\frac{\partial}{\partial r}\left(r\frac{\partial u}{\partial r}\right) + r\frac{\partial^2 u}{\partial z^2} = 0 \tag{2.59}$$

$$\frac{\partial}{\partial r}\left(r\frac{\partial u}{\partial r}\right) + \frac{\partial}{\partial z}\left(r\frac{\partial u}{\partial z}\right) = 0 \,. \tag{2.60}$$

The only essential difference, then, is a weighting within the two derivatives. But we have already provided for a weight function in the form of a in (2.1). Unfortunately it involves a slight abuse of symbology, since the weight we need for cylindrical systems is part of the Laplacian operator in (2.1), not a; but if we redefine a, and work in Cartesian systems with a weighting of x incorporated into a, we get the same result as with special axi-symmetric elements.

Consequently, although we shall continue to visualise axi-symmetric problems with r and z axes, the actual solving process proceeds in, formally, a

[5] Other types of non-polynomial shape functions, for instance in "singularity elements", might require a different approach. We do not use these.

Cartesian system:

$$\frac{\partial}{\partial x}\left(a'x\frac{\partial u}{\partial x}\right)+\frac{\partial}{\partial y}\left(a'x\frac{\partial u}{\partial y}\right)=0\,, \tag{2.61}$$

where a in (2.1) is now a'. Using this approach, special care is needed with additional terms in the equation (q and f would be multiplied by r above). Integrals also need to be adjusted (the practical outcome is illustrated in Appendix E).

2.1.7 Dirichlet Boundary Conditions

The Robin boundary conditions in (2.3) have been incorporated as a fundamental part of the formulation; they contribute terms to the element stiffness matrix and forcing vector. But the solution $\bar{u}$ has yet to be constrained to satisfy the Dirichlet conditions defined in (2.2). This can be done, once the linear system has been assembled, by substituting for all a_j where node j is on Γ_D. The obvious next step would be to reduce the size of the linear system by eliminating row and column j. This would carry the advantage of marginally reducing the linear system solving time and storage requirements. However, it happens that deleting matrix rows and columns is generally—and specifically with the matrix storage format that we use (see § 2.3.4 on page 78)—a slow operation. We therefore adopt a different strategy.

If a node j lies on a Dirichlet boundary, where we know $u = \alpha$, we set all entries in row j of K to zero, except for setting $K_{jj} = 1$. We also set the force vector entry $F_j = \alpha$. This takes care of row j, clearly enforcing the condition $a_j = \alpha$ there, leaving column j for consideration. It should be clear that substituting $a_j = \alpha$ would leave, on each row $i \neq j$ intersecting the column j, a of value αK_{ij}. This, being a constant, needs to be shifted to the right-hand side of the equation—to the force vector—leaving a zero entry in the matrix. The value α is therefore substituted into the system, but the degree of freedom is not explicitly eliminated; we are just guaranteed that the solution of the linear system will give node j the correct value. There

are important computational optimisations that need to be considered when implementing this step, and they are addressed later in § 2.3.4 on page 78.

We exemplify the operation with a 4 node case, where node 3 is on a Dirichlet boundary where $u = 10$; that is, $a_3 = 10$:

$$\begin{pmatrix} K_{11} & K_{12} & K_{13} & K_{14} \\ K_{21} & K_{22} & K_{23} & K_{24} \\ K_{31} & K_{32} & K_{33} & K_{34} \\ K_{41} & K_{42} & K_{43} & K_{44} \end{pmatrix} \begin{pmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \end{pmatrix} = \begin{pmatrix} F_1 \\ F_2 \\ F_3 \\ F_4 \end{pmatrix}$$

$$\downarrow$$

$$\begin{pmatrix} K_{11} & K_{12} & 0 & K_{14} \\ K_{21} & K_{22} & 0 & K_{24} \\ 0 & 0 & 1 & 0 \\ K_{41} & K_{42} & 0 & K_{44} \end{pmatrix} \begin{pmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \end{pmatrix} = \begin{pmatrix} F_1 - 10K_{13} \\ F_2 - 10K_{23} \\ 10 \\ F_4 - 10K_{43} \end{pmatrix} \quad . \tag{2.62}$$

When it comes to computer implementation, difficulties sometimes make other options more attractive. These are also discussed in §2.3.4.

2.1.8 Summary

The progression from the PDE and boundary conditions to the final linear system is summarised in Figure 2.3. If the variational approach had been taken, the initial problem would not have been the PDE, but an *energy functional* that we wished to minimise with respect to all the degrees of freedom available. Doing so would have led directly to the weak formulation. While shorter, the energy functional approach suffers from unfamiliarity and only applying to a subset of problems susceptible to Galerkin treatment. In any case, much of the detail remains the same.

The end product, of course, is a linear system. Numerous standard methods exist to solve such things, detailed in numerical texts [69–72]. A brief sketch of the options is given in Appendix C. We choose, as many others have done, to use the *preconditioned conjugate gradient* (PCG) method for

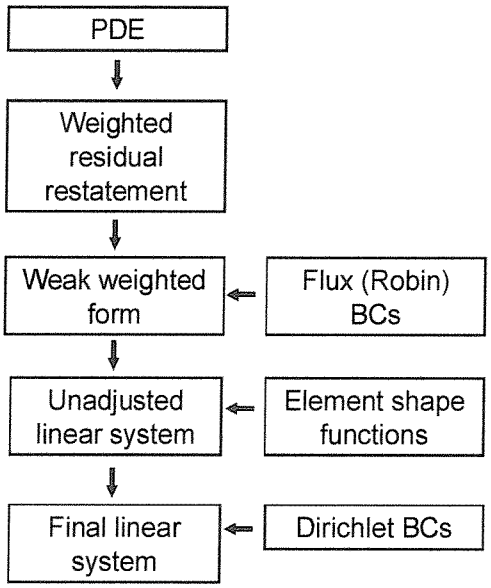

Figure 2.3: The theoretical steps from PDE formulation to linear system approximation.

self-adjoint problems. Where convection is present, no one solver is clearly preferable, so a number were implemented, including biconjugate gradient (BiCG), stabilised biconjugate gradient (BiCGStab) and generalised minimum residual (GMRES), all of which are documented at length by Barrett *et al.* [71] and Saad [72]. These all come under the heading of Krylov space solvers.

From a mathematical point of view, conjugate gradient is complex, and it is only possible to give a short outline of it in Appendix D. Fortunately, computationally it is among the simplest solvers—simpler even than Gaussian elimination—and the only worry is optimisation, which we cover later in 2.3.2 on page 74. The other Krylov space solvers are even more complex, and the reader is referred to the previously cited texts.

Preconditioning is a primary concern among many researchers in this field, and we use a few simple preconditioners, but since our problems do not, partly thanks to efficient adaptive meshing, have that many degrees of freedom, advanced preconditioning techniques were not found to be neces-

sary. More important from our point of view was the efficient error controlled meshing described in the next chapter.

2.2 Petrov-Galerkin Stabilisation

While the Galerkin weighting described above is uncontroversial for purely diffusive problems, this is unfortunately not true where convection is a strong effect. This is not too surprising, as Galerkin weighting gives the "best" answer for self-adjoint problems, in the sense that it is the closest in a certain norm [16], but no such guarantees are available in the considerably more difficult non-self-adjoint case. This has been researched a great deal, as the scalar convection-diffusion equation is commonly used as a prototype for the more complex Navier-Stokes equations arising in fluid mechanics [73]. Unfortunately researchers appear to disagree on the best method of stabilisation, or even if stabilisation techniques are desirable.

Essentially, as Zienkiewicz and Taylor show [74], the various stabilisation methods boil down to using modified weight functions of the form

$$W_i = N_i + \tau \mathbf{v} \cdot \nabla N_i \,, \tag{2.63}$$

where τ is given in terms of another constant, α, that we shall term the *stabilisation parameter*:

$$\tau = \frac{\alpha h}{2|\mathbf{v}|} \tag{2.64}$$

(see below for the definition of h, etc.). These modified weight functions are applied to one or more of the terms in the PDE.[6] As the shape and weight functions are no longer identical, this is generally termed a Petrov-Galerkin weighting scheme. Since the matrix and vector coefficients K_{ij} and F_i described above derive from linear operations on W_i, the effect of this alteration is the addition of various values to the Galerkin versions already derived.

[6] For the reasons mentioned above, the case of nodes on Dirichlet boundaries is ignored.

The aim of the modification is to correct a particular perceived deficiency in Galerkin FEM where convection is the dominant type of mass transport. The degree to which convection predominates is usually measured by the Peclet parameter, which is essentially the ratio of the magnitude of the convective velocity to the diffusion coefficient. Zienkiewicz and Taylor define the *element Peclet number* as

$$Pe = \frac{|\mathbf{v}|h}{2a}\,, \tag{2.65}$$

where $\mathbf{v}$ and a are the familiar coefficients from the PDE, and h is some measure of element size (there is no one definition in the literature; the longest element side is used here, as it is easy to calculate).

2.2.1 The Problem

Zienkiewicz and Taylor [74] illustrate the difficulty that can arise when employing Galerkin weighting to solve diffusive-convective mass transport problems: with some meshes, most notably uniform ones, the approximation becomes steadily worse as Pe increases; instead of a smooth curve passing through the nodes, a wildly oscillating result can arise, bearing no relation to reality (for instance, large negative concentrations can appear). Finite element practitioners agree that this scenario must be avoided, but differ on the means to effect this. Perhaps the most balanced overall view is given by Donea and Huerta [75].

There is no doubt that uniformly spaced meshes are not adequate for use in combination with Galerkin weighting for solving convection dominated problems. Zienkiewicz and Taylor (*ibid.*) show this in detail in the one dimensional case. Harriman *et al.* show the same uniform mesh phenomenon in two dimensions, and on this basis advocate a Petrov-Galerkin stabilisation scheme [37]. The difficulty is that these authors apparently take uniform mesh results to mean that Galerkin weighting would be inadequate with error controlled adaptive meshes. As they give no results to show this, they would appear to be relying on some implied mathematical consensus to justify this

stance. If Gresho and Sani [73, 76] are to be believed, however, no such consensus exists; instead they characterise the debate as "religious".

In fact, Gresho and Sani suggest that one only requires the right mesh to allow Galerkin finite element to operate accurately with strong convection. They term this "Galerkin Finite Element Intelligently Applied" (GFEMIA). For instance, they show that the catastrophic failure of high flow rate Galerkin FEM with a uniform mesh can be rectified using a mesh with only one internal node, provided that node is placed correctly. Further, they suggest that results obtained using the various stabilisation schemes advocated by some are dubious, as they effectively entail solving a different, easier, problem, which may give physically plausible results, but which is nonetheless not accurate. Thus things would not seem to be as clear-cut as Zienkiewicz and Taylor (and Harriman *et al.*) suggest.

On top of this fundamental disagreement, there are a number of different stabilisation approaches documented [74] (e.g. Streamline Upwind Petrov Galerkin, Galerkin Least Squares, Finite Increment Calculus, balancing diffusion). While similar, they can result in different values for τ in (2.63), and in some cases entail applying the weight functions to different terms in the PDE. Zienkiewicz and Taylor advocate a particular functional form of the stabilisation parameter (see below), on the grounds that it is optimal for one dimensional problems, but this does not seem to be widely accepted. Harriman *et al.*, for instance, do not use it (in fact they give little detail on the issue, with no clue as to where they switch from no stabilisation to full stabilisation).

Faced with these differing approaches, it is clearly not possible to pretend that any optimal strategy is used here. As has been noted, a particular problem with the literature on the subject is that it pays little attention to adaptive meshing. Thus, while it may be that GFEMIA is the best approach, it is not clear at all how an adaptive algorithm could be devised to utilise it while conforming to the other constraints mentioned in the next chapter. On the other hand, it might be that the well-documented Galerkin

FEM problems with uniform meshes do not arise where adaptivity works to refine coarse mesh areas afflicted with unphysical oscillations. Our approach is therefore guided by the particular requirements of our problem set, and essentially derives from empirical findings arising therefrom.

All this being said, on one particular subject Zienkiewicz and Taylor and Gresho and Sani appear to agree: the use of "balancing diffusion", where modified weight functions are applied solely to the convective terms of the equation, is discredited. This technique is the analogue of the "upwind differencing" used in finite difference for the same reasons. Leonard [77], shows that it yields the wrong results where source terms are present in the PDE, as they might well be in electrochemical simulations (the title of this paper gives a clue to the varying views of stabilisation schemes). As a consequence, where stabilisation schemes are used in this work, they are applied to all terms. However, the last point would appear to bear on the work of Alden [31], who used upwind finite difference for some convective simulations and yet obtained correct results, so there remains some uncertainty.

2.2.2 Our Approach

Testing with various schemes revealed two essential aspects to the particular problems solved with our adaptive methods. Both camps would appear, from the very limited testing conducted, to have been correct to an extent, at least in the context of the electrochemical problems considered in this work.

Firstly, it was found that with low to moderate flow rates adaptivity *did* work to remove oscillations, without any stabilisation scheme. While the initial crude mesh led to the predictable oscillatory effects, mesh refinement, guided by a current error bound, led to reasonable-looking concentration fields. Most importantly, the resulting currents were in accordance with reality. Unfortunately it was not possible to test this over the full velocity field range, for the reason given next.

Secondly, it was found that all Krylov space solvers tested rapidly be-

came unstable or unfeasibly slow as the flow rate increased, typically above a shear rate Peclet number (see § 4.6 on page 190) of around 2. Even using preconditioned GMRES, generally one of the most stable of the Krylov space methods, with a very long recurrence [71, 72] (typically 200 iterations between restarts on a linear system with around 1000 unknowns) failed to solve problems with higher flow rates. Unless a different solver (or possibly a far better Krylov solver preconditioner) were used, Galerkin weighting would not appear to be practical over the full range of flow rates. This conclusion is of course in concordance with the orthodoxy espoused by Harriman *et al.*, but the reasoning is decidedly different. It is congruent, however, with Alden's findings [31] in relation to upwind differencing in finite difference and its effect on matrix diagonal dominance.

Fortunately it was found that Streamline Upwind Petrov Galerkin weighting dramatically alleviated the Krylov space solver problems, allowing simulation over the full range of flow rates. Rather than the slower GMRES solver, stabilised biconjugate gradient (BiCGStab), was found to be adequate; and it solved systems rapidly. Crucially, not only were oscillations not present in the concentration fields simulated, but the currents derived from the stabilised scheme were found to tally with known analytical results. It is not clear, therefore, from where Gresho and Sani's antipathy toward stabilisation schemes derives. Perhaps it is more a result of hostility to "balancing diffusion" and "upwind differencing", which demonstrably fail to yield correct results in known cases.

Whatever the particular stances of finite element authors, the conclusion would appear to be fairly clear. Since stabilised schemes do appear to yield correct results, and because beyond moderate flow rates they are necessary to allow our unsymmetric Krylov space solvers to function efficiently, they are used in this work for simulations involving convection. In order to provide a consistent approach over the full range of flow rates, the "optimal" stabilisation parameter expression given by Zienkiewicz and Taylor [74] was used. This adjusts the degree of stabilisation from zero, where there is no

convection, to full at fully convective transport, according to the expression

$$\alpha = \coth Pe - \frac{1}{Pe} \,. \tag{2.66}$$

For the channel flow model problem this was found to give correct results, as well as reasonably fast solutions, at all tested flow rates. That said, during an extended discussion, Donea and Huerta [75] suggest that the formula should be modified depending on element order and node type, so it may be that improvements could be made.

2.2.3 Element Stiffness Matrix Calculations

For linear elements Petrov-Galerkin weight functions do not involve any particular difficulties. No change is made to the diffusive term, as there the weight function is differentiated, reducing the new part to zero. With higher order elements a problem arises, as the weight functions are not continuous between elements. Since in the diffusive term they are differentiated, this leads to integration of a singularity at element edges. From our global weight function perspective, two basic approaches appear to exist to combat this: either modified forms of (2.63) are used where the derivative is multiplied by some sort of "bubble" function that equals zero at the element boundary [74], or some effort is made to incorporate the singular part of the integral in a "distributional sense" [78].

Harriman *et al.* eschew both courses because they operate on an element-by-element basis, saying that after breaking the domain integral into element-sized pieces the offending derivative term "makes sense" [37]. The results achieved thereby are encouraging, and so with admittedly little theoretical justification we follow their approach. Ideally a more detailed analysis would be conducted.

2.3 Computational Aspects

The computational aspects of the implementation are quite different in their emphases than the mathematical ones. For instance, where the linear system assembly occupies pages of algebraic manipulation, once the formulae for stiffness matrix and force vector elements are available, the process is achieved in a few hundred lines of code. As might be expected, the mesh generation and refinement of Chapter 3 required relatively involved programming, but ultimately occupied approximately a thousand lines of code.

Instead, so far unmentioned areas took considerable implementation effort. Rather than a command line interface, taking as input a problem description, and outputting the numbers describing the solution, it was thought preferable to have a graphical interface, with solving an interactive process, and the resulting concentration field plotted instantaneously. Thus a graphical user interface was constructed, with code drawing the mesh along with the various solution and error fields. Ultimately it would be preferable to have problems described by interactively drawing the domain boundary, but currently the program still loads a text-based problem description. However, the solution process is nonetheless faster for the user, and more convenient, with less likelihood of catastrophic failure going undetected, since the result is instantly plotted.

Another important aspect of the interface is less obvious, but has been touched upon already in § 2.1.6 on page 61. The program solves a set of equations of the type

$$\frac{\partial}{\partial x}\left(a(x,y)\frac{\partial u}{\partial x}\right)+\frac{\partial}{\partial y}\left(a(x,y)\frac{\partial u}{\partial y}\right)-v_x\frac{\partial u}{\partial x}-v_y\frac{\partial u}{\partial y}+q(x,y)u=f(x,y)\ , \tag{2.67}$$

with an arbitrary (in principle) mixture of Robin and Dirichlet boundary conditions. In each governing PDE, then, we have five arbitrary functions.[7]

[7] Actually, as might be discerned from Appendix E, the program can accept different diffusion coefficients in the x and y directions, but this capability is not used here.

And it might also be desirable to prescribe functions of spatial coordinates as boundary conditions, for instance, where an analytical concentration is imposed for testing purposes. Typically simulation programs take as input only numeric constants, because these are readily handled by computer languages' in-built capabilities: they are a fundamental data type. But limiting a, v_x, v_y, etc. to constants curtails the program's range of applicability. We have already seen that axi-symmetric problems can be incorporated without special elements simply by—assuming a constant diffusion coefficient for a simple example—setting $a(x, y) = x$. Thus, allowing functions to be input, and evaluating them as needed, significantly contributes to the generality and versatility of the program, and reduces the code required for special cases.

The three primary challenges of the programming were essentially, ignoring the bulky but relatively simple graphical interface code, meshing, efficient sparse matrix solving, and parsing and evaluation of functions. The first of these is not well documented, and deserves a relatively detailed explanation, which is given in the next chapter. The second is a well researched problem; once a sparse matrix format and an iterative algorithm have been selected it is straightforward. We therefore describe only the matrix format and the matrix-vector product routines. The last is a challenge met in every computer language compiler, but is certainly not described in any finite element text. Because it is not central to our purpose we discuss it only briefly, although it took more programming than the meshing. Finally, the mesh and matrix data structures used had certain consequences for efficient implementation, of which we give an outline.

2.3.1 Programming Platform

Everything so far described was implemented in C++ [79]. The only viable competitor in terms of speed and suitably for scientific computing would be Fortran in one of its later incarnations, but there are a host of reasons to prefer C++. Firstly, almost all APIs (application programming interfaces)

are geared towards C [80] and C++ (C is almost exactly a subset of C++). Since graphical displays and windowing interfaces depend on API availability, this severely limits Fortran's usability. Secondly, Fortran compilers are expensive and relatively unavailable.

Finally, Fortran does not have the same linguistic functionality of C++. It is wrong to characterise C++ as purely a mixture of *structured* C with new *object oriented* functionality; it incorporates an important third programming paradigm with its *templates*, a form of *generic programming*. Object oriented programming has a number of advantages—data encapsulation, inheritance, etc.—and it was used extensively, but generic programming also had an important rôle. Not only does generic programming allow, for instance, automated code generation for both single and double precision versions of the same routines; it also makes possible the extensive C++ Standard Library [81]—in particular the Standard Template Library—used throughout our finite element implementation.

For these reasons, Fortran (even with its newly acquired object orientation) and the fashionable Java are generally less capable, in spite of potentially allowing more elegant expression of simpler programs. (In fact Java would have to be discarded on performance grounds in any case, on account of issues with run-time compilation and "garbage collection".) One qualifier should be attached to the previous statement: Fortran 90 and later do have in-built constructs for parallel programming, and might prove more convenient for leveraging parallel systems. But no parallel systems were used for the finite element program because it was fast enough on serial machines, and if they were C++ could still use their facilities; it would just not be with native language capabilities.

The particular programming environment chosen was Microsoft Visual C++, principally because it best supports Windows programming; but it also happened to be competitive performance-wise: in tests it consistently produced faster code than a popular alternative, GCC. The two latest versions at the time of writing, 6 and 7, were both used, with little difference

between them for our purposes. The simulations were run on a variety of standard Intel® 80x86-compatible processor platforms, primarily a mobile Pentium III® running at 850MHz. Most of the adaptive finite element simulations took a few seconds, so no outstanding processing speed was required. The memory requirements were also modest, typically being around 5 Mb, much of which was consumed by the graphical interface.

For double precision arithmetic, as was used throughout this work, the Pentium III does not offer any vector processing capabilities. It is therefore probable that the machine instructions generated by the compiler are not easily improvable with hand-tweaking, given the complexities of Pentium III instruction reordering, etc [82, 83]. The possibilities of other platforms, and of single precision, are discussed in Chapter 6.

2.3.2 Krylov Space Solvers

It is assumed in the analysis of Appendix D that the multiplication of a vector by the system matrix is an $\mathcal{O}(n)$ operation, where n is the number of nodes, and consequently the number of rows and columns in K. Further, using a simple two dimensional array matrix format, one quickly discovers that problems will not fit in memory if the number of nodes is more than a few thousand, so it would be preferable for the memory footprint to scale with n, not n^2. These aims are only attainable if a sparse matrix format is used that ignores zero entries.

Sparse matrix formats abound [69, 72], but for unstructured sparsity patterns there are only a few viable options. Because we are using iterative solvers we do not need to randomly access matrix elements; we only need to access them in rows at a time. This leads us, fairly inevitably, to the *compressed row storage* (CRS) format (*ibid.*). Here for each row an array of non-zero entries is stored, along with an array of their respective column indices. We expect, given the limited connectivity between nodes, just a handful of entries per row. Of course, where we deal with self-adjoint prob-

lems the matrix is symmetric, so we only store the upper triangle, the lower being implied. As a final space optimisation, because diagonal entries are always non-zero, they are stored separately as an n dimensional vector. This also marginally simplifies, and hence speeds up, the inner loop for matrix multiplication and related operations.

The matrix-vector multiplication function is simple enough to give in Figure 2.4 on the next page. In fact, because we implement two basic preconditioners, the various tricks used to speed them require a number of related functions; but the essential simplicity remains. All the other computational elements of the Krylov space solvers are operations with dense vectors, and are hence straightforward in their implementation.

2.3.3 Function Parsing and Evaluation

The theory of mathematical function parsing and evaluation is a subset of the theory of compiler writing. It is therefore covered in many books (e.g. [84]), and no attempt is made here to explain it. We simply note the particularities of our problem, and the techniques used to address them.

A simple *lexical analyser* (*ibid.*) was used to convert raw characters—digits of numbers, addition signs, etc.—into *tokens*—numeric constants, arithmetic operators, etc. The second phase, parsing, was more involved. Adopting the standard classification system of Chomsky [85, 86], mathematical formulae conform to a simple *Type 2* or *context-free grammar*,[8] and as such can be parsed by standard techniques. However, normal mathematical notation, also called *infix* notation, is harder to parse, and slower to evaluate, than *postfix* notation. A particularly intuitive means of parsing expressions, called *recursive descent*, is in fact not applicable, because some fundamental arithmetic operators are not left-associative, which it requires. Thus, iterative loops were added, alongside the recursive parsing, to handle these.

[8] In this technical sense *grammar* means, roughly, the rules that govern valid constructions from component symbols.

```
void CSSymMat::MulVec(const CVec& vec, CVec& ans) const
{
    ans.Zero();
    vector<CMatRow>::const_iterator row = m_rows.begin();

    for (int row_i = 0; row != m_rows.end(); ++row, ++row_i) {
        // do diagonal
        ans[row_i] += m_diag[row_i] * vec[row_i];

        vector<CMatEntry>::const_iterator entry =
                m_rows[row_i].ConstBegin();
        for (; entry != m_rows[row_i].ConstEnd(); ++entry) {
            int col_i = entry->m_n;

            // do upper and lower triangles
            ans[row_i] += entry->m_val * vec[col_i];
            ans[col_i] += entry->m_val * vec[row_i];
        }
    }
}
```

Figure 2.4: The C++ function for the symmetric CRS matrix-vector multiplication function. Each row is dealt with both as a row and a column on account of the symmetry. The results are accumulated in the ans vector structure, addressed with array syntax. The diagonal is handled separately. This code fragment exemplifies both object oriented programming, and use of the STL, which subsists on generic programming.

The end product of parsing was a postfix version of the expression, which had the minor advantage of eliminating parentheses. Since stiffness matrix assembly requires numerous function evaluations, one optimisation, termed *constant folding* (*ibid.*) was implemented. This simply means that constant expressions—e.g. $\pi/2$—are evaluated once during parsing, and stored as constants to avoid processing overhead every time the expression containing them is evaluated.

As well as the four elementary arithmetic operators and parentheses, powers, elementary functions (e.g. $\sin x$), and some special functions (for example Bessel functions) are usable in the input expressions. To allow more concise and readable input files, expressions can also refer to one another.

The effect of all this was to allow arbitrary functions where some programs only allow constants: the governing PDE, the boundary conditions, and even the boundary vertex coordinates are all—where applicable—functions of spatial coordinates and other user-definable expressions. But perhaps the greatest advantage is that the error norm used to guide adaptive refinement in Chapter 3 is a user-defined expression. Thus any applicable mechanism can be solved with any suitable error criterion, all with the same program. This seems to justify the work put into function parsing and evaluation; it exposes the full generality of the error estimation code to the user.

The only concern might be performance, but here we are relatively safe. Since the linear system solving scales worse than linearly with n, it will always eventually dominate $\mathcal{O}(n)$ steps like stiffness matrix assembly and error estimation. The crossover point depends on the proportionality constants—asymptotic performance is after all not the only measure of algorithmic efficiency—but in practice it was found that for most real problems the parsing code had a tolerably small impact on solving speed. Possible optimisations are discussed in Chapter 6.

2.3.4 Linear System Assembly

It is straightforward to assemble the element stiffness matrix of § 2.1.5 on page 57 using simple data structures. The same applies with Robin boundary conditions. But it should be noted that the Dirichlet boundary procedure described in §2.1.7 can be made significantly more efficient by using the full information in the mesh data structures.

The CRS matrix format does not allow efficient random access to row elements, so picking out each matrix entry on column j would be inefficient (this would need to be done for each node on the Dirichlet boundary). Further, given the sparsity of K, it seems inefficient to seek to zero each element in an entire row and column when most elements therein would be zero already.

The neighbour information stored in our more comprehensive mesh data structures (documented in the next chapter) helps here. Using the edge pair linking it is possible to quickly determine the nodes sharing edges with the Dirichlet boundary node. Since these correspond to the only non-zero entries on the relevant row and column (because elsewhere the two shape functions have no overlap), only these need be zeroed. Instead of an $\mathcal{O}(n)$ operation per Dirichlet node, then, a roughly constant time operation was possible instead. This effectively eliminated Dirichlet boundary conditions from performance considerations.

One other possibility for Dirichlet adjustment has already been mentioned: elimination of the degree of freedom. It is not easy to envisage an efficient implementation of this with the CRS format. The offending row could be eliminated easily enough; and so could the offending non-zero column elements, using the neighbourhood information mentioned above; but row entries after column j would need to be re-numbered, giving again an $\mathcal{O}(n)$ algorithm per Dirichlet node.

Another, inexact, method of incorporating Dirichlet conditions is given by Zienkiewicz and Taylor [16]. Termed the "penalty method", it retains the degree of freedom, but avoids manipulating any row apart from j by using

a large parameter γ to give row j overriding precedence, meaning one of the linear system's equations becomes $\gamma u_j = \gamma\alpha$. Perhaps without the mesh data available to us it is an attractive option, but we see no advantage over our approach, with only potential additional inaccuracy arising.

2.4 Summary

In this chapter we have presented the fundamentals of the finite element method, a viable scheme for solving a wide range of electrochemical mass transport problems. We have, it is true, ignored transient and non-linear problems, but no fundamental barriers are apparent in the way of their implementation; in principle the method we have described is fully general. Instead we choose to focus on solving, in a relatively complete way, first order steady state problems. The rest we leave to the future, and the final chapter.

In order to apply this discretisation scheme, a mesh is required, and if it is to be efficient, one tailored to whichever electrochemical problem is under study. In order to achieve this for the generality of problems, adaptive mesh generation is required. This is described in the next chapter. Where the current is of interest, it is only natural to use the error in this to guide adaptivity, and so this too is covered next.

Chapter 3

Adaptive Finite Element

After examining the generic machinery for finite element solving *per se*, in this chapter we present an adaptive mesh optimisation algorithm capable of efficiently modelling the boundary singularities (see § 1.1.5 on page 18) in microelectrode geometries. Eschewing the *a priori* meshing approach of many practitioners (e.g. [87]), we instead look to adaptive mesh control in general [16], and error control of the current particularly—a notion introduced to electrochemistry by Harriman *et al.* [33–38]. Harriman *et al.*'s results are encouraging, and an important step forward for electrochemical simulation, as they tie mesh adaptivity to the quantity of interest. But their formulations were apparently problem-specific, and the meshing would appear to be inflexible, so we modify and extend their work significantly.

The general idea of mesh refinement is to use a series of successively improved meshes, each solved with the standard FE techniques of the previous chapter, until an answer of the necessary accuracy is reached. The flow diagram for this approach is shown in Figure 3.1. In the *adaptive* case, only specific areas of the mesh are refined in each step; elements contributing most to the error are identified, and replaced with more elements, it is hoped more accurately modelling the solution. This is important, as refining the entire mesh is generally inefficient because it generates too many nodes.

Apart from structured meshing of regular regions, there has been little

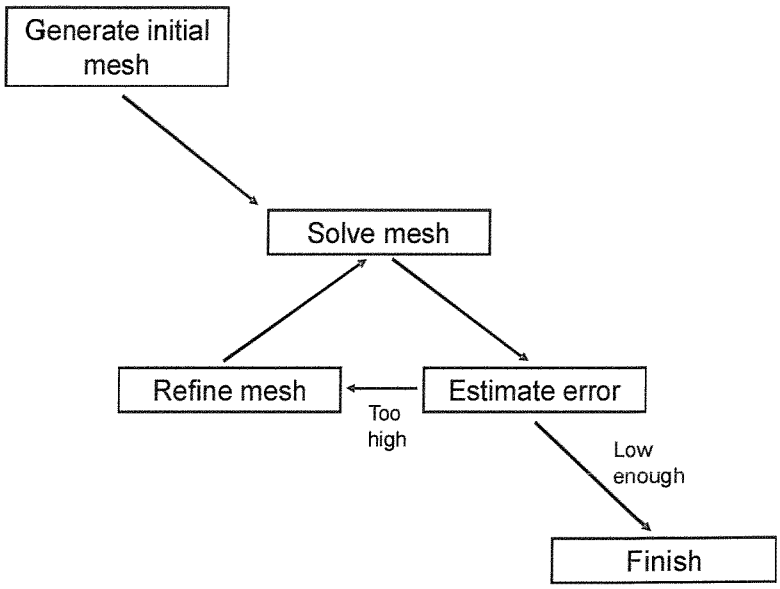

Figure 3.1: The algorithmic flow for finite element with successive refinement. Note that the error of the initial mesh is in principle irrelevant. The entirety of the Galerkin formulation and the matrix solving comes under the "solve mesh" step, which is by far the most involved.

use of automatic meshing within an electrochemical finite element context. Consequently much of this chapter is novel. In deciding on a mesh refinement algorithm, one is generally constrained (unless some non-standard finite element scheme is being employed) by the need for certain mesh attributes. These are described here, before an algorithm is outlined which can generate meshes in general two dimensional simply connected regions, and refine them as needed.

After establishing a means of refining the finite element mesh to increase accuracy, we turn our attention to measures of error, and present new theory, simpler and more flexible than that used by Harriman *et al.* (*ibid.*), for the accurate computation of currents (and in fact other quantities related to the concentration field).

3.1 Mesh Generation and Refinement

All the theory in §2.1 is applicable to any valid mesh, such as the one shown in Figure 1.4 on page 28. The validity or otherwise of meshes is considered in the next section. This bears on the question of how to first generate a mesh (the first step in Figure 3.1), and then how to refine it in order to increase accuracy—a question with more than one answer. We present our approach, a form of Delaunay triangulation, in §3.1.2.

Unfortunately, unlike the basic FE formulation, automatic mesh generation is not covered well in standard texts, and is an active area of research. Generally it is assumed that a mesh is available, possibly having been manually generated. It is difficult, therefore, to have a definitive view on the best approach, if indeed there is one. We adopt what seems a logical strategy, but it is certainly not the only one. Doubtless as automatic mesh generation becomes *de rigueur*, as surely seems inevitable, the situation will improve.

The idea of mesh refinement is simple. As the element size tends towards zero we might expect, as with decreasing finite difference grid spacing, to achieve convergence to the exact solution. If we wish to simulate to a pre-ordained accuracy then an obvious strategy is to subdivide a crude initial mesh until the solution is accurate enough. This is called h mesh refinement. The alternative—increasing the order of elements, which in turn means effectively raising the order of the Taylor series approximation—is called p mesh refinement.[1] Theorems exist [16, 32] that prove convergence, under certain conditions, with both decreasing element size and increasing element order, so one might reasonably wonder which is the superior approach.

Mathematically speaking, p refinement is the more attractive technique

[1] Yet another type of refinement, r refinement, exists. This moves node positions while keeping the number of degrees of freedom the same. It is described by Zienkiewicz and Taylor [16] as being "theoretically of interest" but having "little to recommend it". This is probably true as regards general engineering practice, as well as in the context of our problems.

because, under certain common assumptions,[2] with increasing degrees of freedom it can converge at an exponential rate. However, it has been pursued less commonly than h refinement, principally because it is difficult to implement. It clearly requires a practically unlimited degree of element available in the basic Galerkin (or whichever FE formulation) solver, which while perhaps not as difficult as the derivation in §2.1.3 might suggest (there exists a more systematic theory of elements) is nonetheless a challenge. Secondly, if the refinement is adaptive, as it is here, where different domain areas are refined more than others, elements of different orders (with different numbers of nodes) would abut, raising further difficulties, not least in the data structures used to store them. Some researchers have combined h and p refinement into hp refinement, but again this has proved problematic to implement. Since the majority of research exists on h refinement, and the programming challenges are manifestly less daunting, we follow this trend, noting that more efficient alternatives may well exist, some mention of which is made in Chapter 6.

There are two basic approaches within h mesh refinement: *element subdivision* (also called *mesh enrichment*) and *mesh regeneration*. In both an initially crude mesh is refined according to an error criterion until the desired accuracy is reached. The difference is simply that in the former new nodes are added to the existing mesh, and possibly old ones removed ("derefinement"); whereas in the latter an entirely new mesh is generated at each step. It is not obvious which is better. Derefinement can be difficult to implement, but then it is not always necessary (indeed, we do not implement it). And mesh regeneration, given a fresh start each iteration, might be expected to produce closer to optimal meshes. On the other hand, generating a completely new mesh can be time consuming. Element subdivision seems more intuitive, and is more commonly employed, and for these reasons we adopt it, but doubtless mesh regeneration could prove an adequate, possibly better, replacement.

[2]See *ibid.* Notably, we assume here that singularities are not present, which is not usually true with our type of problems. We address this concern below.

3.1.1 Meshing Aims

Convergence theorems supply information about how the mesh affects solution accuracy. It can be shown [16], for instance, that in the absence of singularities, the order of convergence of a concentration field approximated by linear elements is $\mathcal{O}(h^2)$, where h is a measure of element size, and we assume all elements' sizes tend towards zero uniformly (*uniform refinement*). Another way of stating the order of convergence is $\mathcal{O}(n)^{-1}$, since the number of degrees of freedom (the number of nodes) n is approximately inversely proportional to h^2 in two dimensions.

As noted in §1.1.5, nearly all of the problems under study possess boundary singularities; and it happens [16] that the order of convergence for uniform refinement can be, and usually is, much lower with such singularities. However, it can also be shown (*ibid.*) that with a type of *adaptive refinement*, where, broadly speaking, elements near singularities are refined, the rate of convergence above with increasing n can be recovered.

Note that in the following section we shall not discuss the quality of the mesh *vis a vis* its ability to model the phenomena we are simulating; that belongs later, where error analysis is discussed. The point of adaptive mesh refinement is to add elements where needed—and only where needed—to capture important phenomena. This will be defined by the governing equation and the boundary conditions. However, regardless of the particular physical problem under study, there are criteria that all meshes must satisfy, and we sketch them next.[3]

[3]This is not entirely accurate. For instance, long thin "sliver" elements are sometimes encouraged in convective simulations. And "non-conforming" elements breaking continuity requirements are also sometimes used. Nonetheless, what we say *does* apply to the great majority of finite element simulation, and certainly to the electrochemical problems that we study; such unorthodox approaches demand special justification.

Valid meshes

There exists a basic condition for any mesh used, regardless of accuracy. It is this: if an edge is shared by two elements, they must also share the nodes at its ends. This proviso is not surprising when one considers the necessity of inter-element continuity. If a node breaks an edge in one element, but not the edge in the neighbouring one, then the edge field values can disagree, and C^0 continuity is lost. See Figures 3.2 and 3.3 for an illustration of the implications of this necessity. Essentially, so-called *hanging nodes* are not allowable. This has consequences for when we wish to refine the mesh by splitting elements: if an edge is split, both elements sharing it must also be split, not just one.

Mesh quality

The other important constraint one must keep in mind, particularly with adaptive meshing, is that convergence only holds when the lengths of all the edges of the elements being refined tend towards zero at approximately the same rate. This statement can be made mathematically precise—see [32]. An obvious implication is that the element subdivision procedure one must not allow angles to tend towards zero while edge lengths remain finite. This sends the element area to zero, but it destroys the convergence of the finite element method. A simple means of subdividing elements that suggests itself in the light of the complications of splitting edges mentioned above—adding nodes at triangle centroids—is illustrated in Figure 3.4 on page 87, and clearly breaks this criterion. Obviously this technique alone would not be a usable means of refining the mesh, but adding nodes at elemental centroids and then rearranging mesh edges to enforce the Delaunay constraint (see below) *is* a usable strategy, and is the one employed in our adaptive finite element simulations.

An important generalisation of the above rule (*ibid.*) is that "badly shaped" elements give inaccurate results [88]. Elements with small angles

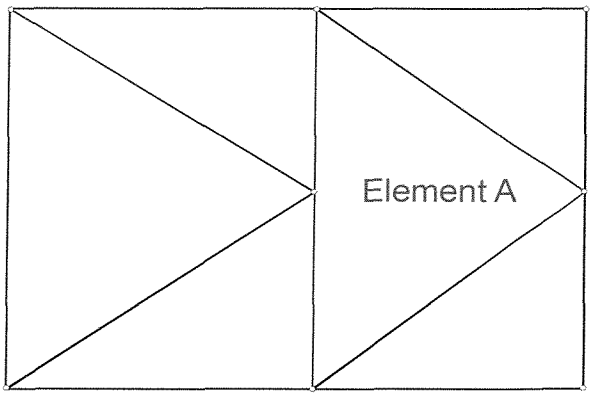

Figure 3.2: Nodes not shared by elements on which they border, as in here, are not allowed in meshes. The solution is simply to subdivide element A, as shown in the figure below.

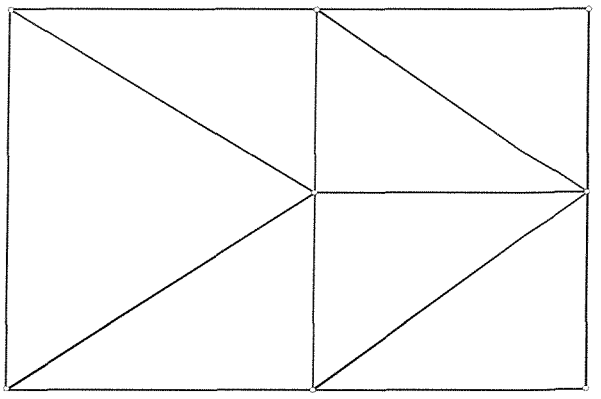

Figure 3.3: Having added an edge, the offending node of the previous figure is now legal.

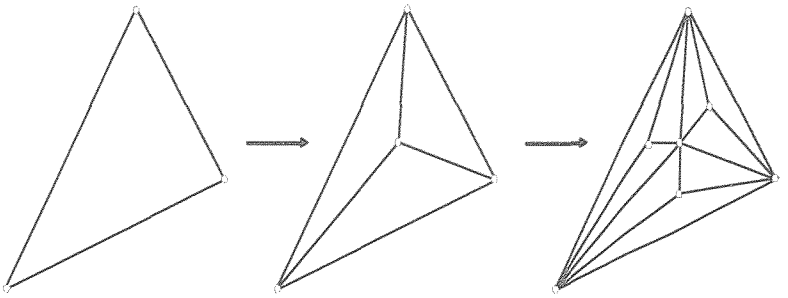

Figure 3.4: A faulty element subdivision strategy. Even though the element areas and some edge lengths tend towards zero, some edges remain untouched. This destroys convergence.

give *ill conditioned* (see [89, 90], and Appendix D for the implications) system matrices [91]. Large angles, on the other hand, can cause a large FE discretisation error [92]. Thus it must be our aim, in constructing any mesh, to ensure triangles are "nicely" shaped, with angles close to $\pi/3$. As already noted, we use *Delaunay triangulations* for this purpose, motivated by their special properties.

3.1.2 Delaunay Triangulation

We take as our starting point for this section a closed two dimensional domain boundary composed of straight line segments. No curved edges are allowed, and no holes may be cut out of the region so described. In mathematical terminology the domain must be *simply connected* [13], and described by a *polygon* as defined by O'Rourke [93].

The absence of curved boundaries considerably simplifies automated meshing, and anyway two dimensional curved domains[4] can be reasonably approx-

[4]Note that we do not here mean cylindrically symmetric three dimensional domains whose curved boundaries are eliminated on the symmetric reduction to two dimensions—we perform simulations in many such domains in Chapters 4 and 5. Instead we mean, for instance, that an SECM tip with curved tapering would not be exactly representable in the input file.

imated in most cases by subdivision into enough straight lines. Most simulation domains will require no such geometric approximation in any case, as in two dimensions they generally have straight edges.

The lack of provision for holes in the domain is a more serious restriction, preventing some domains from being simulated, and with no accessible approximation. It would not, however, be impossible to incorporate at a later time; there is no fundamental barrier to doing this.

For the bare minimum of meshing, for output we wish to produce a *triangulation* [93], which will satisfy the node connectivity requirements of §3.1.1. But mindful of the criteria for a good mesh we shall also insist that it be a sort of Delaunay triangulation—the *constrained Delaunay triangulation* (CDT) in fact—the construction of which boils down to two steps, in our case:

1. Constructing a valid initial triangulation;

2. Flipping edges to enforce the constrained Delaunay triangulation property.

These steps are combined in algorithms that create the CDT immediately from the input data, but our method carries the advantage that any subsequent triangulation not of the constrained Delaunay variety can, by repeating step two, be converted to one that is. Since we plan to refine the initial mesh this will prove to be useful.

We now describe the theory of the two steps in detail, leaving computer implementation until later.

1. Initial triangulation

The precise definition of triangulation is given by O'Rourke [93]. We describe it as the subdivision of the boundary polygon into non-overlapping triangles by connecting its vertices with edges. The nature of the initial triangulation is (at least in principle) not important, provided it is valid. Since with most boundaries possessing more than three vertices there is more than one

possible triangulation, this is helpful. If, before enforcing the CDT constraint, we were desirous of "nicely shaped" elements, as talked of previously, then we would have to pay attention to the angles of triangles produced. Since, as we shall see, this is dealt with by turning it into the constrained Delaunay triangulation, we can pick any algorithm from those available in O'Rourke [93], or any other source.

Figure 3.5 shows two different triangulations of a simple shape. The second one would tend to give less accurate results than the first owing to the high aspect ratio of one of the triangles. However, enforcing the Delaunay property would flip the internal edges to produce "nicer" triangles—see below.

It should be noted that neither the initial triangulation nor the CDT creation can break the node connectivity requirements discussed above as no nodes are introduced during the process, only edges; and in the case of initial triangulation the preexisting nodes reside only on the boundary, where edges will only belong to one triangle (element) anyway.

O'Rourke (*ibid.*) presents a variety of triangulation algorithms, ranging in their speed from $\mathcal{O}(n^4)$ to $\mathcal{O}(n)$, where n is the number of boundary vertices. Since the boundaries of most problems that we tackle generally have of the order of ten vertices (although approximated curved boundaries could have many more), and the initial triangulation process is only performed once per simulation, we have no particular worry about the order of the algorithm used, providing it is reasonable. The $\mathcal{O}(n^4)$ and $\mathcal{O}(n^3)$ and $\mathcal{O}(n^2)$ triangulation algorithms are all relatively simple, whereas those of $\mathcal{O}(n \log n)$ and better are difficult to virtually impossible to implement. Thus O'Rourke's[5] $\mathcal{O}(n^2)$ "ear removal" algorithm (Algorithm 1.1 in O'Rourke [93]) was selected as the best compromise.

The crux of the algorithm is the definition of vertices as being, or not being, "ears"—that is, vertices whose immediate neighbours can be joined

[5]O'Rourke quotes $\mathcal{O}(n^2)$ triangulation algorithms as having been "implicit in proofs since at least 1911".

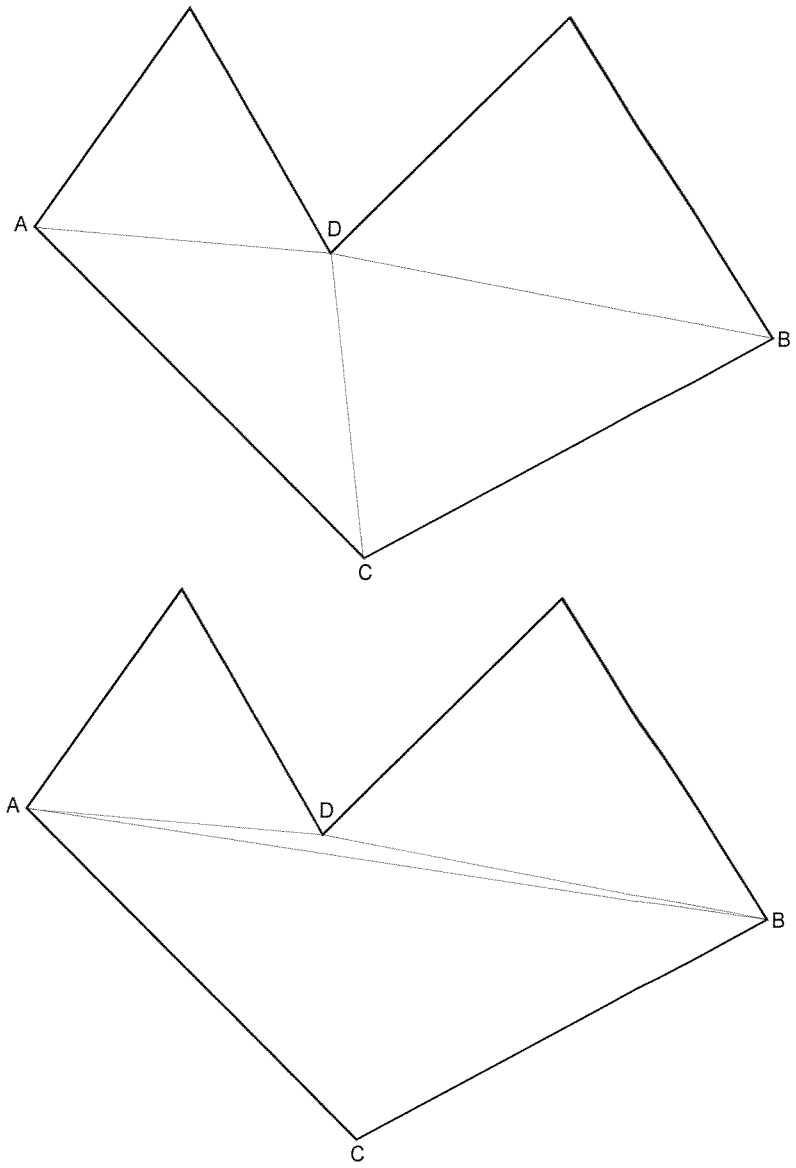

Figure 3.5: Two possible triangulations of the simple shape described by the bold lines—the thinner lines are edges added during the triangulation process. Note that no nodes are added, only edges.

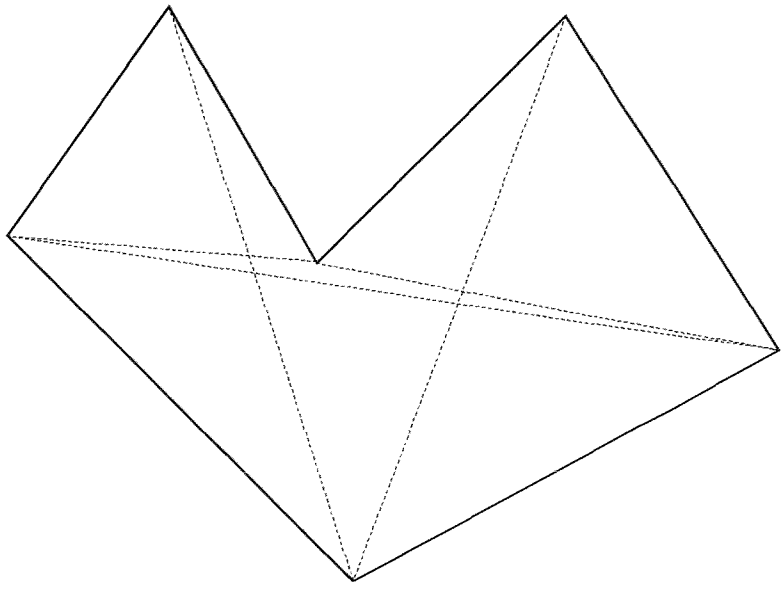

Figure 3.6: The "ear removal" triangulation algorithm illustrated. The unbroken line is the boundary of the region being triangulated. Five of the six vertices are classified as "ears", and the corresponding subdivisions for their removal are shown in dashed lines.

by a line inside the polygon. Any such "ear" can be chopped off, leaving a smaller polygon to be triangulated and a triangle to be added to the stack of triangles making up the triangulation. The concept is shown in Figure 3.6.

By careful maintenance of data structures classifying the status of vertices as "ears" or not, this technique repeatedly applied leads to an $\mathcal{O}(n^2)$ algorithm. This is deemed more than adequate, as there will be later in the process many more element nodes, and the CDT algorithm scales, in the worst case, quadratically with the number of these.

The only complication with the algorithm adumbrated above is the data structure used for the boundary, and for the resulting mesh—briefly, a circularly linked list is used for the boundary; the mesh is more complicated, and will receive more attention presently, in §3.1.5.

2. Enforcing the constrained Delaunay condition

The (unconstrained) Delaunay triangulation is a certain triangulation[6] that has a number of useful properties. The particular property that we exploit is given by Edelsbrunner [94]: over all potential triangulations, the Delaunay triangulation maximises the smallest angle of all the triangles. This ensures that, given a certain nodal placement, the elements arising from its triangulation will keep to a minimum small angles; it maximises the minimum angle. We also wish, for finite element, to minimise the maximum angle; but large and small angles are often concomitants, so the two qualities are closely related. The more classical—and easily testable—property is that no triangle circumcircle (the circle defined by a triangle's vertices [93]) contains a fourth vertex inside its boundary.

We do not use a "proper" Delaunay triangulation, but a constrained one. This is because Delaunay triangulations operate on point lists, whereas we have a boundary with preexisting edges; therefore we must ensure that, whichever edges are added, the edges given as the domain boundary remain. This constrains our ability to meet the Delaunay properties. For instance, a user can supply a boundary polygon with an angle of $\pi/100$ and it cannot be removed without mutilating the domain. Nor can the circumcircle property above generally be ensured. However, Edelsbrunner [94] quotes the *Constrained MaxMin Angle Lemma*, which states that, among all the constrained triangulations, the CDT maximises the minimum angle; in this sense it is optimal for a given boundary.

Since we are starting from a preexisting triangulation, and because we wish to add subsequent vertices during mesh refinement, we use an *incremental* algorithm for Delaunay triangulation. Other approaches exist, for instance Fortune's plane-sweep $\mathcal{O}(n \log n)$ algorithm [95] for Voronoi dia-

[6] In fact, in the case where four vertices are cocircular, the Delaunay triangulation is not unique, but for our purposes this does not matter, as any of the candidates would be equally good.

grams,[7] but we use one of the initial $\mathcal{O}(n^2)$ algorithms—due to Lawson [96]—that relies on successive insertion of vertices. A later, more common, algorithm from Bowyer and Watson [97,98], relying on the circumcircle property, is said by Shewchuk [99] to exhibit problems with numerical imprecision: the circumcircle test can, with unfortunate combinations of imprecise calculations, lead to inconsistency, and complete algorithmic failure. We consequently avoid the Bowyer-Watson approach.

It should be noted that the performance of the CDT step of the meshing procedure is considerably more important than that of initial triangulation since it is applied repeatedly, and with many more nodes. However, although with some configurations $\mathcal{O}(n^2)$ performance is expected, typically the cost is far lower with the meshes we use; this has also been reported with some explanation by Shewchuk (*ibid.*). Intuitively it is relatively easy to understand, as with the iterative steps of element subdivision and CDT enforcement one can expect the majority of the mesh to comply with the Delaunay property before it is re-enforced, as the mesh complied before the new nodes were added.

Lawson's algorithm does not directly use the circumcircle property, but another defining property instead: for a constrained Delaunay triangulation, if d is a diagonal (i.e. an internal edge) of the triangulation, then the quadrilateral associated therewith—the union of the two triangles sharing d—cannot have internal angles split by d whose total is less than π. The algorithm follows simply from this property: if such a diagonal breaks this rule then it is flipped, curing that particular angle problem, but potentially altering the status of neighbouring diagonals, and so on. The idea is shown in Figure 3.7 on the following page. The number of flips with typical cases is actually relatively low.

In Figure 3.5 on page 90 the algorithm is seen in context. The first triangulation is in fact the CDT; the second is a poor (from a finite element

[7] Voronoi diagrams, also called Dirichlet tessallations, are the geometric "dual" of Delaunay triangulations, and contain the same information [93]

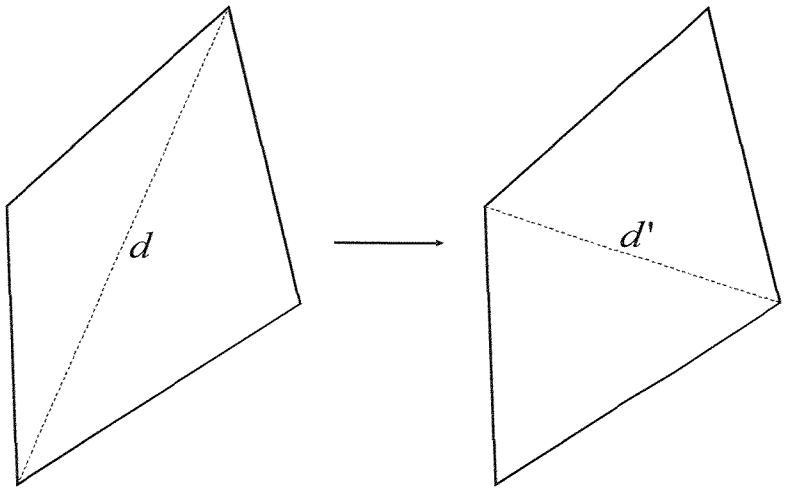

Figure 3.7: The Delaunay quadrilateral angle property and Lawson's algorithm. The sum of the angles split by the dashed diagonal d is less than π in the left-hand example. This can be rectified by flipping d, as shown, to give d'. In the single quadrilateral example here the process is complete, but were there neighbouring triangles these would need to be checked too.

perspective) triangulation. By flipping edge AB to give edge CD the situation is improved.

A very crude approach would check all internal edges for the requisite property, and upon finding a violating case, flip the edge, before starting again. (The search could not just continue because previously compliant edges might have had their status changed by flipping the first edge.) This would be extremely slow, however, and to ensure quadratic complexity a slightly more sophisticated approach is necessary. In a similar vein to the "ear removal" triangulation algorithm above, the status of edges is stored, and the implications of edge flipping carefully considered to minimise disruption of surrounding edges. It can be proven that such an approach must terminate (it might be feared that cyclic cases could arise, with repeated flipping to and fro of edges; this cannot happen). When nodes are added during

adaptive element subdivision, again the implications for surrounding edges are considered, and the work required to restore the CDT is minimised.

3.1.3 Delaunay Mesh Refinement

With the ability to enforce the CDT constraint—and hence ensure good quality elements within the constraints of the set of vertices—in place, the adaptive meshing algorithm is almost complete. It simply remains to find a strategy to add new nodes where needed. Assuming offending elements can be identified, the question remains of how to subdivide them without breaking the node connectivity requirement of § 3.1.1 on page 84.

Harriman *et al.* [33–38] used an initial structured mesh, apparently generated trivially over a rectangular or similarly simple region, followed by a "red-green" refinement strategy due to Bank [100]. Unfortunately that type of initial mesh generation restricts the problem domain to the realm of the trivial; and the "red-green" Bank refinement scheme is computationally inelegant, because it requires maintenance of two types of refinement, and removal of temporary edges during each refinement cycle. Further, the advantage of the "red-green" procedure is that it guarantees to introduce no angle smaller than half the smallest angle in the initial mesh, but the difficulty is that it says nothing about generating that initial mesh, which must of course have no small angles. For the geometric generality that we seek, then, it is not a complete solution, and if the initial mesh generation and refinement can be performed with the same basic tool (the CDT in our case) then it is unnecessary.

As has been mentioned, the element subdivision problem was solved for elements without external edges by adding a node at the centroid, splitting the triangle into three, as shown in Figure 3.4 on page 87. Elements with external edges are split on their longest external edge, yielding two new elements. The alternative strategy at the boundary is clearly necessary as otherwise external edges would never be split. Internal edges, while never

being split, can be flipped, leading to the same effect.

Initially a considerably more complicated algorithm, due to Ruppert [101], was implemented. This added new internal nodes (by this we mean those not on external boundaries) at bad element circumcentres (the circumcentre of a triangle is the centre of its circumcircle). Given the circumcircle Delaunay property this must lead to the splitting of the bad element, and usually others too. Some theory exists to justify this approach in terms of the quality of the elements produced. Unfortunately, with circumcentres not necessarily lying within their triangles it also requires cumbersome searching for the element containing the new node.

Another apparent difficulty with Ruppert's algorithm comes when a bad element circumcentre lies outside the domain. Here theory is used to show that such a point could only lie in the *diametral circle* of a boundary edge (diametral circle meaning the circle whose diameter is inscribed by the edge). Such a boundary edge would be termed *encroached*, and would be split by a new node at its midpoint. The elegant theory breaks down, however, with boundaries having only relatively small angles: there, cyclic encroachment can occur, with a new midpoint encroaching another edge, that being split, and so on. Not splitting at the midpoint, and using special formulae for the split point can alleviate this, but the algorithm becomes more complex.

For some time Ruppert's algorithm was used, and could be restored to the program, but it was found to be relatively slow and complex to maintain, and removed. On the other hand, the triangle quality of the current strategy is by no means perfect, particularly if the initial triangulation has extremely elongated elements. This typically happens when problems of infinite extent are approximated by very large domains, or where the domain has a high aspect ratio. Essentially, while the Delaunay property guarantees good quality triangles given the distribution of vertices, the location of vertices is still a limiting factor, and is not optimal with the centroid subdivision approach. While no simulation failed badly because of poor quality elements, in retrospect the Ruppert algorithm seems preferable from a robustness point of

view, and it might, in spite of its additional complications, yield better performance by producing meshes with fewer nodes and system matrices with lower condition numbers. Whichever element subdivision rules are used, however, the constrained Delaunay flipping algorithm seems efficient.

3.1.4 Other Approaches

Delaunay triangulation is one of the more recent techniques to be used for automatic mesh generation. Earlier algorithms tended to revolve around the *advancing front* method and *quadtrees* (in two dimensions—these become octrees in three dimensions), advancing front being the older.

As has been stated, the attraction of Delaunay methods is the minimum angle property. Advancing front and quadtree generators do not have this easily explicable attraction. Advancing front algorithms appear to have little theoretical support, but have been used widely, with success. For solving problems such as ours they rely on advancing from the boundary inwards, producing triangles as they go. The difficulties occur when the fronts meet in the middle. It is probably true to say that advancing front methods are becoming obsolete as they cannot offer the theoretical guarantees of quadtree and Delaunay approaches. They also do not translate well to three dimensions.

Quadtree methods and their higher dimensional analogues create a hierarchical partition of the domain. With a quadtree, an initial grid of four squares is laid over the domain, and squares containing high geometrical complexity are recursively subdivided until a certain threshold is reached. By warping the squares so produced, the mesh can be made to fit the boundary. Perhaps surprisingly, there have been proven theoretical guarantees on the angles produced by such methods, for instance by Bern *et al.* [102]. However, as documented by Shewchuk [99], quadtree methods typically produce several times as many triangles as necessary. Given their preference for axially-oriented edges this is understandable.

In sum, it is probably fair to say that in two dimensions, Delaunay triangulation is the most elegant and efficient, at least given our current requirements. As will be discussed more fully in Chapter 6, the minimum angle property does *not* generalise to three dimensions, so the central benefit of Delaunay triangulation disappears there. Attempts have been made to work around this, but it might be that octree methods, which seem to retain more of the characteristics of their two dimensional versions, work better in three dimensions. The problem of three dimensional meshing is a very active area of research, and there is no universally satisfactory solution. It is primarily for this reason that we stay in two dimensions, where meshing is closer to being a solved problem.

A second factor in mesh generation arrives with transient simulations. If the mesh is refined for each time step, the problem of locating new nodes within the old mesh (in order to evaluate field values) becomes crucial to the overall efficiency. Here the hierarchical methods mentioned above have *prima facie* advantages. We return to this in Chapter 6.

3.1.5 Mesh Data Structures

Mesh enrichment requires more complex data structures than those for simple, static mesh finite element solving. This complexity manifests itself in additional information that must be stored with elements, and in the way elements and edges are stored; but there are commonalities, for instance in the way node coordinates are used.

Since in an unstructured mesh nodes are shared by arbitrarily many elements, it is logical to store them only once to eliminate redundancy. Thus they might be stored in an array, with each element having three indices or pointers referencing nodes therein. This is not simply a question of space efficiency: the alternative approach of storing duplicate nodal coordinates allows greater potential for inconsistent data structures, and hence program bugs. An obvious data structure, then, for a triangular element, usable in

simple finite programs, would be a three element array of node indices per element pointing into a global array of node coordinates, as in Figure 3.8.

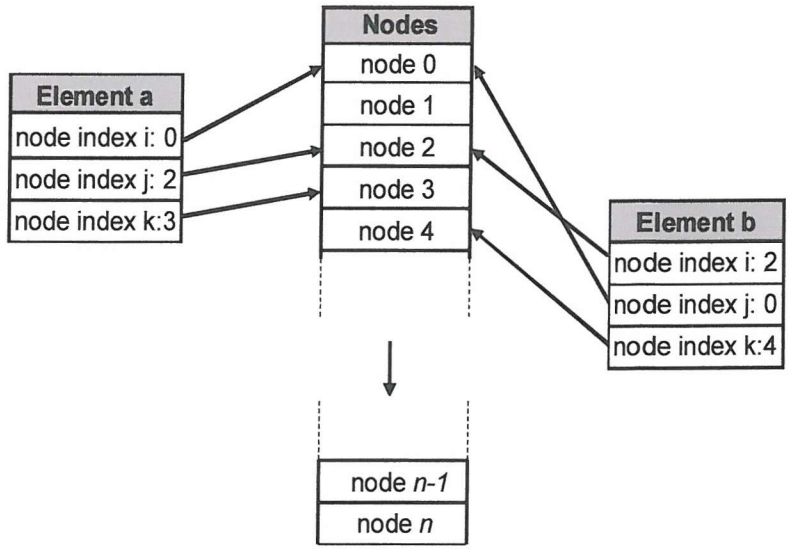

Figure 3.8: A simple element data structure. By using indirection, node coordinates are stored only once in a global node array. Two elements, a and b, are shown referencing the node array. They share the edge between nodes 0 and 2.

Apart from node coordinates, some provision needs to be made for boundary conditions, and it is here that it should be realised that the node index lists stored with elements implicitly define edges. If we select a convention (we, perhaps unwisely given the prevalence of the opposite standard, chose clockwise), the node index array ordering defines the edges—for instance we may take the node to implicitly refer to a following edge. We would then, if we did not need to alter the mesh, store boundary condition data in another three element list associated with each element; the linear system assembly requires no more information.

For mesh refinement, however, we need more information if we are to flip edges, as required by our constrained Delaunay meshing algorithm. This requires knowing which elements border each other—something not instantly

available from the data structure in Figure 3.8. By means of exhaustively searching the element and node lists this information could be extracted, but the process would be hopelessly slow. The solution is to ascertain upon initial triangulation, and maintain during the refinement process, information about the connectivity of the elements. A workable solution would be to store a second list with each element, with pointers or indices referencing the elements sharing each edge. This structure does introduce redundancy—fundamentally it encapsulates no new information—but without it performance would be unacceptable, so the potential for problems caused by incorrectly maintained mesh structures must be tolerated.

As well as edge flipping, we need to perform element and edge subdivision, as detailed previously. With the three element array approach, this becomes cumbersome and error prone, because array lists do not well represent circular edge patterns. Commonly, for instance, the code needs to step round the triangle from one edge to the next, but the action it takes depends if it is at the end of the element edge array (here it must loop round to the beginning again), or at the first or second element (in which case the edge index is incremented). The difference may seem trivial, but when it is magnified by the number of times such code is required it becomes a liability.

Therefore we elect, instead, to introduce another level of indirection, by storing with each element a pointer to a doubly (i.e. it can be traversed in both directions) circularly linked list [103] of edges, which by its nature has no beginning or end. For each element then, we have the topologically more faithful edge chain, in turn linked to the node list, shown in Figure 3.9. The new intermediate edge structure now holds boundary condition data, as well as node positions and pointers to its pair, if there is one, in a neighbouring element. The linking of these structures is now more complex, but easier to work with. In fact, in our implementation edges do not store boundary information, but point to segments of a special boundary data structure, not described here; however, this is not fundamental. The maintenance of all these links was the central challenge of the meshing code. Indeed, this is

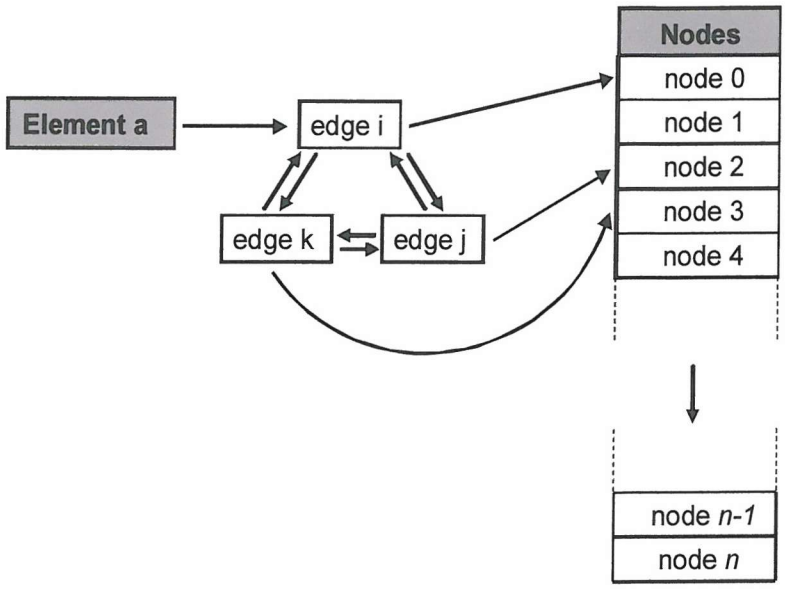

Figure 3.9: The mesh data structure used. Element a links to an edge in a circularly linked list. The edges in this list link in turn to the nodes. Of course, despite how it is shown, the linked list is linear in memory, as all things are, but traversal is dictated by pointers rather than memory contiguity. Note that it does not matter which edge in the linked list the element points to. Not shown are the links from edges to their pairs in neighbouring elements, or the back-links from edges to their respective element.

a significant factor, seldom discussed when considering the alternatives of mesh enrichment and regeneration.

3.2 Error Estimation and Current Calculation

With the ability to generate and solve meshes, the final step needed to complete the iterative cycle of Figure 3.1 is error estimation. Whereas much automatic mesh generation—for instance that of Shewchuk [99]—has been

devoted to optimising meshes for purely geometric factors, the future of finite element clearly lies in adaptive mesh refinement guided by the solution-dependent quantities of interest.

3.2.1 Previous Work

Zienkiewicz and Taylor [16] give an overview of error estimation, but primarily from a structural mechanics perspective. There certain *error norms*—measures of error—related, for instance, to the total deviation in stress, are typically in need of minimisation. Our requirements are rather different, but we use many of the same ideas translated into the electrochemical arena—something apparently not done previously. Zienkiewicz's focus has largely been on superconvergent patch recovery and related error estimators.

Various more mathematically-oriented practitioners have also pursued adaptive finite element. Babuška and Rheinboldt introduced *a posteriori* residual-based error estimators [104, 105] in 1978, and in 1984 Babuška and Miller introduced the notion of the "influence function" key to this work [106–108] (see below). Babuška and co-workers' approach was largely centred on residual-based estimators, and the cited papers deal with steady-state problems. Much of it is summarised in [109].

Through the nineties Eriksson and Johnson published a series of papers dealing with adaptivity in parabolic problems [110–114], including non-linear ones. When transient electrochemical problems are tackled their work will doubtless become more relevant to ours. Rannacher, Becker, and others have developed residual-based estimators for non-linear elliptic and transient problems [115–118] (collected together in [119]) based on similar duality arguments to those used by Harriman *et al.* Much of this work is devoted to the discontinuous Galerkin method—a different formulation from the standard Galerkin one used here—and so is somewhat divorced from our concerns. Of course, none of these authors addresses electrochemical problems, except peripherally when dealing with flow through chemical reactors.

Harriman *et al.* point the way [33–38] for adaptive electrochemical finite element. Previous practitioners, some using commercial finite element packages, have typically, if they have used adaptive finite element, adopted some standard error norm of the concentration field error. If, however, we desire an accurate current from the simulation, it is the error in this specifically that must concern us. It should be used to guide the mesh refinement, not some indirectly related quantity.

Unfortunately Harriman *et al.*'s approach has difficulties for the electrochemical practitioner. The error estimation theory is extremely mathematical. Numerous pages of abstruse (for a non-specialist) "Sobelev spaces" and "trace theorems" lead to very specific results, but this precision is, as we shall show, unnecessary for error-controlled simulations; it also does not necessarily imply any more efficiency in the estimator. The titles of all their papers on the subject advertise "guaranteed accuracy" of currents calculated using their techniques. To the layman this is potentially misleading, as the content reveals, because guarantees are contingent on having to hand the exact solution to a second, *dual problem* (defined by the adjoint equation, possibly with different boundary conditions), and "in general the analytical solution to the dual problem is not known... In this case the reliability of the... error bound can no longer be guaranteed" [33]. In certain cases, therefore, it is possible for the "bound" to underestimate the error.

Without guaranteed error bounds or wide generality to bolster the claim for such recondite mathematics, we are free to make modifications, as long as we can ensure—like Harriman *et al.*—that the *estimate* of the error of our solution is within a prescribed bound. Consequently we put aside the detailed theory and, focusing on the key idea presented in [33–38], develop something simpler and possibly more general.

The most important idea used is that of re-expressing the boundary integral current definition as a domain integral of both the solution field (somtimes termed the primal field) and the an *influence function* (see below). This was borrowed by Harriman *et al.* from the work of Babuška and Miller in en-

gineering [106,109]. It is a fundamentally important idea, and we shall use it extensively below, although it will be found that actually solving a separate influence function problem is frequently unnecessary. Harriman *et al.* restrict the influence function to be the dual, as do Rannacher *et al.* [115–120], which is something that our formulation does not demand, but which does appear to be more efficient—see § 4.6 on page 190.

3.2.2 Current Calculation

Until now our efforts have been directed to solving the mass transport equation for the dimensionless concentration field u. However, usually we are not concerned with this directly, but the current defined by integrating the flux from Fick's first law along the electrode surface. For instance, we have already given the definition of the dimensionless microdisc current:

$$i_{norm} = \frac{\pi}{2} \int_{r=0}^{r=1} \left. \frac{\partial u}{\partial z} \right|_{z=0} r dr \; . \tag{3.1}$$

Generally speaking, we wish to evaluate a boundary integral of the form

$$I(u) = \int_{\Gamma_e} \zeta \frac{\partial u}{\partial n} d\Gamma \; , \tag{3.2}$$

where Γ_e is the section of the boundary Γ occupied by the electrode of interest. ($I(\cdot)$ is here an operator on, sometimes called a *functional* of [121], the dimensionless concentration field, u). The weighting function ζ is either unity, in the case of Cartesian problems, or r in the case of cylindrically symmetric problems—see § 2.1.6 on page 61.

The difficulties of directly evaluating (3.2) are fairly severe. The obvious approach of simply substituting the approximate solution $\bar{u}$ and using Taylor series-type numerical differentiation to evaluate the integrand does not yield very accurate results. Various practitioners have adopted more sophisticated strategies, using field values further inside the domain to interpolate a smoother function to the boundary for numerical differentiation

(e.g. [87,122]). Reasonable results have been produced this way, but at bottom remain fundamental difficulties. Numerical differentiation, as copiously documented (see, e.g., [30]), is an inherently inaccurate operation. Further, it is intuitively unsatisfactory to produce an entire concentration field and yet use only a few points within it to calculate the desired quantity; it seems wasteful of effort. Yet another practical difficulty is simply that, given an arbitrary domain, the location of interpolation points is difficult to generalise.

But the most important problem arises when considering an adaptive refinement strategy guided by the error in the current. Given (3.2) it is impossible to say which elements contribute to the error in $I(\bar{u})$. Therefore it is essential, if we wish to bring all the laboriously assembled adaptive finite element machinery to bear on the problem of current calculations, to seek an alternative, equivalent, expression for the current. This final objection to (3.2) gives a clue as to how we should proceed: a domain integral, as opposed to the boundary integral of (3.2), would allow the possibility of adaptive element refinement, and also carries other advantages.

Adopting a very similar tack to that used in deriving the weighted weak problem statement, we imagine multiplying (2.1) by an as yet undefined function v' and integrating over the whole problem domain:

$$\int_\Omega v' \left[\nabla \cdot (a\nabla u) - \mathbf{v} \cdot \nabla u + qu + f\right] d\Omega = 0 \ . \tag{3.3}$$

As before we use (2.8) to rewrite the Laplacian-type term, whence

$$\int_\Omega \nabla \cdot (v' a \nabla u) d\Omega - \int_\Omega \nabla v' \cdot a\nabla u + v'\mathbf{v} \cdot \nabla u d\Omega + \int_\Omega v'[qu + f] d\Omega = 0 \ . \tag{3.4}$$

Again as previously, the divergence theorem can be used to rewrite the leftmost term as a boundary integral:

$$\int_\Gamma v' a \frac{\partial u}{\partial n} d\Gamma - \int_\Omega \nabla v' \cdot a\nabla u + v'\mathbf{v} \cdot \nabla u d\Omega + \int_\Omega v'[qu + f] d\Omega = 0 \ . \tag{3.5}$$

The new surface integral is clearly similar to (3.2). Aiming for this goal,

we define v' as having the following boundary values:

$$v' = \begin{cases} \zeta/a & \text{on } \Gamma_e \\ 0 & \text{on } \Gamma/\Gamma_e \end{cases} . \tag{3.6}$$

Immediately we can write down our alternative definition of the current as a *domain integral*:

$$I(u) = \int_\Omega \nabla v \cdot a\nabla u + v'\mathbf{v} \cdot \nabla u d\Omega - \int_\Omega v'[qu + f]d\Omega . \tag{3.7}$$

In practice we define $v' = \zeta v/a$, where

$$v = \begin{cases} 1 & \text{on } \Gamma_e \\ 0 & \text{on } \Gamma/\Gamma_e \end{cases} . \tag{3.8}$$

In fact (3.8) is overly stringent, since if $\frac{\partial u}{\partial n}$ disappears at any point on the boundary then the first integrand of (3.5) is zero anyway, and v may have any finite value there. This is more important than it might seem, since abutting contradictory Dirichlet boundaries caused the adaptive algorithm to fail to converge with the current error estimator developed in this chapter.[8] Fortunately, on account of this relaxation of the stipulations relating to v, this is not a practical difficulty as we usually have an insulating surround between electrodes and between electrodes and the bulk boundary.

Everything is as we wanted it, but we have introduced a new field, v. In the work of Harriman *et al.* this influence function is always the *dual*,

[8]The issue would appear to be a moderately subtle one, relating to integration of infinite integrands. We successfully integrate the theoretically infinite fiux at the electrode edge gradient discontinuity—refinement allows convergence to a meaningful value there. But where contradictory Dirichlet conditions impose a field discontinuity (note that the *field* is continuous at the edge of a microdisc) one must integrate the derivative of a discontinuous function, where the derivative itself has been estimated numerically. This would not seem to be a stable procedure. Using an error norm defined in terms of the field itself, e.g. the L_2 error norm in u (see later), allows such problems to be solved, but of course without current error estimates.

meaning that it not only must have the correct boundary conditions to make the domain integral give the correct answer, but for technical reasons it also satisfies the *adjoint equation* (see Appendix B on page 263 for the definition). Practically speaking, purely diffusive problems are *self-adjoint*, and so this means v must satisfy the same PDE as u; convective problems, on the other hand, will have a dual problem with convection in the opposite direction to the primal problem.

A key difference in our case is that the derivation of the error bound does *not* hinge on v being the dual solution. In order for the vector calculus theorems used to derive (3.7) to hold, v must have continuous first derivatives within the domain, as well as the desired boundary values, but no PDE need be involved in its definition at all (we always do use one to define it though). This offers a greater degree of flexibility which could be useful, but this not to say that the nature of v is irrelevant to the efficiency of the bound, as we shall see in the convective case. Indeed, the limited empirical evidence suggests that the dual is indeed the optimal form of v.

For the bulk of the problems examined in this work we are in the happy position of having a dual solution available with no extra work because the PDE is self-adjoint and the boundary conditions are in a convenient arrangement. If, for instance, we have pure diffusion, $u = 0$ on the electrode (Γ_e) and $\frac{\partial u}{\partial n} = 0$ or $u = 1$ everywhere else on Γ, then we can set $v = 1 - u$. This situation is actually fairly common, since it corresponds to a large class of diffusion-controlled generator-collector problems. We term it the *inverse dual* case. Harriman *et al.* cite the simpler but less common *self-dual* case of $v = u$, which applies, for instance, to the EC' simulations in the next chapter.[9] We can expect an approximate halving of the solution time where these simplifications apply (the problems are not coupled, so the computational ef-

[9]Note that both of these cases, in spite of this terminology, are self-adjoint: the distinction is purely one of boundary conditions. This terminology is perhaps slightly misleading in its implications for the meaning of "dual", but it allows a useful distinction to be made between different boundary condition configurations.

fort scales linearly with their number). Although commonly applicable, one or the other of these cases does not always hold because sometimes we wish to calculate the current at just one collector among several, say, in which case we need to pick this one out with a v of a different nature from u.

The instance of only one problem necessitating solution is perhaps more theoretically important than it seems at first glance, because it bears out an apparently unexplored relation with the variational FE formulation—at least in the case of Laplace's equation. Leaving out the $\mathbf{v}$, q and f terms, and setting $a = 1$, the current expression (3.7) becomes

$$I(u) = \int_{\Gamma_e} \frac{\partial u}{\partial n} d\Gamma = \int_{\Omega} \nabla v \cdot \nabla u d\Omega \,. \tag{3.9}$$

In the particular case of $v = 1 - u$ this is obviously

$$I(u) = \int_{\Omega} \nabla(1-u) \cdot \nabla u d\Omega \,. \tag{3.10}$$

Since the grad operator is linear, $\nabla(1-u) = -\nabla u$, so

$$I(u) = -\int_{\Omega} \|\nabla u\|^2 d\Omega \,. \tag{3.11}$$

(Note that norms without subscripts, as here, are assumed to be Euclidean unless otherwise stated.) The procedure is virtually identical in the self-dual case, the only difference being the absence of the minus sign in (3.11).

The negated integral is the energy functional (assuming no inhomogeneous Robin boundary conditions, as in our situation) minimised in the variational Laplace equation derivation [16]. This is perhaps a coincidence, but it could be useful, because the finite element solution, not being the exact solution, will always give a value for this quantity higher than the true one because the functional is positive definite. Thus, in some cases, disregarding other approximations such as numerical integration, the true current could be rigorously bounded from above by the approximate value. Crucially, in other words, the discretisation error of the finite element solution—the finiteness of its solution space—can only push the calculated current in one direction.

The possibilities for truly bounded currents are explored slightly further in Chapter 6. Here we note some aspects relevant to the present algorithm. Firstly, ignoring where necessary the inaccuracies of numerical integration and geometric approximation, we can have reasonable confidence that, where our simulation differs from another given value of the current, if it is lower, it *must* be more accurate. The Galerkin approximation guarantees to minimise (3.11) within a finite subspace of the infinite subspace of all admissible solutions. If our simulation finds a solution in its chosen (via adaptive meshing) subspace with a lower value of this functional, then it has improved on any higher alternative. A corollary of this is that, when later we generate smoothed gradient currents, with purported greater accuracy, we can be sure that, if things are as they should be, the improved approximation must be lower; if it is not, the smoothing has failed in an important sense.

Aside from approximations other than the finite element discretisation, there are limitations of these ideas—notably the functional minimisation only holds in self-adjoint cases—and we have only shown a connection in the case of self- or inverse dual problems, but given the potential advantages we discuss this line of thought further in the final chapter. It has not, as far as is known, been pursued previously in the context of current calculations.

3.2.3 Error Estimation

Now we have a suitable current expression, we need to find a means of estimating error in such a way that we can guide mesh refinement until the desired accuracy is reached. Specifically, following Figure 3.1, we need two types of error estimator, whatever the quantity in which we are interested:

1. An estimate of the global error in the current, to know whether further mesh refinement is needed (the *global error estimate*);

2. An estimate of each element's contribution to this global error, in order to guide mesh refinement (the *local error estimate*).

In fact the two are closely linked, but it is not always the case that the global error is the sum of the elemental errors; it depends on the form of the error norm. It is here that we diverge most significantly from Harriman *et al.* We pursue instead the approach of Zienkiewicz and others, but adapt it to our purposes. Since their treatment focuses on certain standard error norms we describe these first, putting aside current calculations momentarily. This has its own end, too, since electrochemists are not exclusively interested in current calculations: sometimes the shape of the concentration field might be all that is desired, and slightly paradoxically a mesh optimised for current calculations may not give the most accurate picture of this.

Error norms

Clearly the absolute error of the concentration at a single point is defined by

$$e = |u - \bar{u}| \,, \tag{3.12}$$

but since generally we are interested in the global error, we might take, in effect, the root mean square integral over Ω:

$$\|e\|_{L_2} = \sqrt{\int_\Omega (u - \bar{u})^2 d\Omega} \,. \tag{3.13}$$

This is termed the L_2 norm of the error, and would be used if we were interested in the fidelity of the simulation over the whole domain in a least-squares sense. For visual inspection of the concentration field it would be an obvious choice, but an alternative is also available that can be easier to calculate—the L_2 norm of the gradient error:

$$\|\mathbf{e_q}\|_{L_2} = \sqrt{\int_\Omega \|\nabla u - \nabla \bar{u}\|^2 d\Omega} \,. \tag{3.14}$$

In mechanics applications this is closely related to the *energy norm*, which is a logical choice there.

Typically the relative error is of more interest, so we would divide by a similar measure of the field magnitude. In the latter case we retrieve the relative gradient error:

$$\eta = \frac{\|\mathbf{e_q}\|_{L_2}}{\|\mathbf{q}\|_{L_2}} = \sqrt{\frac{\int_\Omega \|\nabla u - \nabla \bar{u}\|^2 d\Omega}{\int_\Omega \|\nabla u\|^2 d\Omega}} \,. \tag{3.15}$$

We therefore have a global error measure, and its relation to the local error measure should be clear. Evidently, from the fundamental definition of integration, error norms in the L_2 norm are square additive, so (dropping the norm subscript):

$$\|\mathbf{e_q}\| = \sqrt{\sum_{i=1}^{n} \|\mathbf{e_q}\|_i^2} \,, \tag{3.16}$$

where the sum runs over all n elements, and $\|\mathbf{e_q}\|_i$ is the error of element i. Practically, the global error measure calculation would entail calculating each element's error contribution. Having done so, we would have both the information to decide on the necessity of further refinement, and where such refinement would be needed. The global error would be compared with a prescribed tolerance (e.g. 5% relative error), and if too high, a criterion for individual error refinement would in turn be generated; any "bad" elements would then be subdivided.

Usually it is decided to try to distribute error equally over elements—an approach given theoretical backing by both Zienkiewicz and Taylor [16] and Braess [32], as well as used by Harriman *et al.* Assume the user has specified a relative error tolerance of $\bar{\eta}$. Then we must ensure that

$$\eta = \frac{\|\mathbf{e_q}\|}{\|\mathbf{q}\|} \leq \bar{\eta} \tag{3.17}$$

or

$$\sum_{i=1}^{n} \|\mathbf{e_q}\|_i^2 \leq \bar{\eta}^2 \|\mathbf{q}\|^2 \,. \tag{3.18}$$

This is the global error criterion that determines whether future meshing is required. If we assume equidistribution of error amongst n elements, then

we can substitute for the sum on the left to give

$$n\epsilon_i^2 \leq \bar{\eta}^2 \|\mathbf{q}\|^2 \,, \tag{3.19}$$

where ϵ_i is the measure of element i's error. Rearranging,

$$\epsilon_i \leq \frac{\bar{\eta}\|\mathbf{q}\|}{\sqrt{n}} \,. \tag{3.20}$$

This is the local error estimate. If, after it is determined that further global refinement is required, any element's flux error norm is found to be greater than the right-hand side of the inequality above, it is subdivided. The theory carries over, *mutatis mutandis*, for any other square additive norm. The current error norm—developed later—which is not square additive, results in a slightly different but analogous pair of criteria.

The difficulty with all the above, of course, is that we do not have the exact solution u, or its gradient $\mathbf{q}$. We can only hope in general, therefore, to *estimate* error norms. Numerous means of estimating quantities in order to circumvent this difficulty exist [16, 32]—residual, local Neumann and Dirichlet problems, dual element orders, and recovery methods to name four. The classification is complicated, and sometimes, as with Harriman *et al.*, more than one strategy is used in tandem. We make no attempt to exhaustively evaluate the alternatives; we solely present our strategy, and note the differences from that of Harriman *et al.*

Higher order solutions

A simple, perhaps obvious, means of gauging FE solution error is to solve with elements of two different orders—linear and quadratic for instance—and compare the results. Harriman *et al.* used a comparison between certain quadratic and linear solutions of the dual problem, in tandem with gradient jump quantities, to create a current error estimator tailored to a form of the current error norm. Alternatively, we can take a different, rather simpler and more general, step if we denote by $\tilde{u}$ the solution with higher order elements,

and substitute into (3.14) to give the estimate:

$$\|\mathbf{e_q}\|_{L_2} \simeq \|\bar{\mathbf{e}}_\mathbf{q}\|_{L_2} = \sqrt{\int_\Omega \|\nabla\tilde{u} - \nabla\bar{u}\|^2 d\Omega}\,. \tag{3.21}$$

In general it is wasteful to solve globally at a higher order, with many more degrees of freedom, solely to obtain an error estimate (it is seen in Appendices C and D that doubling the DOF would usually more than double the linear system solving time). Thus the motivation for local problem techniques above, and the *recovery* methods below. That said, if a higher order solution—quadratic, say—is available, then a lower order solution—linear in this case—can be extracted from it simply by ignoring some nodes. This was in fact how Harriman *et al.* produced the two orders of dual solution used in their error estimator, and how in this work we achieve higher-lower order comparisons for problems where recovery methods alone do not suffice. If recovery methods are applicable, then they will inevitably be more efficient, since they take the basic finite solution and compare it with a more accurate one, whereas quadratic/linear comparison, say, compares it with a *less* accurate one. One of the major findings of this thesis is that a combination of the two is effective.

Recovery methods

Recovery methods attempt to obtain an improved solution *without* solving again with higher order elements, either locally or globally, but with a type of smoothing. Generally this is simpler with element gradients rather than concentration values, partly because they are less smooth to begin with. Because it often happens that we are only interested in the gradient, this suits our purposes; and where we are not, we find we can augment the technique in an efficient manner. We shall not discuss methods for improved concentration recovery, even though they exist, for reasons that we explain presently.

Since inter-element concentration continuity is only C^0, the gradient is generally discontinuous at element boundaries. A one dimensional schematic

illustrates this in Figure 3.10 on the next page. If this discontinuity is smoothed out we might expect the resulting gradient field to more accurately reflect the true gradient field. This generally proves to be the case.

Recovery estimators are the most recent, and probably the most popular, type of estimator, owing in part to ease of implementation and general robustness (see Zienkiewicz and Taylor [16] and Babuška *et al.* [123] for extensive comparisons with residual and other estimators). Their popularity is also probably related to the widespread use of the energy norm in mechanics applications, which solely requires gradient recovery. In current error norms, depending on the mechanism, the field as well as the gradient could appear, negating some of their advantages.

In common with methods where higher order solutions are obtained, error norm estimates can be written down directly by substituting the improved solution $\tilde{u}$ (or its derivative) instead of the exact solution u in the norm definition, just as in (3.21). This offers a considerable flexibility advantage over residual-type estimators, where specific analysis is required depending on the error norm, no better exemplified than by the extensive mathematics underpinning Harriman *et al.*'s error estimator. At the same time, the method is considerably quicker, since no additional solution is required. As a consequence of these attractive properties, we use recovery estimators in our adaptive finite element implementation. We therefore describe the recovery process in detail next.

Gradient smoothing

For the main part we limit our discussion to linear elements, but most of the ideas can be transferred to higher order elements easily. We sketch later the practical implementation challenges for quadratic elements.

In 1974 Hinton and Campbell [124] proposed using global least-squares smoothing to produce improved stress fields in mechanical applications. This approach did not prove particularly efficient or accurate for error estimation.

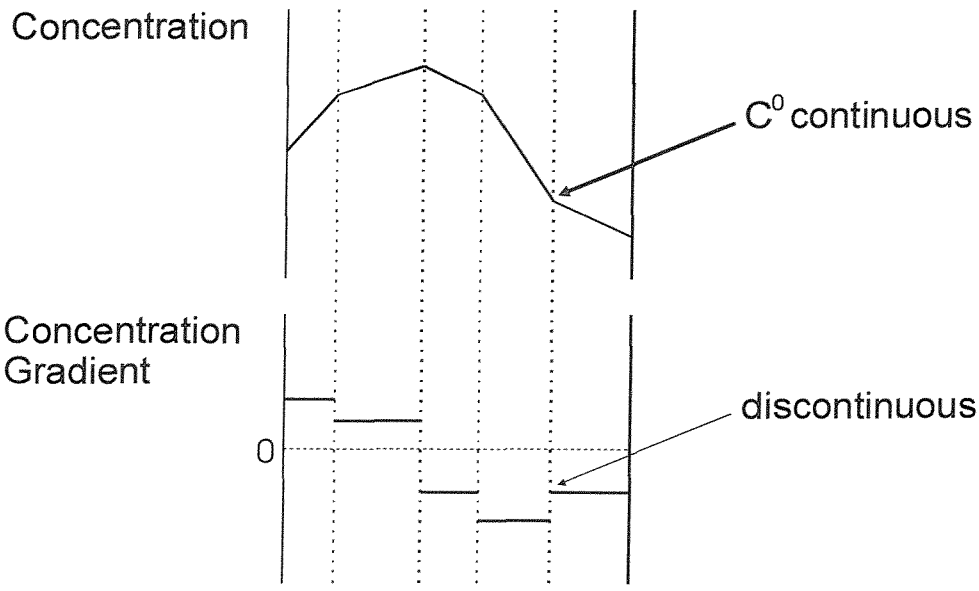

Figure 3.10: A C^0 continuous field composed of linear elements compared with its discontinuous gradient. Note that even with higher order elements the gradient is still discontinuous; the inter-node interpolations are just higher order curves.

By shifting the smoothing to local *patches* of elements, Zienkiewicz and Zhu [125] created a more useful, and greatly simpler, technique, termed here *nodal averaging.*[10] In nodal averaging, a continuous piecewise linear gradient field is constructed by approximating nodal gradient values by the average of the surrounding elements' gradients. The result is surprisingly accurate, and the procedure is fast enough to add no appreciable cost to the solution time. Nodal averaging was implemented in our simulation program, and used successfully in a number of simulations. There exist more sophisticated methods, though, relying on *superconvergence*, which were also implemented. These have more theoretical support.

[10]While nodal averaging had apparently been used before Zienkiewicz and Zhu for mechanics stress calculations, the idea of using it in an error estimator would seem to originate with them.

Before considering more advanced smoothing techniques, and two dimensional implementations, characteristics of the smoothing process are best elucidated by a one dimensional example. Taking as our problem a form of the one dimensional Poisson equation,

$$\frac{d^2u}{dx^2} = -4\pi^2 \sin(2\pi x) \quad 0 \leq x \leq 1 \,, \tag{3.22}$$

with the boundary conditions

$$u(0) = u(1) = 0 \,, \tag{3.23}$$

it is easy to verify that

$$u = \sin(2\pi x) \tag{3.24}$$

is the exact solution. This is plotted in in the first graph of Figure 3.11 on page 118. It is a curious fact that, for problems of this type, one dimensional Galerkin finite element with linear elements yields the exact solution at the nodes [16]. We can therefore plot, without actually performing the computation, the piecewise linear finite element solution with five regularly spaced elements. This is also shown in Figure 3.11 on page 118.

As Figure 3.11 illustrates graphically, the gradient obtained by differentiating the finite element solution is piecewise constant. The gradient so obtained is plotted in the second graph of Figure 3.11 on page 118—clearly a very crude approximation to the analytical gradient,

$$\frac{du}{dx} = 2\pi \cos(2\pi x) \,, \tag{3.25}$$

also plotted. However, by linearly interpolating the approximate gradients at element centroids, a more realistic approximation is secured. The result for the simple five element problem illustrates both the advantages and limitations of the method. For the most part the smoothed gradient is closer to the exact value, but not everywhere, and because the end nodes are only connected to one element the smoothed field there must be extrapolated from further in. With more elements the situation improves: here a minimum

occurs within an element, which cannot be expected to be handled well—refinement would quickly rectify this. All of these issues translate to higher dimensions.

It can be shown [125] that, under the Galerkin formulation, the most accurate points for the concentration in linear elements are at the nodes (at least for a class of self-adjoint problem). This is unsurprising given the form of nodal shape functions shown in Figure 2.2 on page 58: used as a weight function its strongest effect will be, one might expect, where it has the highest value—at the node. It is perhaps less intuitively obvious, but it can also be shown that in the same elements the most accurate points for the gradient are at the centroids of elements. (Formulae exist giving the location of the corresponding points in higher order elements.) These are termed *superconvergent points* because the order of convergence of the gradient there is higher than elsewhere. In the absence of singularities the concentration approximation for linear elements is, as stated earlier, $\mathcal{O}(h^2)$, whereas its gradient is, at most points, $\mathcal{O}(h)$. However, $\mathcal{O}(h^2)$ is achieved at superconvergent points, making these of special interest. The situation with convection and non-Galerkin formulations is not clear, and would seem to demand further investigation, but the empirical results of this work appear to offer a limited validation in those contexts.

Zienkiewicz and Zhu went on to show that a globally superconvergent gradient is obtainable by appropriately interpolating nodal superconvergent values, themselves obtained from superconvergent points within element *patches* (see below) surrounding them. This they termed *superconvergent patch recovery* (SPR). The stated advantage over nodal averaging is apparently greater robustness, with considerably more theoretical backing. The arbitrariness of averaging greatly increases in two dimensions, where any number of elements can surround a node—some researchers advocate the arithmetic mean of surrounding elements' gradients; others weight the values according to elements' sizes, or their angles at the node in question. SPR is a more systematic approach: it finds (with linear elements) the linear gradient field over

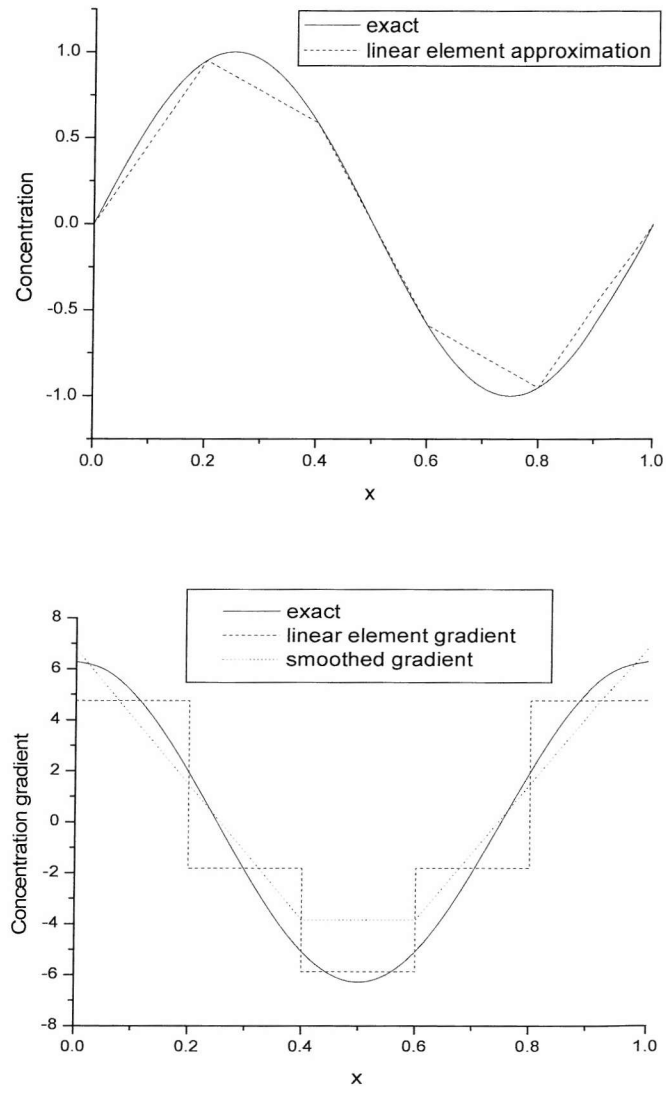

Figure 3.11: The linear element approximation to the solution $u = \sin(2\pi x)$, and the smoothed gradient thereof.

the patch with the minimum least-squares deviation at the superconvergent points. This boils down numerically to solving a small linear system.

It has been proven that, under certain mesh regularity assumptions, error estimators using the SPR gradient field are asymptotically exact, unlike, for instance, residual estimators—that is, they tend towards the exact error with increasing mesh density (*ibid.*). SPR is also apparently generally more robust than other approaches, although clearly the robustness of our specific estimator has not been examined. For these reasons, SPR was adopted as the default in our simulation program, although the option exists for nodal averaging—in fact the difference in results was small for the range of problems tested.

Although there is much empirical evidence justifying SPR, its theoretical underpinnings are not complete, particularly in more than one dimension and with triangular elements. Significant new technical difficulties arise in analytical studies when moving to two dimensions, as described by Babuška and Strouboulis [109]. Known results are fairly specific: under certain assumptions, for instance a uniform or quasi-uniform mesh, and for certain simpler PDEs, the existence of superconvergent points has been determined. For the Poisson equation triangular meshes may or may not have superconvergent points depending on the order of element and the exact (uniform) mesh pattern. Aside from the authors cited above, Zhang [126–129], Zhu [130, 131] and Schatz *et al.* [132] have published various results on superconvergence. The meshes used in this work are not in their generality covered by that work, and nor are the full variety of possible transport equations.

Babuška and co-workers [133] have gone as far as employing computer-based proof, *à la* Appel and Haken's famous resolution of the four-colour problem [134], in the search for superconvergent points. And yet their existence is far from resolved. All is not lost, though, as Babuška and Strouboulis usefully point out [109]: recovery-based error estimators rely on *cancellation of error*, and don't in fact require superconvergence to work. The special cases where superconvergence is proven to occur (and these are rare) we can

expect better performance, but this is not crucial. This is as we expect from the extensive numerical evidence in SPR's favour (*ibid.*).

Of course, in two dimensions the gradient is a vector field; but this simply necessitates that the procedure be performed separately for the two components. The main complication in a two-dimensional implementation is the variable topology of the mesh that leads to several distinct cases. Patches of various sizes can and have been used, but for ease of implementation we chose patches of minimal size for all cases where these were applicable. For non-boundary nodes, the mesh always has at least three surrounding elements with non-colinear centroids—the minimum required for the least-squares process to succeed—so these are relatively straightforward. (Failure is signalled by breakdown of the Cholesky factorisation—see below.) Each internal node can therefore have its superconvergent gradient values determined by the surrounding patch, as shown in Figure 3.12 on the following page.

Boundary cases appear to come in two distinct forms: corner nodes (where one element contains the node) and boundary nodes with two elements; and boundary nodes with three or more elements. The first case clearly cannot work as the internal case does, as there are not as many points as degrees of freedom in the least-squares process. It happens, however, that the final case cannot in general either, since the surrounding elements can—and often do—have co-linear centroids, and lead to a singular linear system (this cannot happen with internal nodes with a valid mesh, as it would imply infinitely thin elements). Thus it is not safe to deal with any boundary nodes in the same fashion as internal nodes. Instead the average of the overlapping internal node patches' values is taken. In the corner case, of course, there is usually only one patch to use (the only alternative is none—see below); at others, two or more. The idea is illustrated in Figure 3.13 on page 122. This averaging process at boundary nodes is hard to justify theoretically, and indicative of the lower quality of smoothed gradients at boundaries, as seen earlier.

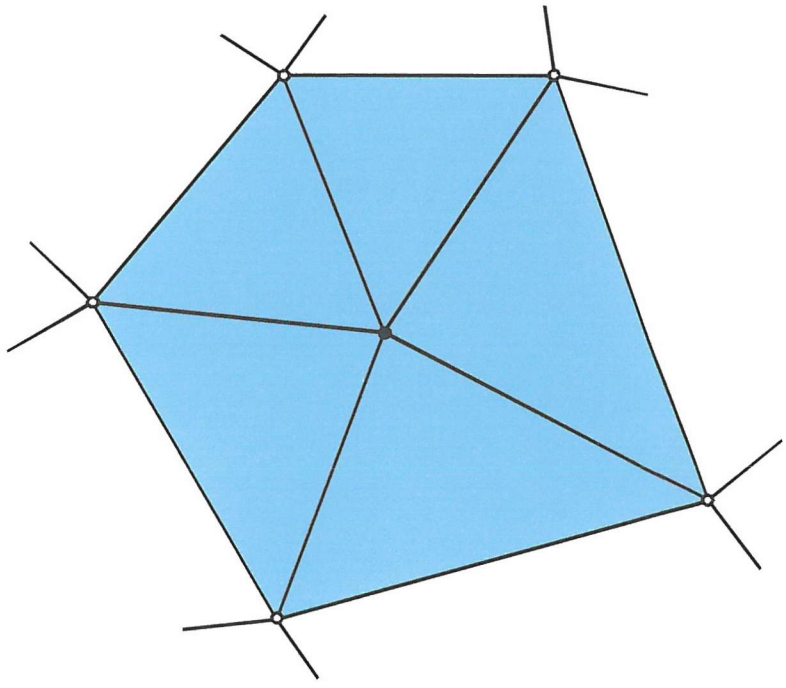

Figure 3.12: An internal node SPR patch. The shaded area is the patch of elements whose superconvergent points are used in conjunction with least-squares minimisation to determine a smoothed value at the central black node. The white node values are determined by other, similar patches.

The algorithm used to perform SPR over the mesh is discussed later. Two final points must be made, however. Firstly, node-centred patches are, by themselves, not sufficient to cover all cases. The minutiae are not important, so we mention it briefly, but some meshes can throw up corner nodes that cannot form part of internal node patches. In these particular and relatively rare cases *element-centred* patches are used in our implementation. Secondly, with quadratic elements, even an internal node-centred patch can cause failure if the surrounding sample nodes lie on a conic section [135]—in other words, quadratic element SPR lacks the topological guarantees available with linear elements. In this case we augment the patch with additional

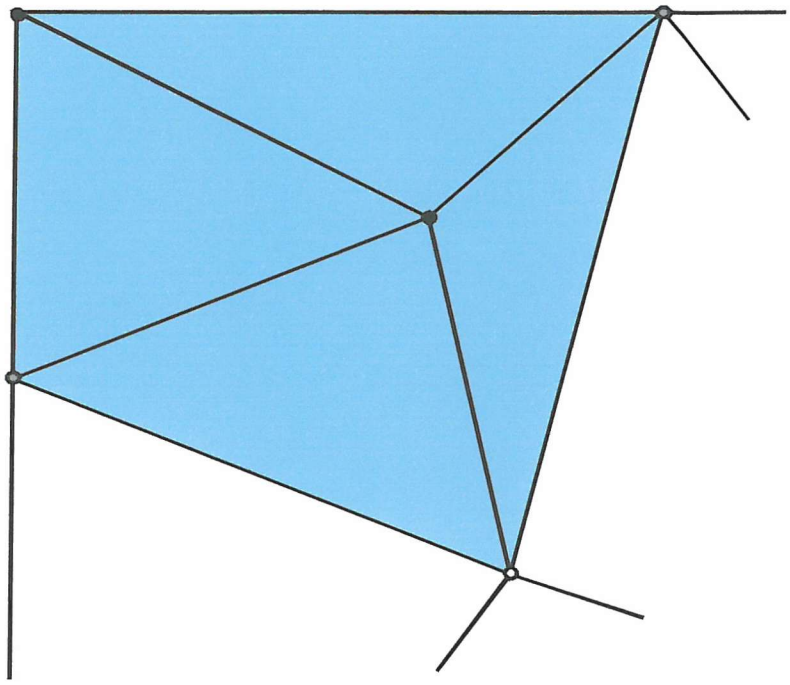

Figure 3.13: An SPR patch containing nodes on the boundary. The black nodes' values are entirely determined from this patch; the grey nodes have a contribution from this patch; and the white node is determined by its own patch.

elements. Neither of these niggling facets is documented in any source cited. While they do nothing to damage SPR theoretically, and are rare enough that they wield little influence on numerical results, they do require require a disproportionate programming effort to address.

The exact formulation of SPR is given in a series of papers by Zienkiewicz and Zhu [136–138]. In the case of linear triangular elements, for each smoothed field component (there are two per concentration field modelled) we find the three constants that define the plane it describes within the patch. The least-squares procedure, as is documented in standard numerical analysis texts [90], yields in one relatively primitive form[11] a symmetric positive def-

[11] This is the so-called *normal equation* form. It gives poor condition numbers, relatively

inite matrix of size n, where n is the number of degrees of freedom. Thus for linear elements our use of SPR boils down to solving two 3×3 symmetric positive definite matrices for each node. This can be efficiently achieved with Cholesky factorisation (*ibid.*; see also Appendix C), which is particularly suited to such matrices, and allows one factorisation to be used for all gradient field components, potentially significantly reducing the solution time, depending on the number of fields. With the plane equation available, substituting a pair of coordinates within the patch yields a superconvergent gradient value at that point.

At the end of either the nodal averaging or SPR process, a higher order gradient field is available, of the same order as the field solution itself. The process by which it is obtained is irrelevant to the purpose to which it is put. In all cases, the smoothed gradient is substituted into the error norm expression in place of the exact solution, sometimes in combination with approximations to the field itself, and the norm evaluated.

3.2.4 Current Error Estimation

The standard error norms described by Zienkiewicz and others are not directly related to the error of the current, but the basic idea of using comparison of lower and higher order solutions can be adapted to our purposes. This still gives a relatively simple error estimator, and avoids the complications of the approach of Harriman *et al.*

Our starting point is the domain integral current expression (3.7), substituting the approximate primal and influence function fields $\bar{u}$ and $\bar{v}$:

$$I(\bar{u}) = \int_\Omega \nabla\bar{v} \cdot a\nabla\bar{u} + v'\mathbf{v} \cdot \nabla u d\Omega - \int_\Omega \bar{v}[q\bar{u} + f]d\Omega \,. \tag{3.26}$$

For explicatory purposes we initially discard the second integral, and simplify the first by specialising to $f = q = 0$, $a = 1$, $\mathbf{v} = \mathbf{0}$:

$$I(\bar{u}) = \int_\Omega \nabla\bar{v} \cdot \nabla\bar{u} d\Omega \,. \tag{3.27}$$

speaking, but the matrix here is very small so this is not a problem.

Assuming we have more accurate approximations to the gradient fields, denoted by $\nabla\tilde{u}$ and $\nabla\tilde{v}$, obtained by recovery procedures as detailed above, we can define another, more accurate, current expression:

$$I(\tilde{u}) = \int_\Omega \nabla\tilde{v} \cdot \nabla\tilde{u} d\Omega \,. \tag{3.28}$$

The modulus of the difference, then, is the most obvious[12] estimate of the absolute current error:

$$\xi_1 = \left| \int_\Omega \nabla\tilde{v} \cdot \nabla\tilde{u} d\Omega - \int_\Omega \nabla\bar{v} \cdot \nabla\bar{u} d\Omega \right| \tag{3.29}$$

$$= \left| \int_\Omega \nabla\tilde{v} \cdot \nabla\tilde{u} - \nabla\bar{v} \cdot \nabla\bar{u} d\Omega \right| \,. \tag{3.30}$$

However, implicit in the analysis for the L_2 norm in §3.2.3 was the assumption that the integrand was positive. This allowed the distribution of the error over the elements, and its minimisation by subdivision. With a potentially negative integrand the error in one place could be cancelled by that in another: subdivision could *increase* the error estimate.

An obvious solution would be to use the less tight bound defined by moving the modulus sign inside the integral. This is valid because of the elementary inequality:

$$\left| \int_\Omega \nabla\tilde{v} \cdot \nabla\tilde{u} - \nabla\bar{v} \cdot \nabla\bar{u} d\Omega \right| \leq \int_\Omega \left| \nabla\tilde{v} \cdot \nabla\tilde{u} - \nabla\bar{v} \cdot \nabla\bar{u} \right| d\Omega \,. \tag{3.31}$$

So another, more conservative, error estimate is

$$\xi_2 = \int_\Omega \left| \nabla\tilde{v} \cdot \nabla\tilde{u} - \nabla\bar{v} \cdot \nabla\bar{u} \right| d\Omega \,. \tag{3.32}$$

This estimate was found to be workable, but it overestimated the error considerably.

The estimate eventually settled upon is in the same spirit as ξ_2, but it enforces positiveness at an *element* level, not on a point by point basis.

[12] Harriman *et al.* use a marginally different form as the starting point for their calculations, similar in its essential characteristics.

Thus negative contributions can cancel positive ones within an element, and the bound is less conservative, making for faster simulations. Consider the practical definition of ξ_1:

$$\xi_1 = \left| \sum_i^n \int_{\Omega^e} \nabla\tilde{v} \cdot \nabla\tilde{u} - \nabla\bar{v} \cdot \nabla\bar{u} d\Omega \right| . \tag{3.33}$$

An inequality like that of (3.31) is apparent:

$$\left| \sum_i^n \int_{\Omega^e} \nabla\tilde{v} \cdot \nabla\tilde{u} - \nabla\bar{v} \cdot \nabla\bar{u} d\Omega \right| \leq \sum_i^n \left| \int_{\Omega^e} \nabla\tilde{v} \cdot \nabla\tilde{u} - \nabla\bar{v} \cdot \nabla\bar{u} d\Omega \right| \tag{3.34}$$

Thus we define:

$$\xi_3 = \sum_i^n \left| \int_{\Omega^e} \nabla\tilde{v} \cdot \nabla\tilde{u} - \nabla\bar{v} \cdot \nabla\bar{u} d\Omega \right| . \tag{3.35}$$

This was the form of the global absolute error estimate used for all the current controlled simulations.

The relative error is then

$$\eta = \frac{\sum_i^n \left| \int_{\Omega^e} \nabla\tilde{v} \cdot \nabla\tilde{u} - \nabla\bar{v} \cdot \nabla\bar{u} d\Omega \right|}{\left| \int_\Omega \nabla\tilde{v} \cdot \nabla\tilde{u} d\Omega \right|} . \tag{3.36}$$

(The denominator's modulus sign exists to avoid calculations being hostage to the sign of the current.) As always, if further mesh refinement is to be avoided this must obey the global criterion

$$\eta < \bar{\eta} , \tag{3.37}$$

where $\bar{\eta}$ is the user-specified tolerance, e.g. 5%.

The form of (3.36) gives the form of the local error criterion—it is not square additive:

$$\epsilon_i \leq \frac{\bar{\eta}}{n} \left| \int_\Omega \nabla\tilde{v} \cdot \nabla\tilde{u} d\Omega \right| . \tag{3.38}$$

We notice that smoothed gradients will be sufficient for the evaluation of this error estimate, as only the gradient appears. But this is only true under the assumption $q = 0$ (f is irrelevant to this, and arbitrary a has no

profound effect).[13] For the EC' mechanism, where $q = -K$ say, the error estimate becomes

$$\xi_3 = \sum_i^n \left| \int_{\Omega^e} \nabla\tilde{v} \cdot \nabla\tilde{u} + K\tilde{v}\tilde{u} - \nabla\bar{v} \cdot \nabla\bar{u} - K\bar{v}\bar{u}d\Omega \right| , \qquad (3.39)$$

and clearly two different orders of u and v are required, as well as two different orders of their gradients. This case is crucially important, as extension to general mechanisms with homogeneous reaction coupling requires this case to be solvable as a prerequisite. We explain our approach to this generalisation, and its rationale, in the next section.

3.2.5 A Hybrid Approach

Although SPR has so far been explained mainly in the context of linear elements, we choose to use quadratic ones in the main. Reference [16] tells us that the approximate field itself, $\bar{u}$ will have $\mathcal{O}(h^3)$ convergence using quadratic elements. It also tells us that the gradient, $\nabla\bar{u}$, obtained from differentiating $\bar{u}$ is only $\mathcal{O}(h^2)$.[14]

SPR, when applied to $\nabla\bar{u}$, generally[15] yields an improved gradient estimate $\nabla\tilde{u}$ which is $\mathcal{O}(h^3)$ for quadratic elements. In the E mechanism case,

[13]The issue of convective terms is less clear. In this section we have used two orders of both the influence function and primal fields. If only one order of v were used, which would still on the face of it yield a usable error estimator, then convective terms would not necessitate the approach used for the $q \neq 0$ case. With orders of v the demands made by convective terms are much the same as those of the homogeneous reaction term.

[14]Note that, as previously mentioned, these convergence rates are only strictly applicable in the absence of singularities. However, according to Zienkiewicz and Taylor [16] near-optimal meshes from adaptive refinement can practically recover them. Moreover, empirical evidence in our specific area lends weight to the approach pursued here; abandoning the attempt to match convergence rates, for instance by solely using higher/lower order solutions, dramatically slows our simulations.

[15]The situation is not simple. It appears that with triangular elements superconvergence is not proven theoretically, although some numerical and theoretical evidence suggests so-called ultraconvergence can give $\mathcal{O}(h^4)$ in some cases. Generally speaking it seems reasonable to assume roughly $\mathcal{O}(h^3)$.

where only gradient terms appear in the error estimate, we would therefore ignore the order of $\bar{u}$ and simply note that we would be comparing expressions with integrands of $\mathcal{O}(h^2)$ and $\mathcal{O}(h^3)$. This is true also of various standard error norms such as the L_2 gradient norm and the energy norm in §3.2.3. Consequently references do not address the case where we need alongside these two orders of the field gradient two orders of the field itself.

The idea of comparing two different orders of approximation in order to assess the accuracy of a numerical technique is not new. In the field of solving ODEs the Runge-Kutta-Fehlberg technique [30, 90] is still popular, for instance. This uses two different orders of approximation for controlling adaptive step size. The idea hinges on the notion that the first Taylor series term discarded by the lower order approximation dominates all the other discarded terms, and is hence a reasonable approximation to the *truncation error* (*ibid.*) of the numerical technique.

So far in the discussion we lack two orders of approximation of u: we have $\mathcal{O}(h^3)$ but nothing else. We could attempt a recovery procedure on the field itself. These techniques are not well developed or widely used, however. Besides, assuming by extension that they yielded a $\tilde{u}$ of $\mathcal{O}(h^4)$, we would then find the comparisons of the field and its gradient would not match: ∇u would be $\mathcal{O}(h^2)$ versus $\mathcal{O}(h^3)$, whereas u itself would be $\mathcal{O}(h^3)$ versus $\mathcal{O}(h^4)$.

Alternatively we could discard recovery methods entirely, and effectively solve using two different orders of element. In practice we would, as Harriman *et al.* did, solve with quadratic elements and, for the linear element version, discard edge nodes. The result would be a field of $\mathcal{O}(h^2)$ and $\mathcal{O}(h^3)$, and, differentiating the two, gradients of $\mathcal{O}(h)$ and $\mathcal{O}(h^2)$. A mismatch remains.

The mismatch is not a fatal problem. EC' mechanism problems were successfully simulated, for instance, with the quadratic/linear approach just described; our approach is not intrinsically tied to recovery methods. However, the rate of convergence was slow. The gradient comparison dominated because the difference between $\mathcal{O}(h)$ and $\mathcal{O}(h^2)$ was greater than that between $\mathcal{O}(h^2)$ and $\mathcal{O}(h^3)$—we were in a sense wasting higher order information

about the magnitude of the error.

The fairly obvious solution was to combine the two approaches. In our final version we use linear and quadratic solutions for u—the linear solution being denoted by $\bar{u}$ and the quadratic by $\tilde{u}$—and the derivative of $\tilde{u}$ for $\nabla\bar{u}$, while $\nabla\tilde{u}$ was obtained by recovery methods. Thus the first two terms in the integrand of the EC' error estimate (3.39) were $\mathcal{O}(h^3)$, and the second two $\mathcal{O}(h^2)$. The result, while more complicated, and apparently not documented in the literature, was considerably more efficient, and produced accurate results in short times, as is seen in the next two chapters. (The analogue for linear elements also works, perhaps surprisingly, but can fail with certain error norms. In any case, it is far slower.)

3.2.6 Possible Theoretical Advantages

The chief motivation of the theory just described was simplicity, but it could be that this method is more easily generalised than that of Harriman *et al.* This seems to be so with heterogeneous kinetics, where there is no barrier to immediate application of our algorithm. Another very important, but unexplored, area is transient simulations, where things are much less certain (some speculations can be found in [139]).

Heterogeneous Kinetics

As we have seen in §1.1.4, the full electrode kinetics expression relates the reactant flux to a linear combination of the concentrations of it and its product at the electrode surface. But for the sake of argument we need only consider the totally irreversible case in dimensionless form (see § 4.3.1 on page 162):

$$\frac{\partial u}{\partial n} = Ku \,. \tag{3.40}$$

This is a Robin boundary condition, as opposed to the Dirichlet condition applied in reversible systems [2].

Harriman *et al.* use the fact that

$$\int_{\Omega} \nabla(u - \bar{u}) \cdot \nabla v d\Omega = 0 \tag{3.41}$$

in the derivation of their error bound. This reposes on the exact satisfaction of Dirichlet boundary conditions by finite element approximations[16] because among other things it requires that $u = \bar{u}$ on Γ_e. Robin conditions are not exactly satisfied, but even if they were we would not in general expect the exact and approximated concentrations to coincide where such a condition holds. Since our approach does not rely on (3.41) we can set a heterogeneous kinetics boundary condition on a portion of Γ_e for the primal problem (the influence function problem still always has the Dirichlet condition $v = 1$ on the electrode).

The easy extensibility to quasi-reversible systems would seem to be a useful advantage to our approach.

Extension to Transient Simulations

The most fundamental difficulty of the theory of [34–38] from an electrochemical point view is in the arduousness of its extension to transient simulations. The basic problem is that the dual problem is a *backward in time* transient problem, which is very hard to solve efficiently. While Harriman *et al.* have produced an extension of it to transients [140], it is heuristic, and it is possible that an equally or more satisfactory approach could derive from our methods. Since, as we shall see, a non-dual influence function can yield results in reasonable times it does not seem inconceivable that, used with care, this fact could yield a transient simulator of at least a similar efficiency. In other words, the lack of hard reliance on duality might prove to be more of an advantage than steady state problems can illustrate.

[16] They are "built into" the space of functions from which the approximation is drawn.

3.2.7 Summary

We use nodal averaging or SPR-obtained gradients in combination with equations such as (3.36) and (3.38) as the basis of all our current-controlled simulations. In cases where homogeneous reactions bring in field terms we use two different orders of solution to furnish a comparison. Based on the theory available—SPR generally increases the gradient order by one; differentiation yields a gradient of an order one lower than the field—the matching of orders so obtained applies to elements of any order.[17]

The specific current error estimate has not been used before, but is a logical extension of previous work. Its derivation is considerably simpler than Harriman *et al.*'s, and it gives us the opportunity to use generic improved solution recovery procedures in combination with specific norms. In electrochemical simulations where the error in the current is not the overriding concern, the norm could simply be changed, keeping the underlying procedures, and could in principle be an integral of any function of u and its gradient. The extension of Harriman *et al.*'s approach to other norms is not as clear.

3.3 Conclusion

At the end of this chapter we have a mesh refinement algorithm and an error estimator that are general enough for our purposes. Indeed the estimator would seem to be at least as applicable and extensible as any so far proposed.

This theory was implemented in a program ultimately of around 20000 lines of C++, with a fair degree of generality in the problems that it could simulate. The extent of its flexibility is perhaps best exemplified by Appendix E, where sample simulation input files are presented. Apart from the aforesaid lack of transient or non-linear mechanism capability, its third major limitation is confinement to solving for only one species at a time. Thus it

[17] But note the caveats with regard to singularities.

cannot simulate, for instance, the $DISP1$ mechanism [2], but it can simulate a type of ECE mechanism. This deficiency is not due to any fundamental problem, and it would seem relatively straightforward to correct.

Armed with a general means of solving electrochemical problems, we can now see what results may be obtained.

Chapter 4

Microelectrode Simulations

Implementation of the theory of the previous chapter results in a relatively flexible simulation system. Its general meshing approach, while confined to two dimensions, and currently to domains without holes, allows a wide range of electrode geometries. Using the simple weighting approach of § 2.1.6 on page 61, both systems with translational and rotational symmetry can be modelled.

Mechanistically, the program is principally limited by only solving for one concentration field at a time, where many of the more complex mechanisms require two or more coupled fields to be solved for simultaneously. Additionally, one is restricted to first order homogeneous and heterogeneous reactions, because the algebraic system solver can only handle linear equations. Nonetheless, some simple common mechanisms involving homogeneous steps—EC' and irreversible ECE for instance—can be incorporated. Pure diffusion control and totally irreversible electrode kinetics can clearly be modelled in all cases, and in some general reversibility can be incorporated by use of the right normalisation procedure, as shown below.

Given its general importance, and treatment by other finite element practitioners, the inlaid microdisc is an obvious choice for validation of our simulation algorithm, and in this chapter fundamental tests of the effect of element order and so forth are conducted using it. Crucially, it exhibits the type

of boundary singularity that much of the theory is designed to address. Of more practical importance are E mechanism recessed microdiscs and EC' microdisc configurations, where no simple exact solutions exist. To test systems with translational symmetry, microbands are simulated in three different configurations. These also provide an opportunity to test the adaptive FE program against BEM. Finally, in order to test convection, the channel flow microband case with a simple E mechanism is simulated. The input files used to simulate most of these cases are given in Appendix E.

Axes along which the simulation settings could be varied include the element order, various iterative solver settings, and the means of gradient smoothing (averaging or SPR). A lengthy investigation could be conducted on the effect of these, but given the focus on electrochemistry we do not present an exhaustive examination. Instead, the effect of some of these is shown in the context of the E mechanism microdisc (where the analytical solution allows conclusions to be drawn with more confidence than other cases), alongside an examination of the importance of domain size. As noted in the previous chapter, SPR is used for gradient smoothing in all simulations, and unless otherwise noted quadratic elements are employed. For purely diffusive problems, SGS preconditioned conjugate gradient is used as the linear solver; for convective-diffusive ones, Jacobi preconditioned BiCGStab.

Since the program is limited to steady state cases, all problems in this chapter are assumed, without explicit mention, to be steady state.

4.1 E Mechanism Inlaid Microdisc

We begin our testing of the finite element program with a case where an exact analytical solution exists: the reversible E mechanism in an inlaid microdisc geometry. Not only can we compare our results with the analytical current, but we can compare performance with Harriman *et al.*'s finite element implementation. Another benefit is that, for testing purposes, the

known analytical concentration field may be imposed on bulk boundaries, eliminating domain truncation error (see below).

4.1.1 The Problem

The problem is one of diffusion of a species, A, from bulk solution to a small disc-shaped electrode embedded in an insulating surround, where a reversible reaction occurs. Assuming the disc to be geometrically perfect, and the solution expanse to be infinite, the three dimensional problem may be reduced to two dimensions: the rotational cross-section on which we solve the problem is shown in Figure 4.1, with the microdisc radius normalised to one.

By a reversible E mechanism we mean homogeneously inactive species A and B undergoing a reaction at the electrode of the form

$$\mathrm{A} \pm ne^- \rightleftharpoons \mathrm{B} \tag{4.1}$$

with a potential-dependent surface equilibrium constant δ. Assuming purely diffusional mass transport, the concentrations of A and B, denoted by c_A and c_B, both obey Laplace's equation:

$$\nabla^2 c_A = \nabla^2 c_B = 0 \,. \tag{4.2}$$

At the electrode the assumption of equilibrium implies a constant surface concentration ratio:

$$\frac{c_A^s}{c_B^s} = \delta \,. \tag{4.3}$$

Additionally, flux balance demands that, at the electrode surface,

$$D_A \frac{\partial c_A}{\partial n} = -D_B \frac{\partial c_B}{\partial n} \,, \tag{4.4}$$

where D_A and D_B are the diffusion coefficients of A and B. These two conditions hold on Γ_1 in Figure 4.1. Far away from the electrode $c_A \rightarrow c_A^*$ and $c_B \rightarrow 0$. On the insulating surround a no-flux condition holds for both

species (i.e. $\frac{\partial c_A}{\partial n} = \frac{\partial c_B}{\partial n} = 0$), and on the rotation axis we have also to impose zero flux to ensure symmetry (if the gradient of the concentration there were non-zero the concentration would be changing along the radial coordinate at $r = 0$, and the symmetry would be destroyed). Therefore $\frac{\partial c_A}{\partial n} = \frac{\partial c_B}{\partial n} = 0$ on Γ_2 in Figure 4.1.

Clearly it is preferable to simulate only one species, and this is possible by considering a relation between the species' concentrations. Consider a possible form for c_B:

$$c_B = \frac{D_A}{D_B}\left(c_A^* - c_A\right) \ . \tag{4.5}$$

This form of c_B satisfies Laplace's equation if c_A does, since

$$\nabla^2 c_B = -\frac{D_A}{D_B}\nabla^2 c_A \ . \tag{4.6}$$

Mass balance at the electrode is clearly also satisfied. The only other condition is (4.3), which implies that

$$\frac{D_B c_A^s}{D_A\left(c_A^* - c_A^s\right)} = \delta \ . \tag{4.7}$$

This may be rearranged to give the surface concentration of species A in terms of known constants:

$$c_A^s = \frac{D_A c_A^* \delta}{D_B + D_A \delta} \ . \tag{4.8}$$

By this means species B, in spite of being coupled to species A, is eliminated from consideration.

The problem is now one of solving $\nabla^2 c_A = 0$ with the bulk concentration c_A^* at the far field boundary, and the known surface concentration c_A^s at the electrode. Introducing the dimensionless concentration,

$$u = \frac{c_A - c_A^s}{c_A^* - c_A^s} \ , \tag{4.9}$$

$u = 0$ at the electrode surface, and $u \to 1$ towards the bulk. Evidently u so defined obeys Laplace's equation,

$$\frac{1}{r}\frac{\partial}{\partial r}\left(r\frac{\partial u}{\partial r}\right) + \frac{\partial^2 u}{\partial z^2} = 0 \ , \tag{4.10}$$

and a no flux condition, $\frac{\partial u}{\partial n}$, still applies on Γ_2.

The bulk boundary condition $u \to 1$ highlights an important problem with our implementation as it stands: infinite domains cannot be incorporated exactly. The mathematical problem requires that we ensure

$$\lim_{r,\, z \to \infty} u = 1 \,. \tag{4.11}$$

Normally we must, with elements of finite extent, approximate this by setting

$$u_{z=z_{max}} = 1 \,, \tag{4.12}$$

$$u_{r=r_{max}} = 1 \,, \tag{4.13}$$

where z_{max} and r_{max} are numbers large relative to other domain dimensions. This is a not completely satisfactory solution, and it requires careful checking. Unless something is known of the nature of the solution beforehand, any numbers so obtained must be verified by running the simulation with several domain extents, checking for any changes on the scale of the error bound. If there are none then one can assume that the result with an infinite domain has been approached. A possibly superior strategy using infinite elements is discussed in Chapter 6.

In the particular case of the microdisc, where the exact analytical field values are known, we can eliminate this source of inaccuracy by imposing the analytical boundary values. (This is of course only useful for validation purposes: if we knew the solution we would not need to conduct simulations.) We initially impose the analytical values on a small domain of $r_{max} = z_{max} = 2$ in order construct an ersatz "infinite" domain. This eliminates geometric approximation as a source of error for the purposes of testing various other aspects of the simulation, such as the element order. We then use a more realistic finite domain, and vary its size in order to test its effect on the simulated current values.

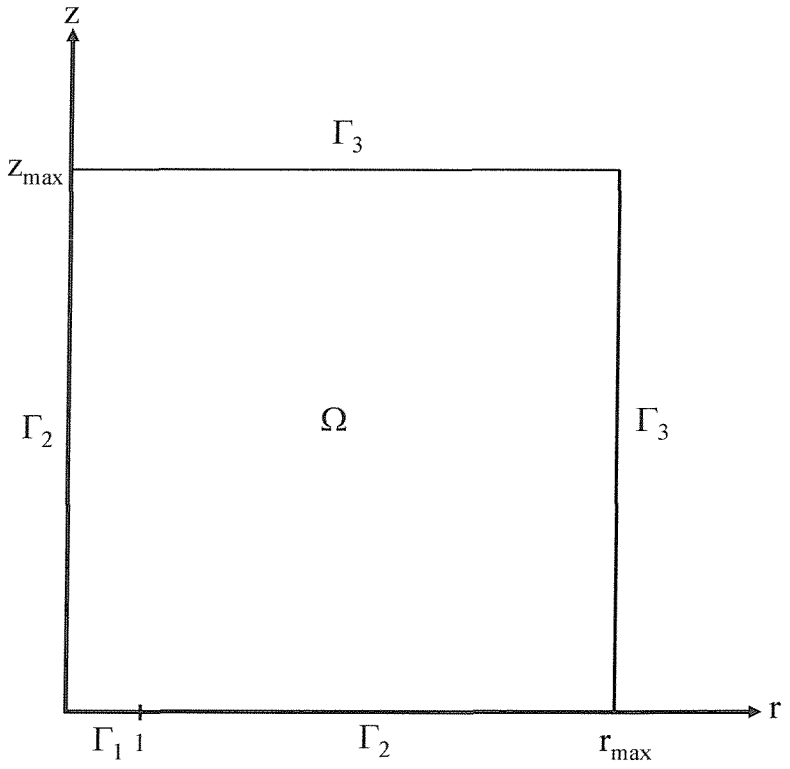

Figure 4.1: The microdisc simulation domain. For the reversible E mechanism, $u = 0$ on Γ_1 and $\frac{\partial u}{\partial n} = 0$ on Γ_2. If r_{max} and z_{max} were infinite we would have $u = 1$ on Γ_3, but for finite domain extents we can, for testing purposes, impose the exact analytical values on Γ_3 as a Dirichlet condition. For the EC' mechanism we impose $u = 1$ on Γ_1, $\frac{\partial u}{\partial n} = 0$ on Γ_2, and $u = 0$ on Γ_3.

4.1.2 Analytical Solution

The analytical solution to the problem described above has been given by various authors, and is derived in Appendix A on page 253. It is

$$u = \begin{cases} \frac{2}{\pi}\cos^{-1}\left(\frac{2}{\sqrt{z^2+(1+r)^2}+\sqrt{z^2+(1-r)^2}}\right) & z > 0\,, \\ 0 & 0 \le r \le 1,\ z = 0\,, \\ \frac{2}{\pi}\cos^{-1}(1/r) & r > 1,\ z = 0\,. \end{cases} \tag{4.14}$$

We thus impose this on Γ_3 in Figure 4.1 on the page before.

The current cannot be directly determined from this definition, as described in Appendix A. The familiar formula, however, has already been given in § 1.4 on page 38, and using the dimensionless quantities described there it is

$$i_{norm} = 1\,. \tag{4.15}$$

Since we know the exact current we can in this case calculate the error of our simulation relative to the exact value, not an estimate. As this value is unity, this is the same as the absolute error here. This simplifies things slightly for this test case. After this section we turn to the relative error estimate, described in § 3.2.3 on page 109, for more realistic simulations.

4.1.3 Simulation

The governing equation and microdisc domain of Figure 4.1, along with the imposed solution on Γ_3, are readily specified to the simulation program—the input file for the effectively infinite domain case is given in Appendix E. All of the results in this section, except of course those used for examining the effect of domain size, were derived using the small domain with analytical bulk concentration values.

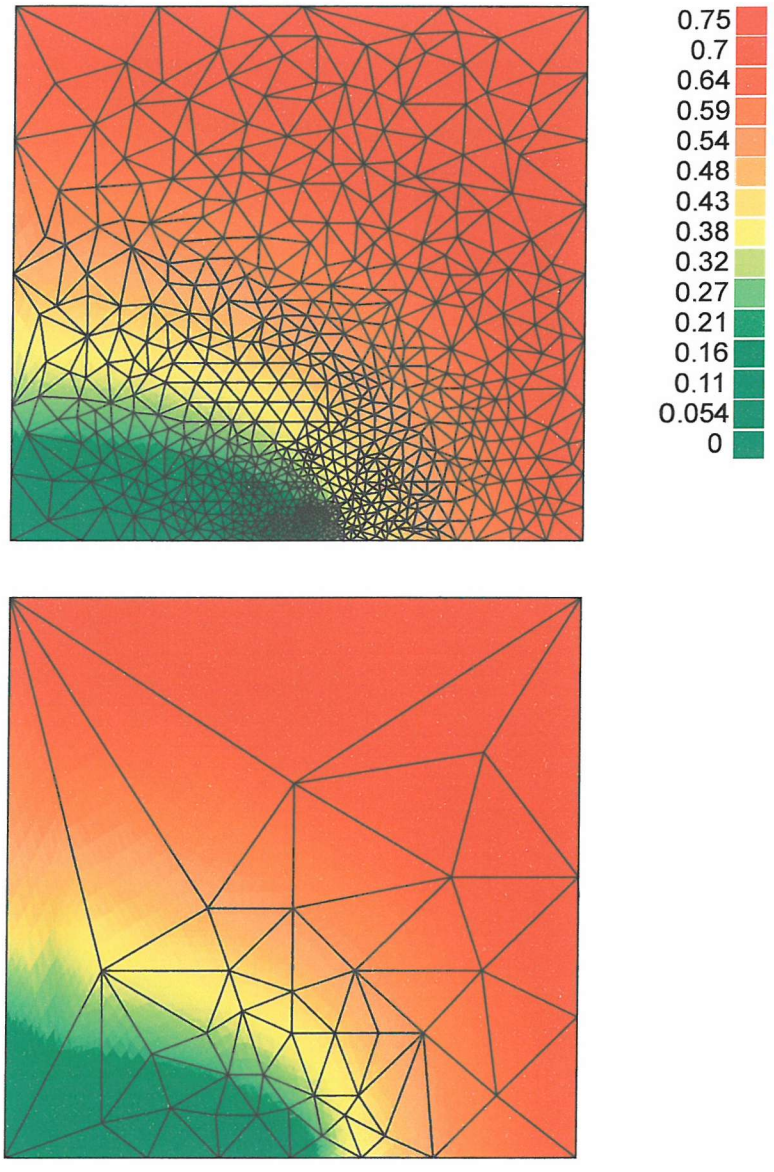

Figure 4.2: The normalised concentration field and mesh produced when simulating the E mechanism inlaid microdisc problem with a 5% current error tolerance. The first mesh is with linear elements, the second quadratic.

Concentration Field and Meshes

The final meshes for 5% accuracy, and the field interpolated on them, are shown in Figure 4.2 on the preceding page. The results look reasonable, given our physical understanding of the problem, and comparison of the concentration field with the analytical values confirms this. The meshes generated *en route* to these are reproduced in sequence in Figures 4.3 and 4.4: the progression to a mesh allowing accurate representation of the concentration field around the singularity at $r = 1, z = 0$ is evident. Note that only five mesh iterations are required with quadratic elements, as opposed to nine with linear ones.

A comparison of the final meshes produced for various error criteria with linear and quadratic elements is shown in Figure 4.5 on page 144. The density of the adaptive mesh produced evidently depends heavily on the element order used, and we expect on this basis that quadratic elements will be significantly more efficient. This is confirmed by the simulation times, which make them on average approximately a factor of four faster with our implementation and computer hardware (as noted elsewhere, all simulations were performed on a mobile Pentium III running at 850 MHz) for 5% error tolerance.

While the mesh has been refined near the singularity, overall the number of nodes is not particularly high. For the same problem Harriman *et al.*, using their error estimate, achieved an estimated 5% accuracy with 396 nodes [34]. Our results are shown in Table 4.1 on page 143. Here "nodes" refers to assembly nodes [16], at the corner of elements. "DOF" (degrees of freedom) includes all nodes (i.e. quadratic element edge nodes as well) and, as the number of unknowns in the global linear system, is the better guide to computer time.[1] Since no figures for DOF are given by Harriman *et al.*,

[1]Note, though, that system matrices for quadratic element nodes apparently tend to have a larger condition number [16], which could increase the number of solver iterations—see Appendix D on page 271

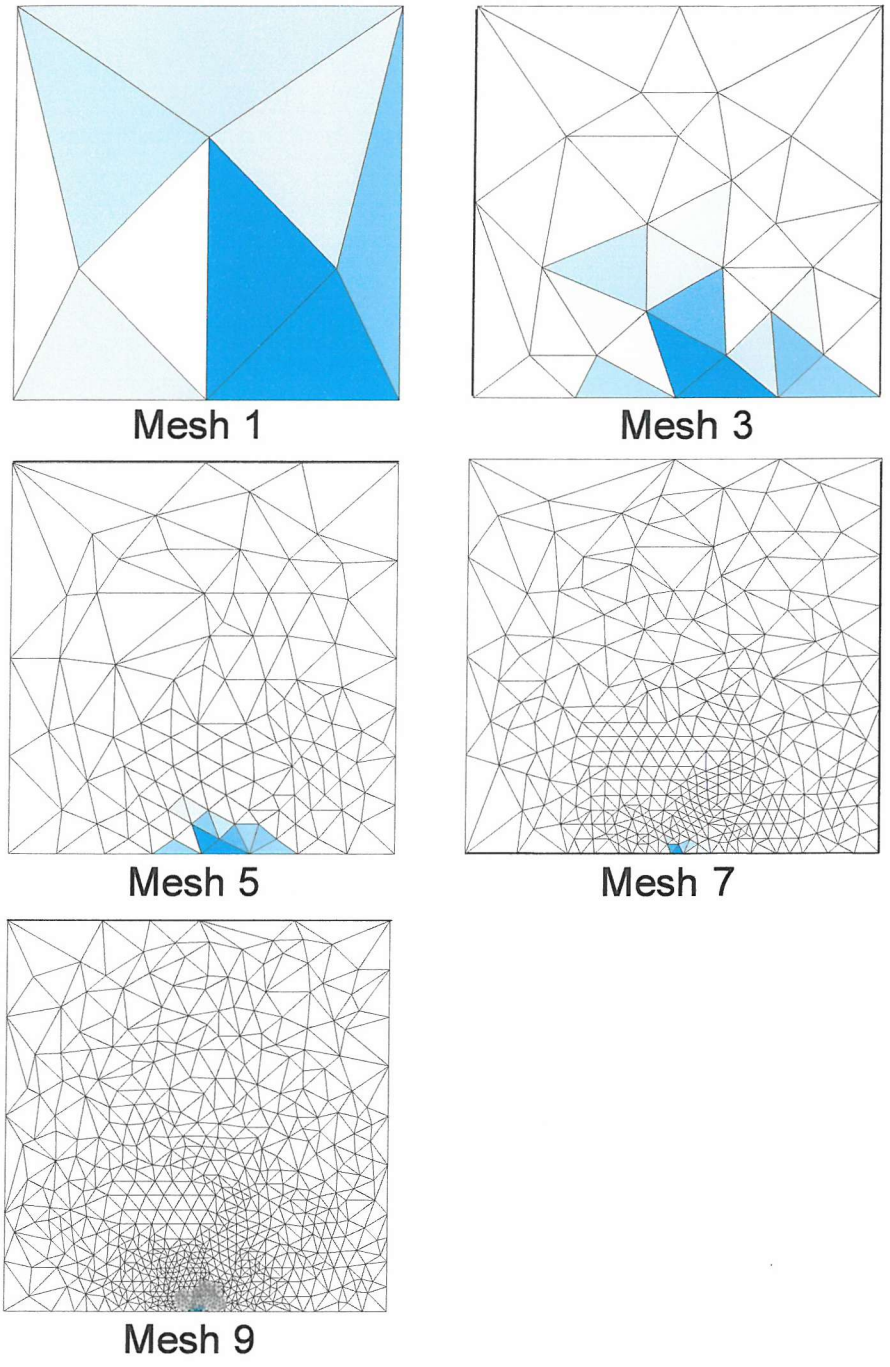

Figure 4.3: Every other linear element mesh produced in the sequence of nine required to refine to an estimated 5% error in the current to the inlaid microdisc. Element colours are proportional to the error indicator, with darker cyan higher. The scale is different for each mesh.

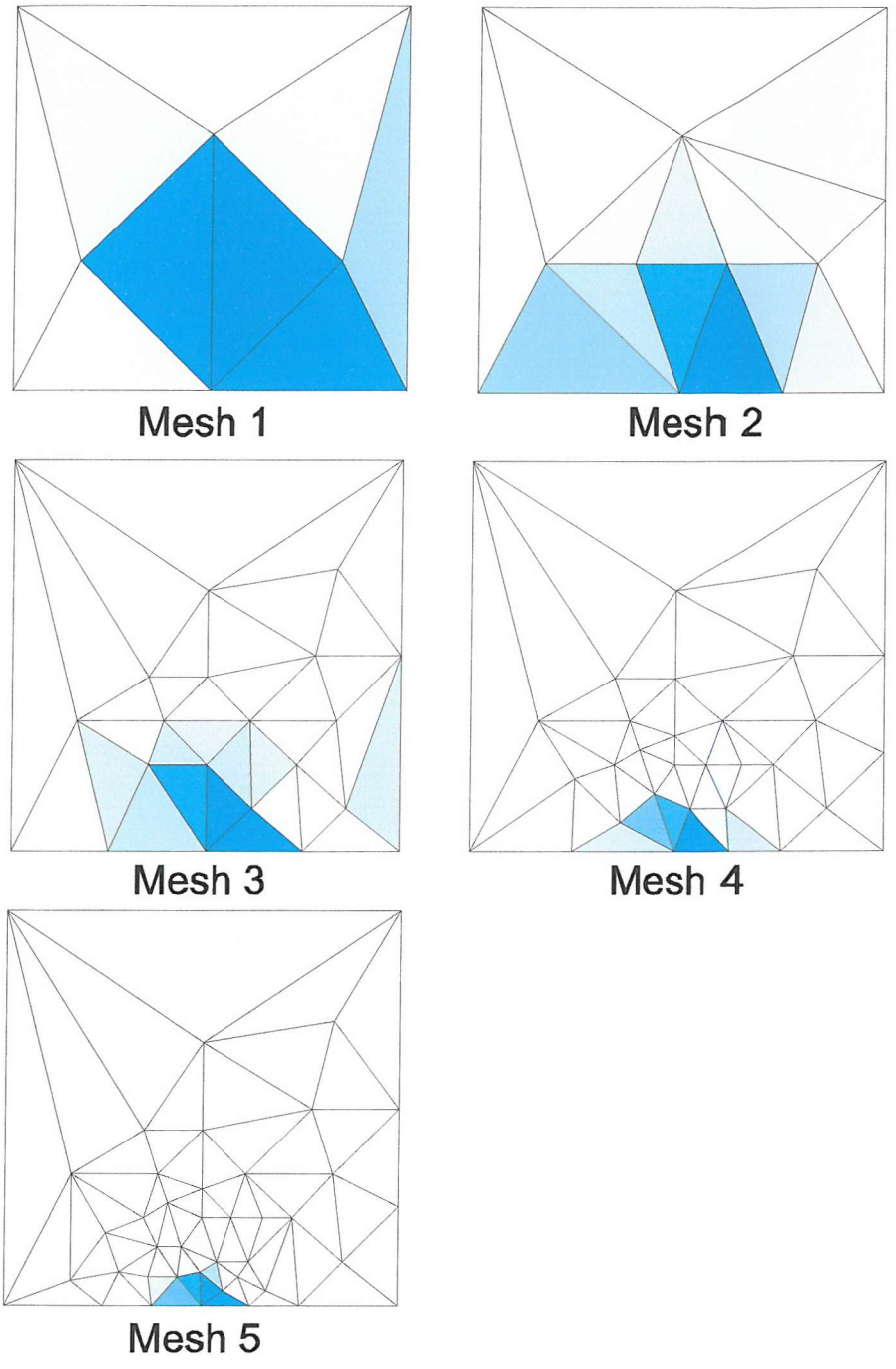

Figure 4.4: The five quadratic element meshes produced in refining to an estimated 5% error in the current. The first mesh is of course the same as that for linear elements, but the error distribution is different, and we see in the second mesh that the first refinement iteration is more targeted towards the boundary singularity.

Element Order	Tolerance ($\bar{\eta}$)	Nodes	DOF	CPU time
linear	10%	177	177	0.15 s
linear	5%	720	720	0.59 s
linear	2%	4707	4707	6.22 s
quadratic	10%	17	58	0.05 s
quadratic	5%	41	150	0.13 s
quadratic	2%	95	352	0.37 s

Table 4.1: The computational cost of linear and quadratic elements for the microdisc simulation for various current error tolerances. The node and DOF counts are for the final mesh. As error tolerances become finer, the number of DOF escalates rapidly, but with quadratic elements the effect is much less pronounced.

we adopt their convention for purposes of comparison.

Since Harriman *et al.* only use quadratic elements for the dual problem, and in a sense their simulation is linear element based, an exact comparison is not possible, because we would generally expect better performance (for a given node count) for quadratic elements, and a poorer one for linear elements (we use either linear or quadratic elements for both the primal and influence function problems). In all performance comparisons this arguable unfairness should be born in mind, but it does not seem unreasonable to compare the schemes on a practical computational basis: for most cases in this work we solve a single quadratic element problem, whereas Harriman *et al.* solve both a linear and a quadratic one.

Despite the provisions of the mesh refinement algorithm, some elements are not as perfect in their shape as might be hoped in the linear element case. The solution would appear to be implementation of the Ruppert algorithm mentioned in the previous chapter. Crucially, though, the mesh is good enough to converge to the desired result. The effect is also less pronounced with quadratic elements, which fortunately are preferable for the efficiency reasons just stated.

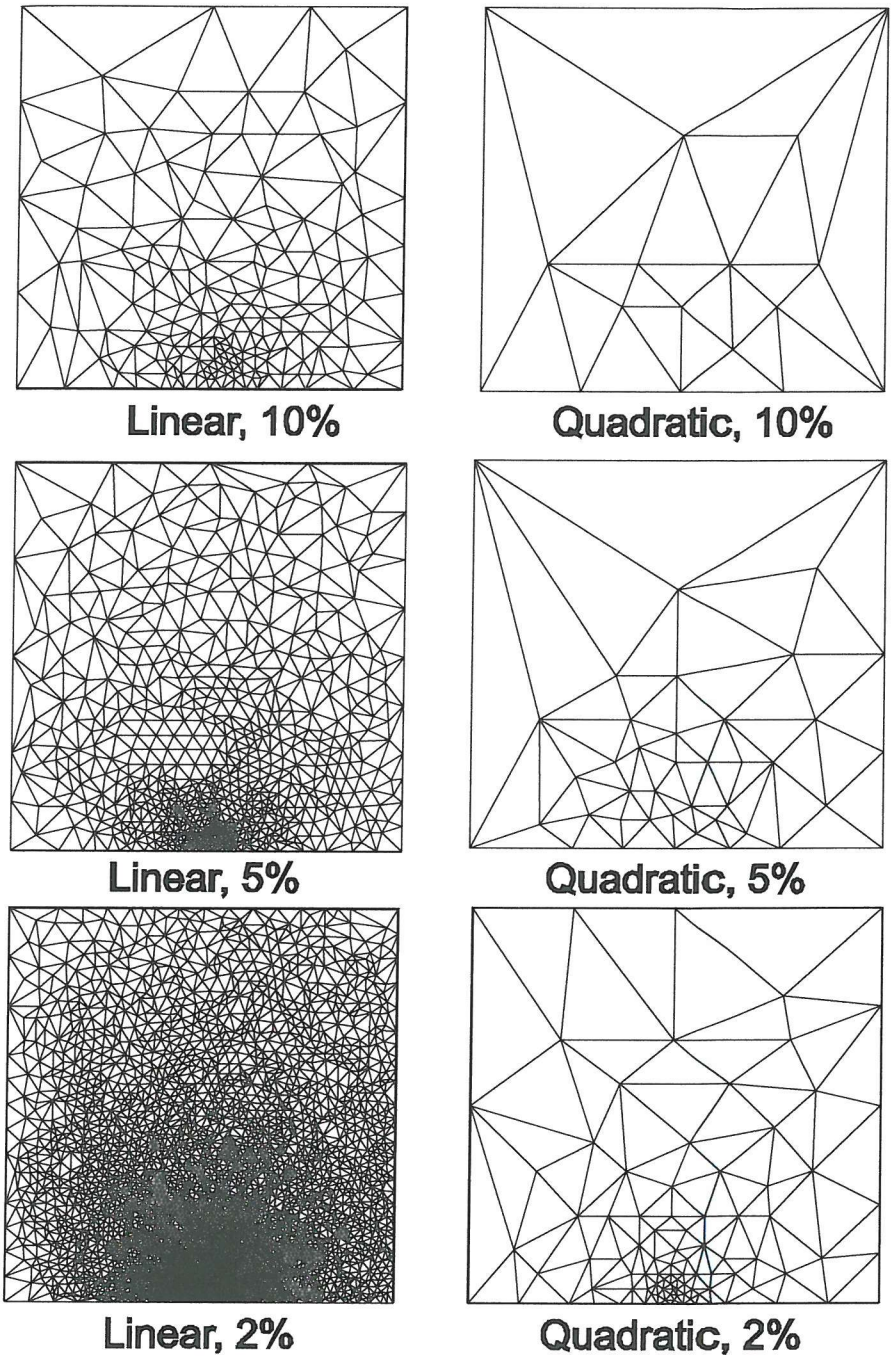

Figure 4.5: The meshes generated for prescribed current error of 10%, 5% and 2% respectively for quadratic and linear elements.

Current Calculation

While examination of the simulated concentration field can give an indication of the accuracy the adaptive finite element algorithm, the only logical criterion on which to evaluate it is the calculated current. This is, after all, the *raison d'être* of this work. We do, however, have several means of calculating the current, and it is important to check their relative merits.

In §3.2.2 we have an alternative current formula from the definition in §1.4, using a domain integral instead of one along the boundary. We therefore have two formulae for the current in terms of the field gradient. We also have two different versions of the gradient ∇u—the directly extracted one, and the smoothed one. Thus there are a total of four comparable formulae that we shall label b, $\hat{b}$, d and $\hat{d}$:

$$b = -\frac{\pi}{2}\int_0^1 \frac{\partial u}{\partial n} r dr \ , \qquad \hat{b} = -\frac{\pi}{2}\int_0^1 \frac{\partial \hat{u}}{\partial n} r dr \ , \tag{4.16}$$

$$d = -\frac{\pi}{2}\int_\Omega \nabla v \cdot \nabla u r dr dz \ , \ \hat{d} = -\frac{\pi}{2}\int_\Omega \nabla \hat{v} \cdot \nabla \hat{u} r dr dz \ . \tag{4.17}$$

Although the error adaptivity was guided by the domain integrals, we can still evaluate the boundary integrals, and it is important to see the difference in accuracy. As was argued in Chapter 3, we expect the best result to come from the domain integral with a smoothed gradient, $\hat{d}$, since the domain integral is not so dependent on local errors at the boundary, and the smoothed gradient is assumed to be closer to the exact solution. It was also predicted that $\hat{d}$ would be smaller than d. These predictions are confirmed in the results for various prescribed errors, given in Tables 4.2 and 4.3, for linear and quadratic elements respectively.

Clearly the domain integral is considerably more accurate, as found by Harriman *et al.*, and this is true irrespective of gradient smoothing. We discard from further consideration the boundary integrals, as not only are they inaccurate, but the error estimate has little connection with their errors. Although SPR does improve their accuracy, it does not do enough to make them competitive with the domain integrals. While some form of interpo-

$\bar{\eta}$	η	b	$\hat{b}$	d	$\hat{d}$
10%	0.099287	0.806364	0.864424	1.03028	1.00684
8%	0.075223	0.840669	0.875394	1.01878	1.00305
6%	0.049179	0.888538	0.922629	1.01167	1.01056
4%	0.033071	0.896765	0.932499	1.00628	1.00437
2%	0.019821	0.919240	0.941902	1.00346	1.00104

Table 4.2: The output for the four current expressions of (4.16) and (4.17) with various prescribed accuracies, $\bar{\eta}$, using linear elements. (The analytical answer is of course exactly 1.) The domain integrals are more accurate, although the smoothed gradient helps to an extent in either case. Note that the estimated error of d, η, is not exactly equal to the prescribed error, $\bar{\eta}$, since the iterative refinement process proceeds in discrete steps.

$\bar{\eta}$	η	b	$\hat{b}$	d	$\hat{d}$
10%	0.094150	0.722469	0.762309	1.06051	0.98481
8%	0.053179	0.904447	0.910635	1.02619	1.01413
6%	0.046929	0.809215	0.862562	1.02421	0.99738
4%	0.035228	0.866900	0.884601	1.01481	0.99274
2%	0.015237	0.902981	0.927150	1.00561	0.99774

Table 4.3: A comparison of current expression accuracies with quadratic elements. See the caption of Table 4.2. These results emphasise that only the quantity explicitly controlled by the error bound, d, is expected to become more accurate with a more stringent error tolerance. The boundary integral expressions are clearly not accurate, and are worse than with linear elements. $\hat{d}$ is more accurate than d, but unreliably so.

lated smoothed field near the boundary would doubtless yield more accurate derivative estimates, it seems improbable that it could match d or $\hat{d}$, and it would still bring with it the attendant difficulties noted in Chapter 3.

The question of d versus $\hat{d}$ is less clear-cut, but we can still make a firm decision. While the error *estimate* applies to d, it is also a conservative *bound* on the error in $\hat{d}$, as we would expect; the smoothed field current is in every case more accurate. We therefore adopt the smoothed gradient current for future simulation results, but keeping in mind the fact that it converges to the true result less reliably as the error tolerance is decreased. We also note that where we find disagreement with other estimates, we can have some confidence in d as an upper bound on the current, unlike $\hat{d}$, which can clearly underestimate the true value.

Error Estimator Effectivity

The error estimate is, as expected, a conservative one in the case of the domain integrals. In Tables 4.4 and 4.5 we show the factor of overestimation, termed the *effectivity index* [16], for both the smoothed and unsmoothed gradients, for linear and quadratic elements respectively.

The difference in effectivity index between smoothed and unsmoothed current expressions clearly varies to a large degree. It must be kept in mind, however, that the point of comparison must be for d, as this is the quantity for which the error estimator was developed. Any additional accuracy of the smoothed gradient is free (computationally speaking, since gradient smoothing must be performed for the error estimation itself anyway), but is also unquantifiable where we do not know the problem's answer before we start. Our case is that, while the error estimator applies only to d, if the Taylor series-type argument that motivates the error estimation theory holds true then we expect the error of the higher order current expression $\hat{d}$ to be lower than that of d.

For this example the effectivity index is consistently higher than that of

$\bar{\eta}$	η	$\eta/\lvert d-1\rvert$	$\eta/\lvert \hat{d}-1\rvert$
10%	0.099287	3.28	14.5
8%	0.075223	4.01	24.7
6%	0.049179	4.21	4.66
4%	0.033071	5.27	7.57
2%	0.019821	5.73	19.1

Table 4.4: Quantification of the error estimator's conservativeness for linear elements. The two right-most columns show the degree to which the error estimator over- or underestimates the true current error for both domain integral current expressions. These are the effectivity indices. A value greater than unity indicates an overestimate, a value under an underestimate. A value of one signifies unattainable perfection.

Harriman *et al.*'s error estimator for linear elements: their estimator ranged between 3.07 and 3.81.[2] For quadratic elements the estimator's effectivity index is consistently lower to about the same degree. Intuitively this makes sense, as we are effectively comparing linear-linear and quadratic-quadratic schemes with a linear-quadratic one.

Of course, we can only speak with definiteness in relation to the inlaid E mechanism microdisc, for which we know the analytical solution. We cannot be certain that properties discerned in this case will carry over to other, more interesting, simulations, where the answer is not known *a priori*. We must make that assumption, however, if testing with known solutions is to have any meaning.

Efficiency

Referring to Table 4.1 on page 143, the simulation times are encouraging. The infinite domain microdisc problem is solved to 5% accuracy using quadratic

[2]Note that the figures given in [34] are incorrect.

$\bar{\eta}$	η	$\eta/\lvert d-1\rvert$	$\eta/\lvert\hat{d}-1\rvert$
10%	0.094150	1.56	6.20
8%	0.053179	2.03	3.76
6%	0.046929	1.94	17.9
4%	0.035228	2.38	4.85
2%	0.015237	2.72	6.74

Table 4.5: The conservativeness of the error estimator for quadratic elements. The error estimator is, generally speaking, more accurate for these than the linear kind, perhaps due to ultraconvergence effects. In both cases, as expected, the effectivity index is lower for the unsmoothed gradient current.

elements in 0.13 s. Harriman *et al.* quote a figure of 1.6 s [34]. Precise comparison of algorithmic efficiency is impossible, not least because of the different hardware platforms, but scaling according to clock speed (a not entirely meaningful procedure), this would translate to 0.6 s on our system.

Possibly the Sparc processor used in the other simulation performs fewer operations on average per clock cycle, but we might conservatively claim a factor of four increase in speed. While our hardware platform is superior, the relative speeds of the linear solvers are not clear. Further, we do, as described in Chapter 2, utilise an entirely general expression parser for calculation of functionals such as element errors and the current. While this must ultimately become of marginal importance as the problem size grows and the linear solver dominates, for this case it is likely that our program is handicapped speed-wise, and if specialised to the pre-programmed expressions used by Harriman *et al.*'s system it could claim a larger speed advantage.

Solving the more realistic finite domain does increase computation times (see below), but since the simulation is relatively insensitive to domain size we have no reason to expect that our simulation scales more poorly than competing methods, and unless truly infinite domains are implemented (as

they are, for instance, with boundary element) we can remain relatively confident that our method is competitive. Note that because $u = 1$ is imposed on the bulk boundary of the finite domain, the problem is what we have termed "inverse dual": the dual is $1 - u$. This means that half as much work needs to be done for solving each mesh, and this goes some way to counterbalancing the effect of the larger domain. This translates to many other, more realistic, problems.

Effect of Domain Size

Regardless of all the error control theory that we have presented, when we turn to infinite domains a modelling error of unknown magnitude remains. Quite apart from inexact numerical integration, and the necessarily estimative nature of our error measures, the approximation of infinite domains with finite elements introduces another form of inaccuracy. It is problem-specific, and unfortunately requires new analysis for reasonable confidence with a new domain or mechanism. Possible approaches to eliminate, or at least reduce, the problem are discussed in Chapter 6. For now, we deal with it as almost all have done: by using large finite domains.

With the reversible E mechanism at an inlaid microdisc, we of course have the analytical current with which to compare. Adopting 5% accuracy then, using quadratic elements, and evaluating both domain integral measures of the current, we observe the tendency towards the true value with varying values of r_{max} and z_{max}. Although the r and z axes are not equivalent in the PDE, as they would be with Cartesian simulations, we see no particular reason to set these two differently. Thus we vary them together from two to two hundred. The numerical results are given in Table 4.6, and plotted in Figure 4.6. Note that in this case, as in the previous one, we artificially calculate the error relative to the known analytical value, rather than to the simulated current. The difference is not great for all but the smallest domain size, which is any case entirely unrealistic.

$r_{max} = z_{max}$	d	$\hat{d}$	n	CPU time
2	1.44052	1.41276	80	0.17 s
5	1.15782	1.13218	55	0.11 s
10	1.10040	1.07061	70	0.16 s
20	1.04970	1.03227	91	0.24 s
50	1.03088	1.00467	128	0.36 s
100	1.02413	0.99544	149	0.48 s
200	1.02971	0.99582	175	0.57 s

Table 4.6: The effect of domain size on the calculated current and computation time, with a 5% prescribed current accuracy, using quadratic elements. Clearly we need a domain with dimensions of at least twenty to get 5% accuracy, although it might be prudent to use a domain larger than this. The number of nodes n generated is shown, illustrating the benefits of adaptive refinement.

Evidently a reasonable domain size is required for error adaptivity to have any purpose at all—solving the wrong problem to 5% accuracy is of no use. Fortunately, because most of the refinement occurs close to the electrode, larger domains do not add too many nodes (although the simulation times do increase noticeably). On the other hand, domains with very large dimensions relative to the features of interest (i.e. the electrodes), stress the limits of the meshing algorithm. We also find that the benefits of doubling the domain size rapidly tail off as we get beyond about twenty units, unsurprisingly, since we expect an asymptotic approach to the true value given the functional form of the field.

Domain size effects are one of the few things that become less of a problem with transient simulations. With steady state, we are assuming an infinite time has elapsed, and consequently that boundary condition effects have propagated all over the domain. This is in contrast to a transient simulation, where, despite theoretically infinite propagation speeds, pronounced bound-

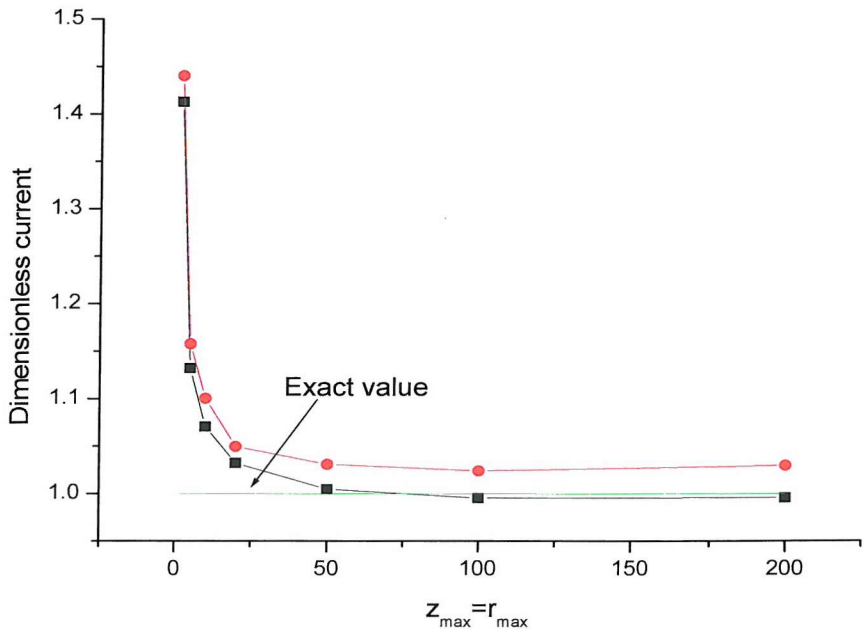

Figure 4.6: The currents d (shown in red) and $\hat{d}$ (in black) of Table 4.6 on the preceding page plotted as a function of domain size. It is seen that the slight undershoot of $\hat{d}$ does not occur with d; indeed d increases slightly between the last two points plotted. Of course, the magnitudes of these fluctuations are far smaller than the prescribed error bound of 5%.

ary effects generally occur only when a notional diffusion front reaches those boundaries.

4.1.4 Conclusion

We have demonstrated the basic efficacy of the current error controlled adaptive approach. The results suggest that the error control scheme is effective enough to make domain truncation, rather than finite element discretisation itself, the more unpredictable source of error. It seems plausible that the domain sizes determined here to be adequate would also be with recessed

microdiscs too and SECM tips, but other mechanisms might, of course, have rather different characteristics.

From the tests conducted, we conclude that quadratic elements are by far the more efficient, and we shall therefore use them in all simulations henceforth. We also adopt the current calculation formula $\hat{d}$ (or natural variants thereof) for all results in the rest of this chapter and the next. As has already been stated, we use SPR gradient smoothing everywhere. While other simulations were conducted with linear elements, and with other variation of parameters, we do not present the results in this work, as nowhere were results observed significantly different in nature from those here.

4.2 E Mechanism Recessed Microdisc

Recessed microdiscs are relatively important on account of photolithographic manufacturing techniques [141, 142], and for a number of reasons we apply the simulation program to this problem. There exist approximate expressions for testing the results, making them a usable benchmark and an additional means of validation. Since some of these expressions derive from finite element simulations, comparison of computational aspects can also be made. Finally, the recessed microdisc is a first opportunity to check that relative error estimation is working correctly, since the current magnitudes vary over the range of recess depths.

4.2.1 The Problem

As we still have an E mechanism in a rotationally symmetric domain, much remains the same with the recessed microdisc. We again solve (4.10), but in a modified domain, shown in Figure 4.7 on the next page. As with the finite domain version of inlaid microdisc, we impose the bulk normalised concentration of unity on Γ_3, away from the electrode; thus the problem is also inverse dual. In fact we alter but a few lines of the input file for that

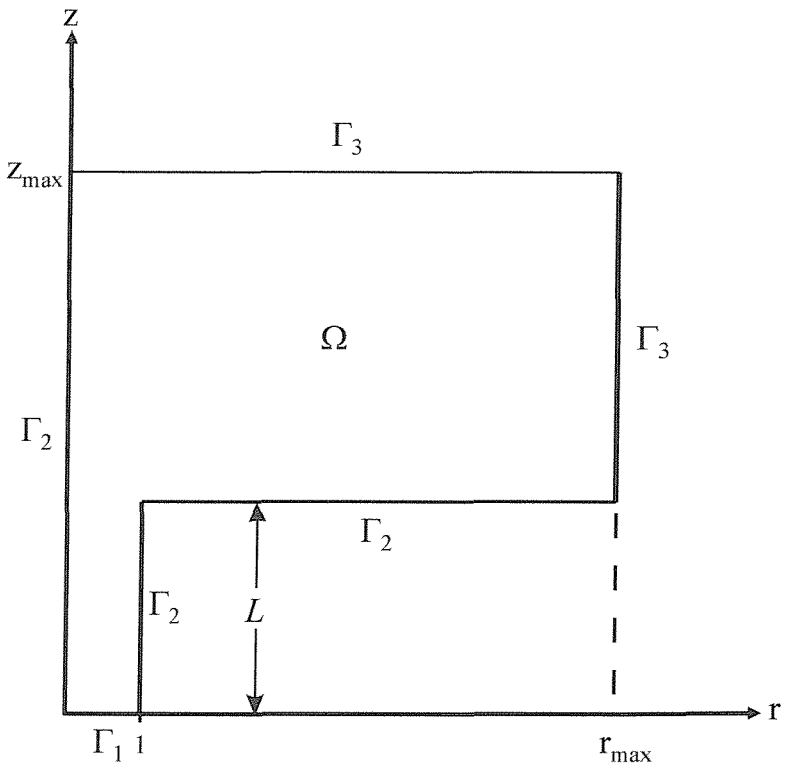

Figure 4.7: The recessed microdisc simulation domain. For the E mechanism $u = 0$ on Γ_1, $\frac{\partial u}{\partial n} = 0$ on Γ_2 and the bulk boundary condition $u = 1$ holds on Γ_3. For the EC' mechanism the values of u on Γ_1 and Γ_3 are swapped, with $u = 1$ and $u = 0$ respectively.

case in order to simulate this one, as described in Appendix E.

As the recess depth L tends to zero, we expect the current to tend towards the inlaid microdisc one. As the recess depth increases, on the other hand, the disc becomes increasingly shielded from the bulk solution, and the current will decrease. The recessed microdisc exhibits a less pronounced singularity than the inlaid one, essentially because the boundary conditions at the electrode/insulator interface are less "contradictory", and we expect this to improve simulation times.

4.2.2 Approximate Solutions

A standard expression for the steady state current is that of Bond *et al.* [25]. This assumes a constant concentration across the mouth of the recess, which only becomes a reasonable approximation for relatively deep recesses, limiting the formula's applicability to these cases. It is, in our units,

$$i_{bond} = \frac{\pi}{4L + \pi} . \tag{4.18}$$

Other expressions derived from curve fitting to simulation results also exist, designed for use with the shallow recesses ill-served by (4.18). Both Ferrigno *et al.* [143] and Bartlett and Taylor [87] used finite element simulations to generate approximate expressions for the range $0 \leq L \leq 1$. These are, respectively,

$$i_{ferrigno} = \exp(-0.96L) \tag{4.19}$$

and

$$i_{bartlett} = \frac{1}{1 + A_1 L + A_2 L^2 + A_3 L^3 + A_4 L^4} , \tag{4.20}$$

where $A_1 = 1.6843$, $A_2 = -1.3237$, $A_3 = 1.7116$ and $A_4 = -0.7585$.

As described by Bartlett and Taylor, these last two expressions are contradictory for almost all relevant values of L, with their formula finding more agreement with (4.18) in spite of its inappropriateness for low values of L. A fresh simulation should add evidence in support of one or the other, assuming one of them is more correct.

4.2.3 Simulation

Although the difference between Equations (4.19) and (4.20) rises to above 10% (of the value (4.20)), for some values of L a fairly fine tolerance is needed to distinguish between them. We also desire to evaluate the results of Bartlett and Taylor's expression versus (4.18), and there the difference is consistently less pronounced. Consequently, in order to show clear separation, a prescribed relative error of 0.5% was used. For almost all values of L the expressions' disparities are several times greater than this.

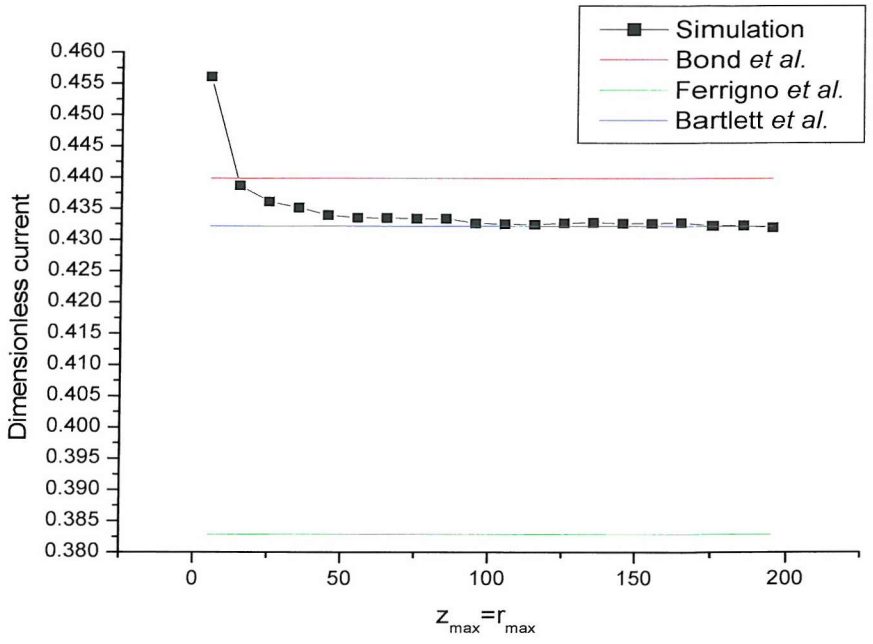

Figure 4.8: The effect of domain size on the simulated current to a microdisc of recess depth 1. The values of the expressions of Bond *et al.* (4.18), Ferrigno *et al.* (4.19) and Bartlett *et al.* (4.20) are also plotted.

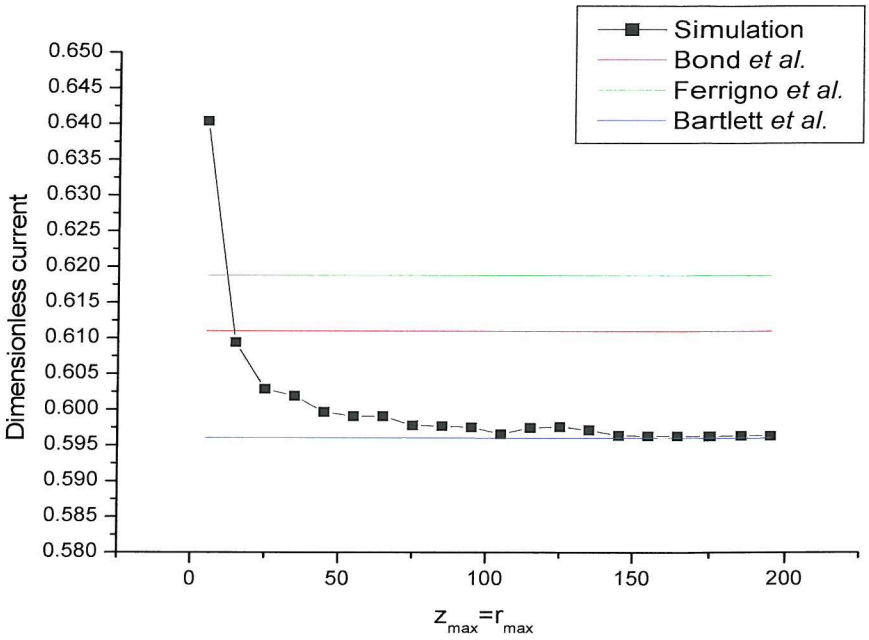

Figure 4.9: The effect of domain size on the simulated current at a recess depth of 0.5.

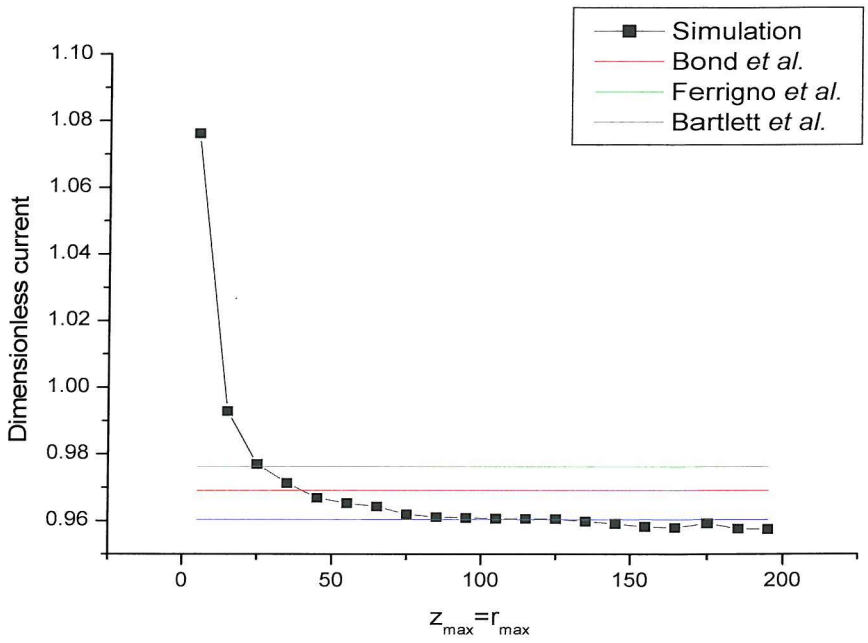

Figure 4.10: Simulated current versus domain size for a recess depth of 0.025. The shallower recess depth clearly increases the effect of domain size on the result because of a reduced shielding effect.

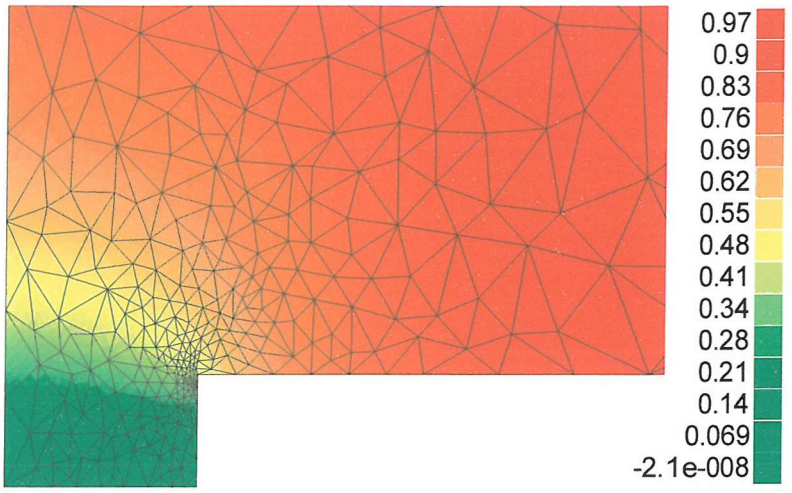

Figure 4.11: The mesh and concentration field around the electrode for a recessed microdisc with $L = 0.5$, $r_{max} = z_{max} = 200$, and a prescribed error of 0.5%. Note the concentration of nodes around the corner singularity at the recess mouth. 518 nodes and 2006 DOF were used in the final mesh.

In order to show convergence with increasing domain size we opt to simulate for three different values of L—0.025, 0.5 and 1.0—which are hoped to be representative, and to use a range of values of $r_{max} = z_{max}$. The results are shown in Figures 4.8 to 4.10. A typical concentration field and mesh combination is also shown in Figure 4.11. It is seen that in all cases the results are within 0.5% of the value of (4.20) once moderate domain sizes are reached. Using the largest domain size of 200, the results for a range of values of L are shown in Figure 4.12. The slowest of these simulations, with $L = 0.025$, took 3.6 s to run.

4.2.4 Conclusion

The results clearly lend weight to the conclusions of Bartlett and Taylor, considering that our finite element simulator is markedly different in construction from theirs, and yet produces results closely in agreement. In each

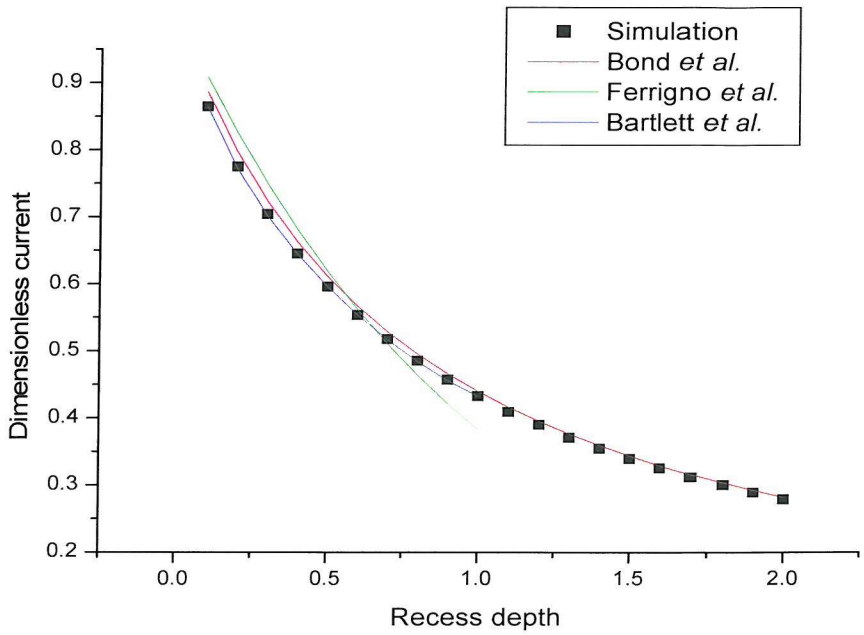

Figure 4.12: Current versus recess depth, using a domain size of 200 and an error tolerance of 0.5%. In the range $0 \leq L \leq 1$ the expression of Bartlett and Taylor (4.20) interpolates the simulation results to within 0.5%. Outside this range the expression of Bond *et al.* is within 2% of the simulated values. This curve took 63 s to produce.

case within the range $0 \leq L \leq 1$ our simulation tends towards their given value, and with a reasonable domain size the difference is less than that mandated by our error bound. Like them, we conclude that the expression of Ferrigno *et al.* is less accurate than that of Bond *et al.* for the bulk of values of L for which it was designed. Since our results do not diverge from those of Bartlett and Taylor for any value of L below 1, we do not present them here; instead (4.20) would appear adequate for the task of predicting currents with shallow recesses. On a side note, results not given here indicate that Ferrigno *et al.* [143] also overestimate the current to raised microdisc electrodes, presumably owing to the same problems as with the recessed

microdisc—ideally a further investigation of this would be conducted.

It is perhaps interesting to briefly compare finite element methodology with the two sets of authors. Ferrigno *et al.* [143] used a commercial finite element package and some form of mesh density specified *a priori*. The number of nodes is not given, and nor are simulation times. Irrespective of efficiency, the authors had difficulty with convergence, and as we have seen, produced an answer actually worse than the approximation they were trying to better. This must surely be an argument for current error-controlled adaptive finite element.

Bartlett and Taylor [87] used a custom-written finite element program, again with a preordained mesh density function. This clearly represents a significant investment of programming effort, given that the program would require modification for a new geometry, and a new grid density function would have to be found and validated. In our case, apart from checking the effect of domain size, everything is automatic, and the simulation program input file might reasonably be expected to take ten minutes to write.

We do not consider performance of the various simulations, as the programs and hardware platforms are too different to make comparison meaningful.

In conclusion, the recessed E mechanism microdisc is a case where our approach works well, and admits of useful information. However, it should be born in mind that the two references with which we compare results also dealt with the transient case, for which our program so far lacks the capabilities.

4.3 *E* Mechanism Hemisphere

The hemispherical electrode presents, of course, one of the more tractable theoretical diffusion problems. Since a solitary hemispherical electrode has enough symmetry to reduce the problem to one spatial dimension, analytical solutions exist for numerous cases, particularly in the steady state. Clearly

it also does not require more than one spatial dimension for simulation, but our interest is in evaluating our program's ability to handle hemispherical microelectrode geometries because they appear in other less symmetric contexts. This is primarily a test of the effect of curve approximation by a finite number of straight line segments.

A second attraction of modelling the well-known hemispherical case is that it offers the chance of testing, in an electrochemically realistic case, our program's ability to handle heterogeneous kinetics. If we consider the simplest scenario of a totally irreversible electrode reaction, only one field need be simulated (aside from any influence function required). Thus a comparison can easily be made with the simple exact analytical expression given below.

4.3.1 The Problem

The problem solved is essentially the same as that described in §4.1.1 except for the change in geometry, which is shown in Figure 4.13 (with the electrode radius normalised to one). The procedure of §4.1.1 can be adopted to compress the range of reversible cases into one normalised simulation. We first consider this scenario, before turning to the totally irreversible one where heterogeneous kinetics come into play.

For the totally irreversible case a concentration normalisation of the form

$$u = c_A/c_A^* \tag{4.21}$$

is adopted, and the electrode boundary condition of §1.1.4 becomes

$$\frac{\partial u}{\partial n} = -Ku\ , \tag{4.22}$$

where $K = \frac{k_f a}{D_A}$ (a is the electrode radius). Therefore simulations are distinguished by the boundary condition on Γ_1; for both $\frac{\partial u}{\partial n} = 0$ on Γ_2 and $u \to 1$ towards the bulk. In the limit as $K \to \infty$ we expect the irreversible case to tend towards the reversible one.

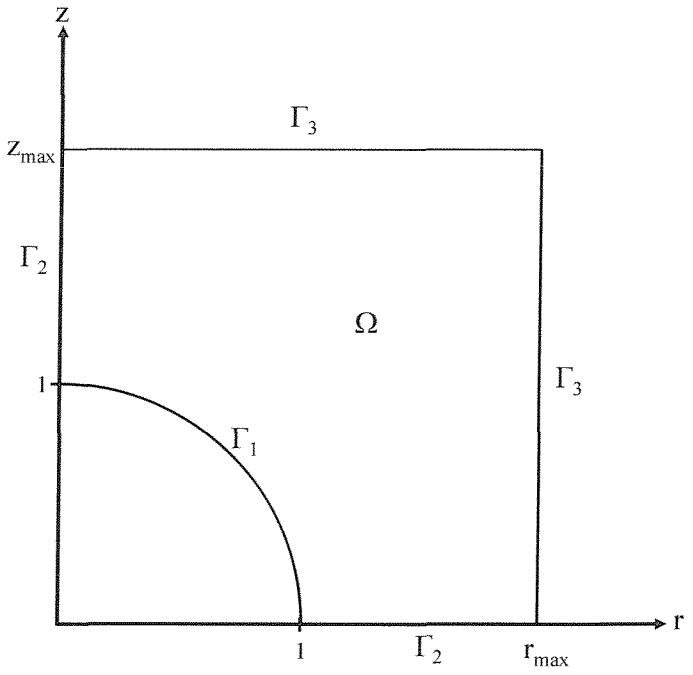

Figure 4.13: The hemispherical electrode simulation domain. On Γ_1 we impose $u = 0$ for the reversible case, and $\frac{\partial u}{\partial n} = -Ku$ for the kinetically controlled one. The two symmetry axes labelled Γ_2 have the usual zero flux condition. On the bulk boundary, Γ_3, we impose the exact analytical value given by (4.23)

There is no singularity with the hemisphere geometry because it meets the insulating surround at right-angles, and the "no flux" boundary condition there is compatible with the solution of the full sphere problem: part of the symmetric reduction to one dimension assumes no angular dependence, and the insulating surround is consonant with this.

4.3.2 Analytical Solution

In spherical coordinates the problem reduces to a simple ordinary differential equation. With the bulk boundary condition given above the solution is, in cylindrical coordinates,

$$u = 1 + \frac{c_1}{\sqrt{r^2 + z^2}} \tag{4.23}$$

(the expression is obviously simpler in spherical coordinates). The remaining constant c_1 is determined by the electrode boundary condition.

Diffusion Control

If there is a concentration of zero on the surface of a unit radius hemisphere the concentration field is given by

$$u = 1 - \frac{1}{\sqrt{r^2 + z^2}} \, . \tag{4.24}$$

We opt for a definition of the dimensionless current that gives the analytical answer unity:

$$i_{norm} = \int_{\Gamma_1} \frac{\partial u}{\partial n} r d\Gamma = 1 \tag{4.25}$$

(i.e. apart from dividing by the usual physical constants, F, D_A, etc., we also divide by the surface area of the hemisphere).

Kinetic Control

Where we have the electrode boundary condition

$$\frac{\partial u}{\partial n} = -Ku \, , \tag{4.26}$$

(4.24) changes to

$$u = 1 - \frac{K}{(1+K)\sqrt{r^2+z^2}} \,. \tag{4.27}$$

The current is then

$$i_{norm} = \frac{K}{1+K} \,, \tag{4.28}$$

which clearly tends to one as $K \to \infty$; at the same time (4.27) tends towards the same value as (4.24).

4.3.3 Simulation

In the reversible case the current as a function of subdivision is examined. This gives an idea of the level of circle subdivision required to push geometric error below that of domain discretisation. The simulation program is then tested over a range of values of K.

Diffusion Control

In order to isolate the effect of the circle approximation, we impose the analytical value from (4.24) on the bulk boundary, and set $r_{max} = z_{max} = 2$. We subdivide the circle segment shown in Figure 4.13 uniformly into various numbers of straight line segments, and impose a relatively strict 0.1% error in order to minimise domain discretisation error effects. It is perhaps worth noting that the first of the subdivision levels—one line segment—actually corresponds to a conical electrode.

The results for various subdivision levels are shown in Table 4.3.3 on the following page. The level of subdivision makes little difference to the simulation time, but the gains of very fine subdivisions clearly run out fairly quickly; for 1% error eight subdivisions are sufficient in this case. This is fortunate, because fine subdivisions have a tendency, in combination with the current meshing algorithm, to produce meshes with highly elongated elements. On the subject of meshing, we note that in the example mesh shown in Figure 4.14 the low error tolerance of 0.1% throws up generally

Subdivisions	i_{norm}	Error	DOF
1	0.716139	0.283861	2568
2	0.916071	0.0839274	2662
3	0.960800	0.0392000	2388
4	0.977478	0.0224795	2596
6	0.989796	0.0102042	2224
8	0.994234	0.00576604	2412
10	0.996305	0.00369527	2585
20	0.999065	0.000934842	2350

Table 4.7: The currents arising from, and simulation degrees of freedom required to simulate, a hemispherical electrode with varying levels of geometric subdivision, all with a prescribed tolerance of 0.1%. 1% accuracy is achieved with eight subdivisions, but in order to neglect geometric approximation error for a 0.1% error target we need closer to twenty.

poorly shaped elements. With more realistic tolerances this problem is less pronounced, but clearly the meshing algorithm is not ideal.

Kinetic Control

In order to have a degree of confidence that the circle discretisation is not the primary cause of error, we opt for ten subdivisions for the kinetics simulations. We then run a series of 1% tolerance simulations with different values of the heterogeneous rate constant, K, again with the analytical bulk concentration imposed. The results are shown in Figure 4.15. In all cases the error is less than the 1%. The simulation therefore works in this case over seven orders of magnitude (we have not tested beyond these). Outside this region other, more suitable, approximations exist.

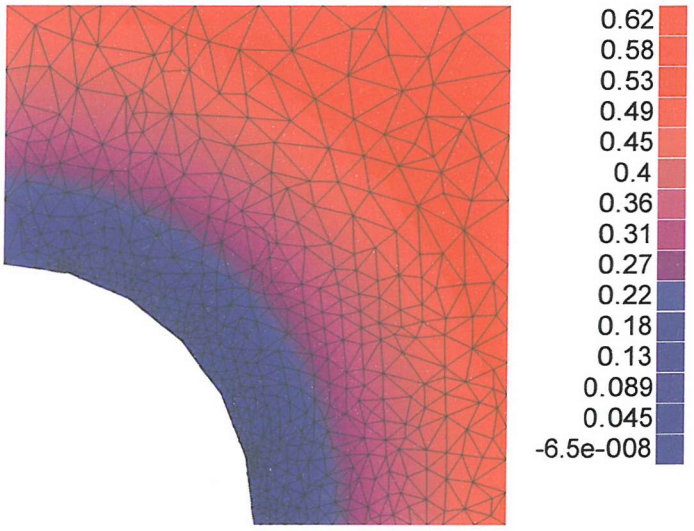

Figure 4.14: The mesh and concentration field around a hemispherical electrode, approximated by six subdivisions, at which a diffusion controlled reaction occurs. Since there is no singularity, refinement is largely controlled by geometric necessity and the radial field gradient. Due to the low error tolerance some elements are poorly shaped. The error, attributable primarily to the geometric approximation, is approximately 1% here.

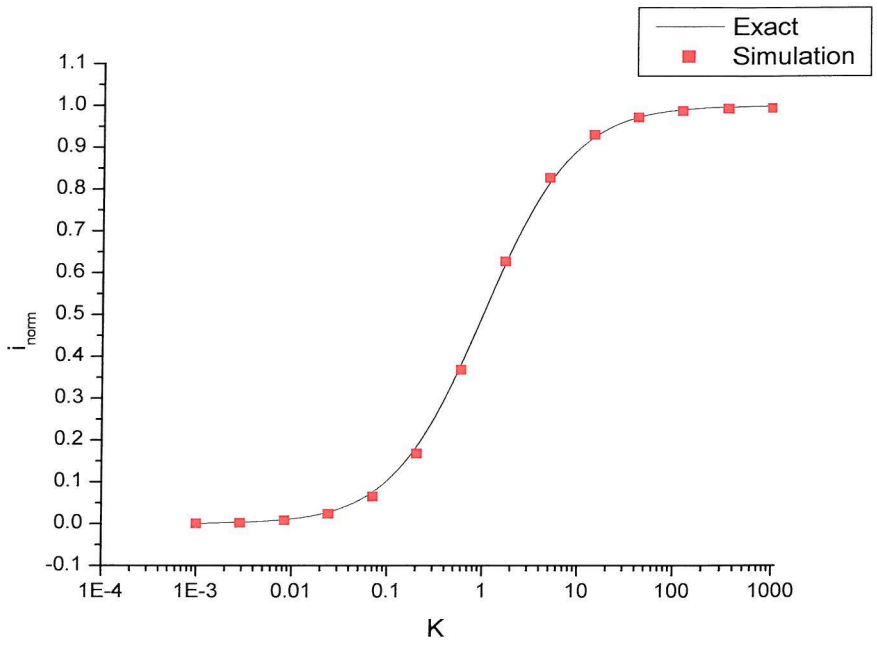

Figure 4.15: The current to a hemispherical electrode with various values of the dimensionless heterogeneous rate constant, K, in the boundary condition $\frac{\partial u}{\partial n} = -Ku$. The hemispherical cross-section was approximated by ten straight edges. The prescribed tolerance was 1%. As $K \to 0$, the electrode becomes an insulator, and the current is clearly zero. As $K \to \infty$, the effect of the boundary condition becomes like that of $u = 0$, and the dimensionless current tends towards the value for that case, unity.

4.3.4 Conclusion

We have only studied one case of curved geometry approximation, but the hemispherical case is an important one in electrochemistry, and the results are encouraging. While it would undoubtedly be more convenient to specify the boundary in terms of exact curves, approximation with straight line segments does not appear a great handicap in the pursuit of accurate results. In common with infinite domains, it does introduce an element of human intervention into the simulation process, but based on the results here there seems little reason to suspect that a hemisphere approximated with ten or more straight lines would yield results significantly different from those produced by an exact boundary. If modelling hemispherical electrodes were our aim, we would use one of the many one dimensional simulation programs already available, and avoid inexact geometry. But if we move, for instance, to hemispherical SECM tips, a two dimensional program is necessary, and the results here are valuable in showing program capability in that area.

As for heterogeneous kinetics, we have evidence to suggest that they do not present any great challenge (provided of course that they are first order; non-linear heterogeneous reactions would require the techniques sketched in Chapter 6). Using our program, changing a single line to reflect the different boundary condition is sufficient to allow the introduction of heterogeneous kinetics. This is in stark contrast to the Taylor series analysis required in finite difference, even for Robin boundary conditions at axially aligned edges; for the relatively complex shape simulated here, finite difference would be even more problematic. This illustrates well the great advantage of finite element over finite difference: its modularity, allowing different features to be combined painlessly. All this said, the case under consideration here would not prove particularly difficult to deal with using boundary element, although many practitioners appear to have ignored heterogeneous kinetics. Finally, perhaps most pertinently, the considerations raised in §3.2.6 and in [139] apply here: the error estimator of Harriman *et al.* insists, as it stands, on

only Dirichlet boundary conditions being present at electrodes. This marks a deficiency.

4.4 Generator-Collector Dual Microbands

Angus [18] has given a BEM-derived currents to various generator-collector microband arrangements, all without homogeneous reactions. For some cases analytical results also exist, facilitating comparisons. As a test of our simulation program in a Cartesian domain with several edge singularities, we examine the dual microband configuration.

4.4.1 The Problem

The inlaid generator-collector microband configuration is shown in Figure 4.16. All dimensions are normalised according to the electrode width, which is assumed to be equal for the the two electrodes. Species A diffuses towards the left hand electrode, reacts there to form species B, and then reverts to species A at the right hand electrode. These are termed respectively, in reference to species B, the generator and collector electrodes. Both reactions are assumed to be diffusion controlled, so the concentrations of A at the generator and B at the collector are zero. In the bulk A is assumed to proponderate, so $c_A \to c_A^*, c_B \to 0$ away from the electrodes.

Inside the domain the two species' concentrations are both described by Laplace's equation:

$$\nabla^2 c_A = 0 \,, \tag{4.29}$$

$$\nabla^2 c_B = 0 \,. \tag{4.30}$$

If one of the concentration fields is known the other may be determined from the usual relation:

$$D_B c_B = D_A (c_A^* - c_A) \,. \tag{4.31}$$

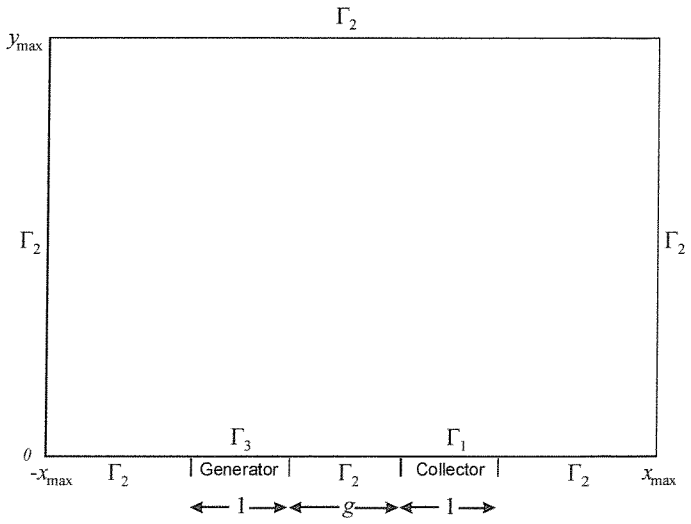

Figure 4.16: The dual microband domain. For the generator-collector experiment we impose $u = 0$ on Γ_1 (the collector), $u = 1$ on Γ_3 (the generator), and $\frac{\partial u}{\partial n} = 0$ on Γ_2, the insulator/far field boundary.

Knowing this, it does not matter which concentration field is simulated. We choose c_B. The normalisation

$$u = \frac{D_B c_B}{D_A c_A^*}\,, \tag{4.32}$$

gives the boundary conditions shown in Figure 4.16.

Since we are solving in Cartesian space, the PDE for the normalised concentration of species B is in full

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = 0\,. \tag{4.33}$$

The dimensionless current is defined by

$$i_{norm} = \int_{\Gamma_1} \frac{\partial u}{\partial n} d\Gamma\,. \tag{4.34}$$

With the boundary conditions used this is easily rewritten in terms of a domain integral of one field, in much the same way as E mechanism problems.[3]

[3]It is found if this is done that the domain integrals for the generator and collector

The question of the bulk boundary condition is discussed by Angus (*ibid.*). Theoretically u tends to zero away from the electrodes, but if this is imposed on a finite domain it leads to drastically wrong currents because the bulk boundary effectively forms its own collector, swamping the results. Moving the far field boundary further away does (with our simulation, at least), partially alleviate the problem, but even with $x_{max} = y_{max} = 2000$ the problem is still pronounced.

If instead an insulating boundary condition is imposed on the bulk boundary, accurate currents can be retrieved with reasonable domain sizes, and this we proceed to do. This option also has the happy advantage that it makes the problem inverse dual, saving computer time. Unfortunately it has the effect of making the concentration field tend toward 1/2, rather than the correct value of 0—the situation seems almost paradoxical. Presumably infinite elements would help considerably with the problem.

The analytical solution to the inlaid dual microband problem is given below. Another known case is that of an infinite expanse of interdigitated microbands, but modelling these exactly would require periodic boundary conditions, which we do not currently allow in our program. If we were so inclined we could automatically generate input files for the multiple electrode cases simulated by Angus. There seems little reason to doubt, however, that the results there are correct, tending as they do towards the analytical infinite array value, and with no particular difference in the nature of each successive microband's edge singularities.

More interesting, from a simulation perspective, are the raised and recessed dual microband configurations. The problem is the same as the analytically solved inlaid version, but with each active surface, generator and collector alike, raised by a dimensionless distance h, as shown in Figures 4.17 and 4.18. The two variants differ in whether the vertical sides are active or

currents differ only in sign, meaning the disparity between the two electrodes' current magnitudes investigated by some cannot exist with this particular arrangement. The choice of Γ_1 in (4.34) is therefore purely arbitrary: it could equally be Γ_3.

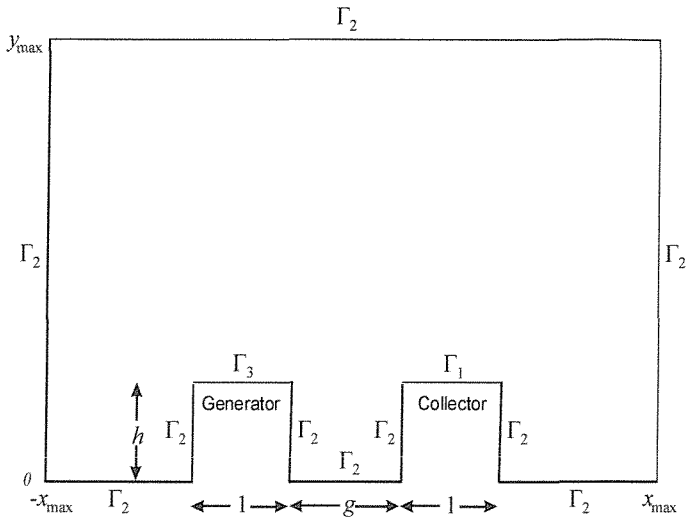

Figure 4.17: The "top" raised dual microband domain. The same boundary conditions as in Figure 4.16 are imposed on Γ_1, Γ_2 and Γ_3. The sides of the electrodes are inactive, and are hence designated Γ_2.

not. These types of electrode could be constructed photolithographically.

4.4.2 Analytical Solution

The inlaid dual microband problem has been solved using a conformal map [19, 20]. Essentially, a mapping in the form of an elliptic integral folds the domain up to transform the problem into a trivial one very similar to that solved for thin layer cells [2]. Apparently this was first done in an electrochemical context by Amatore and Fosset [144]. In our normalised domain the current depends on the ratio $m = g/(g+2)$ (where g is the gap between electrodes shown in Figure 4.16), and is given in our case by

$$i_{dmb} = \frac{K(\sqrt{1-m^2})}{2K(m)}, \tag{4.35}$$

where $K(\cdot)$ denotes the complete elliptic integral of the first kind [145].

In principle the other configurations considered here could be solved with conformal mapping, using the Schwarz-Christoffel mapping theorem [19–21].

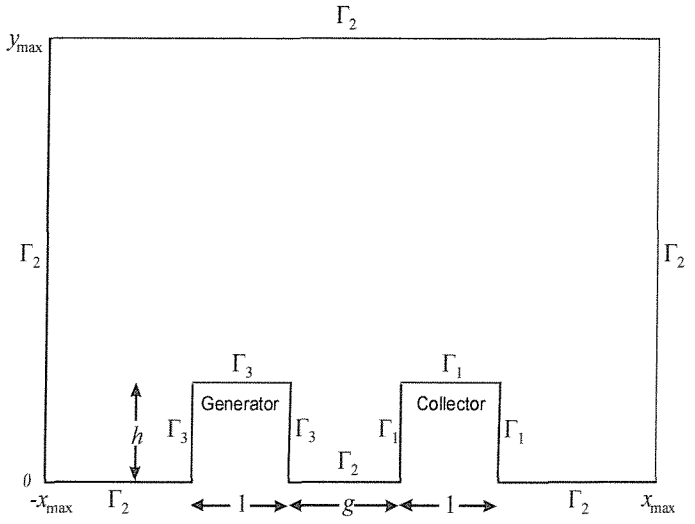

Figure 4.18: The "all" raised dual microband domain, where the sides of the electrodes are active. The same boundary conditions as in Figure 4.16 are imposed on Γ_1, Γ_2 and Γ_3.

The resulting expression might well be unmanageable, however, and there is apparently nothing of this kind in the literature. Qualitatively, we expect reduced currents relative to the inlaid case for recessed electrodes, owing to shielding effects. Conversely, raised microbands would be expected to have larger currents in both cases because of enhanced accessibility. On top of this, the "all" configuration will plainly tend towards a current proportional to h as it increases, since electrode area is proportional to h, and the concentration field within the inter-electrode recess would tend towards a type of thin layer cell situation. By contrast, we anticipate the "top" raised microbands to have a finite limiting current with increasing h, since the electrode area remains constant, and the effect of the insulating surround would become vanishingly small.

4.4.3 Simulation

Only results for $m = 1/3$ (that is, a gap size equal to the electrode size) are given, as simulations with other values of m resulted in similar accuracy and times. After briefly checking the inlaid case, the raised and recessed configurations are simulated with a range of values of h.

A sample mesh with its concentration field for the inlaid case is shown in Figure 4.19 on the next page. With $x_{max} = y_{max} = 50$ and 1% error tolerance, the calculated value of the current is 0.7827. (Testing showed that, for all the simulations run in this section, no significant difference was observed in the results for domain sizes larger than 50, so this size was used for all cases.) The analytical value is 0.7817, which implies approximately 0.1% error. This simulation took 2.1 s to run. Evidently the simulation program is successful in solving the inlaid problem.

The case of recessed and raised microbands is more involved, as we have no analytical expression with which to compare. The results over a range of values of h, positive and negative, are shown in Figures 4.20 and 4.21.

Qualitatively, the currents are in agreement both with physical intuition and with the curves given by Angus. Quantitatively, however, there is clearly a discrepancy with Angus's results for both configurations for certain values of h. In both cases, at its worst, this amounts to approximately 5% of the estimated value, which might not give too much cause to worry. However, the prescribed tolerance of 1% leads us to expect a better agreement if both sets of results are correct, and altering both this parameter and the domain size led to relatively negligible change in the FEM currents.

If our results are indeed the more correct, an explanation could perhaps be advanced. The BEM simulation is reported to be fairly dependent on the element discretisation used, with various effects, such as oscillatory flux values, arising from different spacings. While tests were apparently conducted establishing convergence, it is possible that exhaustive tests were not conducted for all values of h. Perhaps most crucially, the same number of

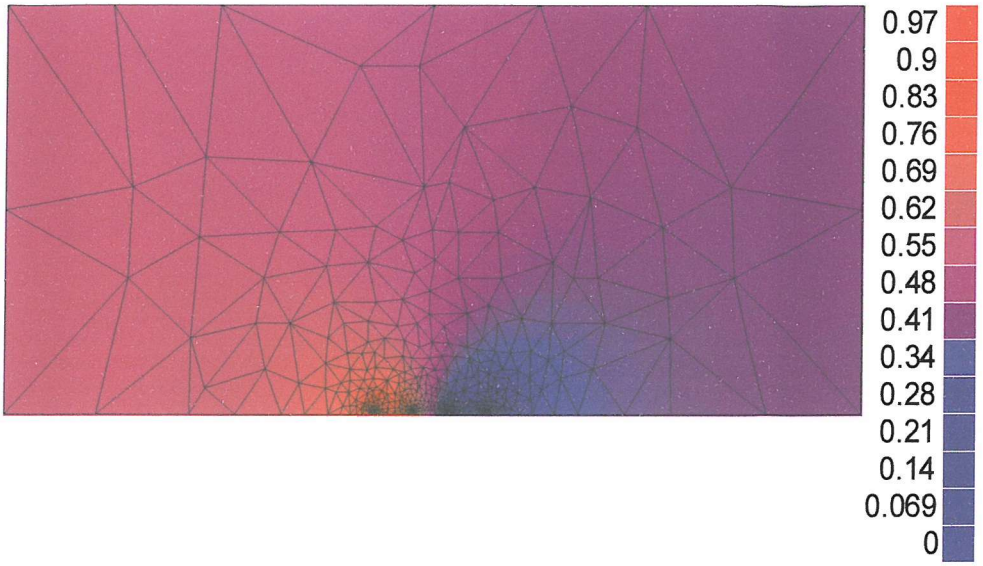

Figure 4.19: The mesh and concentration field generated by the simulation for an inlaid microband with $x_{max} = y_{max} = 10$ and an error tolerance of 1%. Notice the refinement at the four edge singularities. In practice larger domains are necessary for accurate currents. The bulk boundary condition causes the concentration field to tend, incorrectly, to $u = 1/2$ away from the electrodes.

elements was used for each simulation run, irrespective of h. Table 4.8 shows that our finite element simulation requires many more degrees of freedom for some cases than others, and there is a fairly strong correlation between the number of DOF and the discrepancy between our results and those produced by Angus [18]. One way or the other, then, the node counts in Table 4.8 alone would suggest a likelihood of divergence of opinion between the two methods.

The question of which is correct is of course difficult to answer. It may be that our simulation performs poorly in this situation when it deploys more DOF, but validation in other cases as well as theory points away from this. On balance, given the greater sophistication of error control in our

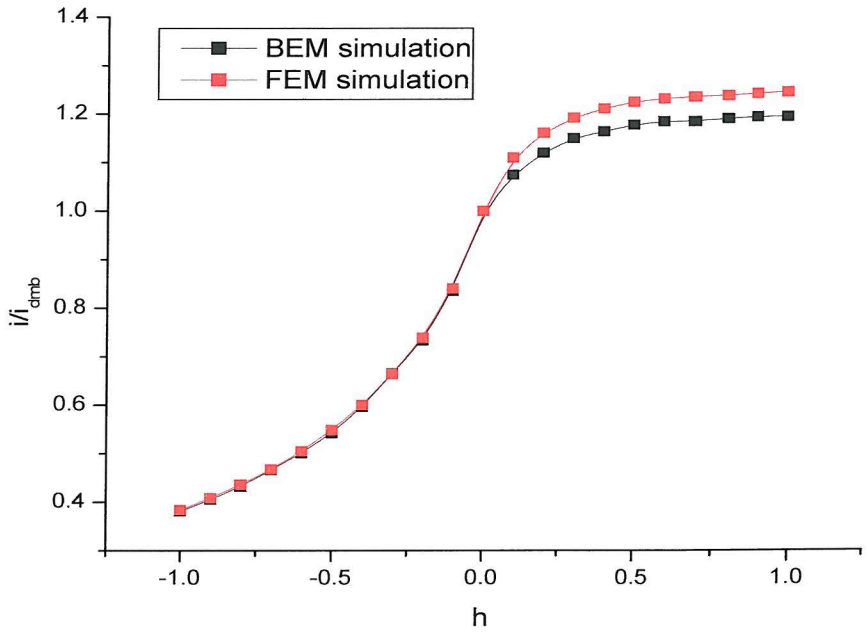

Figure 4.20: The effect of h on the "top" configuration microband current, normalised against the inlaid value. Our simulated values, denoted "FEM simulation", are compared with those of Angus [18], called "BEM simulation". For these simulations $x_{max} = y_{max} = 50$ and the error tolerance was 1%.

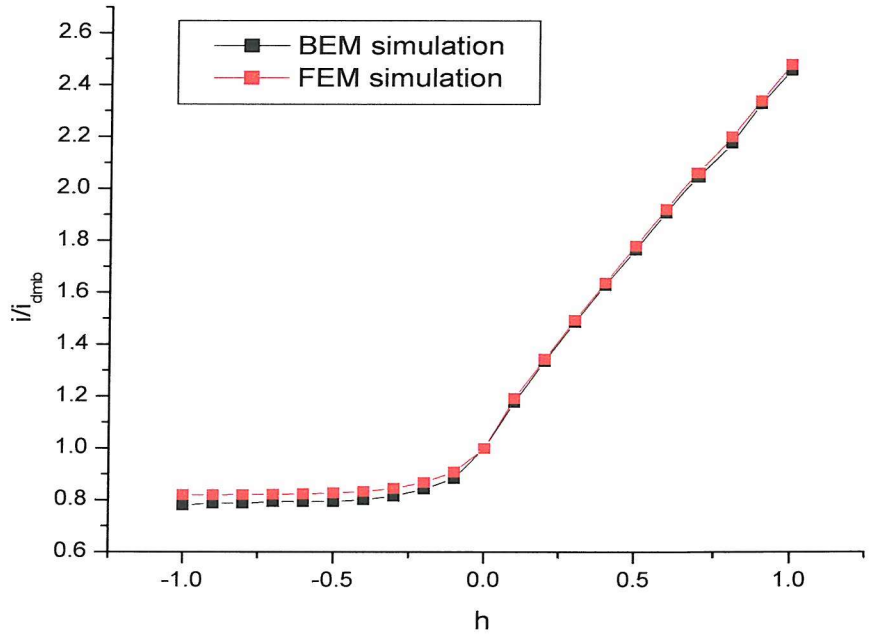

Figure 4.21: The effect of h on the "all" configuration microband current, normalised against the inlaid value. Simulation parameters were the same as in Figure 4.20 on the page before.

h	"All" DOF	"Top" DOF	"All" discrep.	"Top" discrep.
-1.0	4895	1482	4.8%	0.71%
-0.8	4827	1482	4.1%	0.99%
-0.6	5120	1581	4.2%	0.79%
-0.4	4639	1582	3.9%	0.67%
-0.2	4057	2089	3.0%	0.81%
(0.0)	(2193)	(2193)	(∼0.1%)	(∼0.1%)
0.2	2240	4199	0.74%	3.5%
0.4	2079	4699	0.61%	3.9%
0.6	1914	4728	0.67%	3.9%
0.8	1853	5060	1.1%	3.9%
1.0	1516	5051	0.9%	4.1%

Table 4.8: The computational complexity of the various raised/recessed microband simulations, and the discrepancy with the BEM simulation results of Angus [18]. Each FEM simulation was performed with $x_{max} = y_{max} = 50$. The geometric parameters were $w = g = 1$. The percentages are relative to the FEM value.

approach, we think it reasonable to assume that our results are the closer to reality. If this is so then the node counts give an explanation: while the FEM simulation adaptively employs more nodes for difficult cases, reaching a roughly constant accuracy, the BEM simulation, because of its *a priori* element density specification, is only capable of modelling certain simpler cases to within 1%, the accuracy degrading in others. Testing this hypothesis further would require re-running the BEM simulations for the problem cases with more elements, and carefully examining convergence.

4.4.4 Conclusion

As would be expected, having successfully solved cylindrically symmetric problems, Cartesian systems do not prove to be any great challenge. The effort required to obtain our results is obviously considerably less than is expended in showing convergence of the boundary element approximation, or in constructing the analytical solution via conformal mapping for that matter.

As for the results, the inlaid microband case offers further evidence of the reliability of our algorithm. The recessed and raised configurations, on the other hand, inevitably cast some doubt on our procedure in the absence of tertiary validating evidence; but given our stronger theoretical justifications, and the inductive evidence of other simulations, we suggest that the BEM results are the more questionable. If this is indeed the case, it illustrates the danger, once again, of non-adaptive meshing. In the end, of course, the difference is at most 5%, which might be enough accuracy for some applications.

4.5 EC' Microdisc

After covering a variety of geometries with the E mechanism, it is well to test the simulation with a different mechanism. The EC' mechanism is suit-

able for this purpose, as it can still be modelled with one field, and for the microdisc geometry there are approximate expressions as well as simulation results with which to compare. A fundamental part of our error estimation approach—that of comparison between the concentration field and its lower order interpolant—has yet to be tested, so this forms a crucial part of its validation.

Since Harriman *et al.* simulate both inlaid and recessed microdisc cases for the EC' mechanism [36], we do so too, in order to compare the efficiency of our algorithm with that of theirs.

4.5.1 The Problem

The EC' process we model is one where a reversible heterogeneous reaction acts, alongside an irreversible homogeneous one, in a catalytic mechanism of the form:

$$\mathrm{A} \pm ne^- \rightleftharpoons \mathrm{B} \tag{4.36}$$

$$\mathrm{B} + \mathrm{Z} \xrightarrow{k'} \mathrm{A} + \mathrm{Y} \, . \tag{4.37}$$

(Here Z and Y are considered inactive.) If the species Z is in large excess then its concentration may be approximated as constant, and absorbed into a pseudo-first order rate constant, $k = c_Z k'$. The governing PDEs of the active species are then, assuming purely diffusive mass transport,

$$D_A \nabla^2 c_A + k c_B = 0 \, , \tag{4.38}$$

$$D_B \nabla^2 c_B - k c_B = 0 \, . \tag{4.39}$$

Away from the electrode, species A is assumed to preponderate, so $c_A \to c_A^*$ and $c_B \to 0$.

At the electrode surface, assuming reversibility,

$$\frac{c_A^s}{c_B^s} = \delta \, . \tag{4.40}$$

As ever, mass balance must also hold there:

$$D_A \frac{\partial c_A}{\partial n} = -D_B \frac{\partial c_B}{\partial n} \, . \tag{4.41}$$

Galceran *et al.* [23] describe the process for mathematically eliminating coupling between the two species. In a similar fashion to the E mechanism, an acceptable functional form for one of the species' concentrations can be posited, this time for species A. Assume that

$$c_A = \frac{D_A c_A^* - D_B c_B}{D_A} \, . \tag{4.42}$$

Substituting this into (4.38) gives (4.39); therefore, given C_B it is possible to retrieve, via this expression, a form of C_A that satisfies the necessary governing equation. It should be obvious that the mass balance relation, (4.41), is also satisfied by a pair of concentrations c_A and c_B conforming to (4.42).

The surface concentration of B is given by the reversibility condition (4.40):

$$c_B^s = \frac{D_A C_A^* \delta}{D_A + D_B \delta} \, . \tag{4.43}$$

Defining the dimensionless concentration of B to be

$$u = \frac{c_B}{c_B^s} \, , \tag{4.44}$$

the surface condition becomes $u = 1$, and the bulk boundary translates to $u \to 0$.

The two microdisc domains are clearly the same as for the E mechanism microdisc simulations, and are shown in Figures 4.1 and 4.7. In these domains we wish to solve the modified Helmholtz equation in cylindrical coordinates:

$$\frac{1}{r} \frac{\partial}{\partial r} \left(r \frac{\partial u}{\partial r} \right) + \frac{\partial^2 u}{\partial z^2} - K u = 0 \, , \tag{4.45}$$

where K is a dimensionless homogeneous rate constant defined by

$$K = \frac{k a^2}{D_B} \tag{4.46}$$

(a is the dimensional radius of the microdisc). The dimensionless current is defined in essentially the same way as for the E mechanism microdisc simulations.

Considering the one dimensional form of (4.45) gives some insight into the difference in the functional form of the solution. The one dimensional Laplace equation, which would hold (if a steady state existed) for a large flat electrode, admits only linear solutions, whereas an equation of the form

$$\frac{d^2u}{dz^2} - Ku = 0 \tag{4.47}$$

will be expected, if K is real, to give decaying or growing exponential solutions. Since the latter could not represent a concentration field, based on this admittedly crude heuristic argument one might expect the effects of the electrode reaction boundary condition to fall away exponentially from the electrode—more sharply than with the E mechanism. This would have implications for the domain size necessary to approximate accurately an infinite expanse of solution.

4.5.2 Approximate Solutions

Various approximate expressions exist for this EC' case, including for recessed microdiscs, documented by Galceran *et al.* [23]. We give some of these here, but since we also desire to make comparison with the simulation results of [36] it is convenient to run simulations with the same parameters, and so their calculated values of the exact current are available.

Inlaid microdisc

For the inlaid microdisc a number of options exist. Galceran *et al.* [23] give an exact solution, but since it is in terms of the solutions of an infinite set of linear equations it is in practice approximate. In any case, using their solution they found the Padé approximant of Rajendran and Sangaranarayanan to be

more than 0.01% accurate for all values of K. Consequently we can use that expression, which takes the form:

$$i = \frac{1 + 2.0016K^{1/2} + 1.8235K + 0.96367K^{3/2} + 0.307949K^2 + 0.049925K^{5/2}}{1 + 1.3650K^{1/2} + 0.8826K + 0.32853K^{3/2} + 0.063566K^2} . \tag{4.48}$$

For $K > 10$ the more convenient asymptotic expression of Phillips [146],

$$i = \frac{\pi}{4}\left(\sqrt{K} + 1 + \frac{1}{4\sqrt{K}}\right) , \tag{4.49}$$

is reported to be at least 0.27% accurate.

Recessed microdisc

An infinite system expression from Galceran *et al.* [23] is also available for the recessed microdisc, but this is tiresome to evaluate. They also adapt Bond's strategy of assuming a constant mouth concentration [25] to the EC' case, but this was found to diverge from the true value by as much as 3.6%. A simpler alternative, is

$$i = \frac{\pi}{4}\sqrt{K}\coth\left[\sqrt{K}(L + \zeta)\right] , \tag{4.50}$$

where

$$\zeta = \frac{\pi}{4}\left[1 + \frac{1}{21.4479 + 0.2564\coth(0.3439L)}\right]\frac{1}{\left(1 + 0.8\sqrt{K}\right)^{3/4}} , \tag{4.51}$$

which is estimated to be 2% accurate. We use this for graphical comparisons, but since none of the approximate expressions is as accurate as our simulation is designed to be, we use for detailed numerical comparisons the calculated values given in Harriman *et al.* [36], using Galceran *et al.*'s exact expression.

Finally, for large values of K or L it is useful to note that the problem tends towards the one dimensional, and is easily solved by considering (4.47). The solution for finite u having $u(0) = 1$ is

$$u = e^{-\sqrt{K}z} . \tag{4.52}$$

The flux at $z = 0$ is then given by $-\sqrt{K}$, and the current by

$$\sqrt{K}\frac{\pi}{2}\int_{r=0}^{r=1} r dr = \frac{\pi}{4}\sqrt{K} \; . \tag{4.53}$$

4.5.3 Simulation

It is clear both from the explanations above, and from Harriman *et al.* [36], that a relatively small domain approximates the infinite one on which the problem is posed. As a consequence we adopt $r_{max} = z_{max} = 10$ for the inlaid problem, as is done by Harriman *et al.* For the recessed problem we again use their extents, in this case of $r_{max} = z_{max} = 4$ (the recess shields the microdisc, reducing the far field effect). We are in a position then to make a detailed comparison of accuracy and performance with their work.

As described in Chapter 3, the extensions of the domain integral expressions of (4.17) for the EC' case are fairly simple, the only notable difference being that they incorporate references to the field itself, as well as the gradient terms present previously:

$$d = \frac{\pi}{2}\int_{\Omega} (\nabla v \cdot \nabla u + Kuv)\, r dr dz \; , \tag{4.54}$$

$$\hat{d} = \frac{\pi}{2}\int_{\Omega} (\nabla \hat{v} \cdot \nabla \hat{u} + K\hat{u}\hat{v})\, r dr dz \; . \tag{4.55}$$

In practice, since the problem is self-dual, u is substituted for v, and $\hat{u}$ for $\hat{v}$. As previously, we continue to use $\hat{d}$ as our best estimate of the simulation current. This uses the quadratic order smoothed gradient in combination with the full quadratic field, and is therefore expected to give the greatest accuracy. The estimated error, however, remains that of d.

Inlaid Microdisc

The results for a range of values of K, presented in a similar fashion to [36], are shown in Table 4.9 on page 187. Clearly the simulation produces results within the expected error range in reasonable times.

Comparison with Table 1 of Harriman *et al.* [36] is instructive, given the various parameters' equality. We find that their simulation uses far more nodes (their numbers are 2665, 2417, 3452 and 7794 for the four values of K) and, even attempting to compensate for machine speed differences, is slower. The times scaled by clock speed are, respectively, 10.2 s, 7.68 s, 8.44 s and 19.3 s. At its best, their algorithm manages approximately a third of the speed of ours; at its worst, a tenth. This last figure is apparently the outcome of their error estimator's increasing over-estimation with larger values of K: it seemingly has difficulties with homogeneous terms, whereas ours generalises to these more easily. Overall the method may be judged to be markedly more efficient.

On the subject of node counts and overestimation, Harriman *et al.* present their estimator's exaggeration of error as a virtue, stressing that "[w]e have again achieved much better accuracy than the tolerance that we have prescribed, particularly so for $K \gg 1$". This is arguable, since in a sense it represents a lack of control over their simulation. With our scheme, using its tighter bound, greater efficiency can obviously be achieved. (We can also actually achieve comparable accuracy using $\hat{d}$, it but its accuracy is unpredictable.) On the other hand, a more conservative bound is evidently less likely to fail by underestimating the error. Based on the empirical evidence in this work it seems our bound rarely fails, so there are some grounds for claiming its superiority in the electrochemistry arena.

Recessed Microdisc

The recessed microdisc is a similar story. In Table 4.10 on the next page we echo the format of Table 2 in Harriman *et al.* [36], again to allow direct comparison. As can be seen from this table, the recessed problem is easier, with lower node counts, at least for these values of K and L. With very large values of K, however, very small elements are required near the electrode surface to capture the sharply exponential form of the concentration field (of

K	Exact current	d	$\hat{d}$	Est. error	Nodes	CPU time
1	1.689	1.696	1.687	0.007483	243	0.931 s
10	3.322	3.336	3.324	0.004804	331	1.42 s
100	8.658	8.702	8.651	0.006839	700	2.96 s
1000	25.63	25.81	25.64	0.007137	1651	5.01 s

Table 4.9: Results for the EC' mechanism reaction with inlaid microdisc geometry. For each simulation $r_{max} = z_{max} = 10$, and the error tolerance was 1%; the only variable was the dimensionless homogeneous rate constant K. The "exact" current was calculated from (4.48).

K	Exact current	d	$\hat{d}$	Est. error	Nodes	CPU time
1	1.010	1.015	1.011	0.008449	94	0.209 s
10	2.510	2.530	2.513	0.008160	91	0.148 s
100	7.854	7.920	7.854	0.008555	230	0.377 s
1000	24.84	24.98	24.83	0.006050	997	2.24 s

Table 4.10: Results for the EC' mechanism reaction with a recessed microdisc of depth $L = 0.5$. For each simulation $r_{max} = z_{max} = 4$, and the error tolerance was 1%. The "exact" current was taken from [36].

course, here asymptotic expressions are available). Meshes and concentration fields for the simulations of Table 4.10 are shown in Figure 4.22.

The results again show our algorithm to be as accurate as that of Harriman *et al.*'s, but faster, with less over-refinement. The scaled times of their simulation for the four values of K in Table 4.10 are 1.31 s, 0.964 s, 1.33 s and 5.22 s respectively. At best, then, they achieve half our speed; at worst, a sixth.

Harriman *et al.* give log-log plots of i versus K for various values of L. We do this in Figure 4.23. The simulation is robust beyond these parametric values; for instance for the case of $L = 0.05, K = 10^6$, it takes approximately

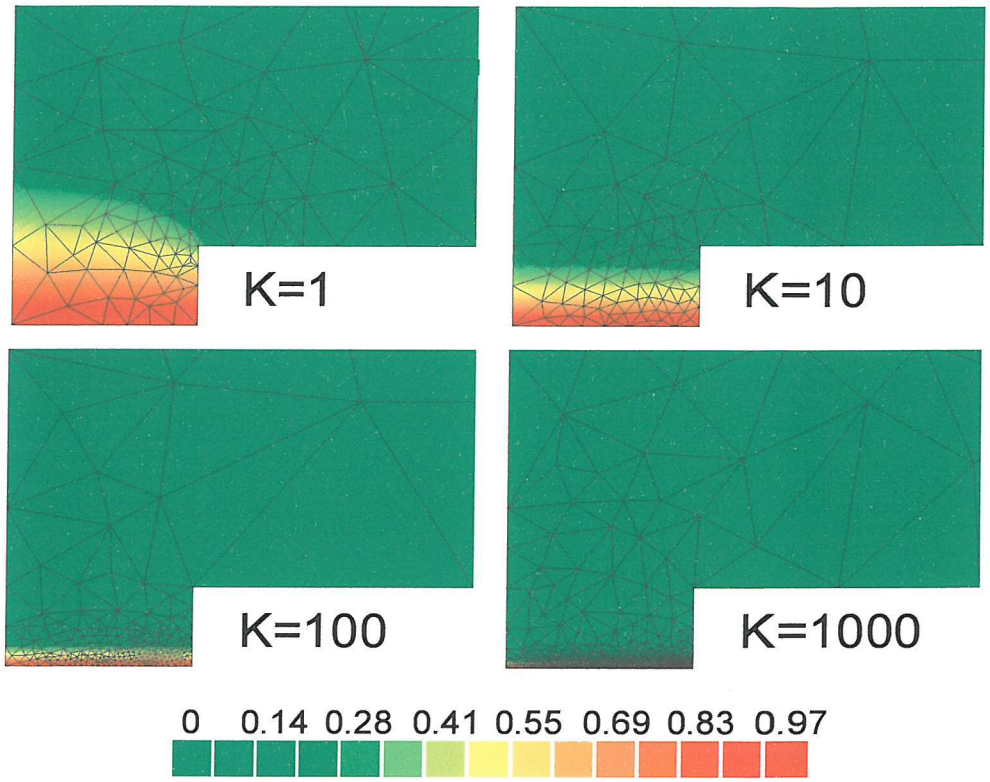

Figure 4.22: The mesh and concentration field around a recess depth of $L = 0.5$ for various values of the homogeneous rate constant, K. As K increases, refinement switches from the mouth corner to close to the bottom of the recess.

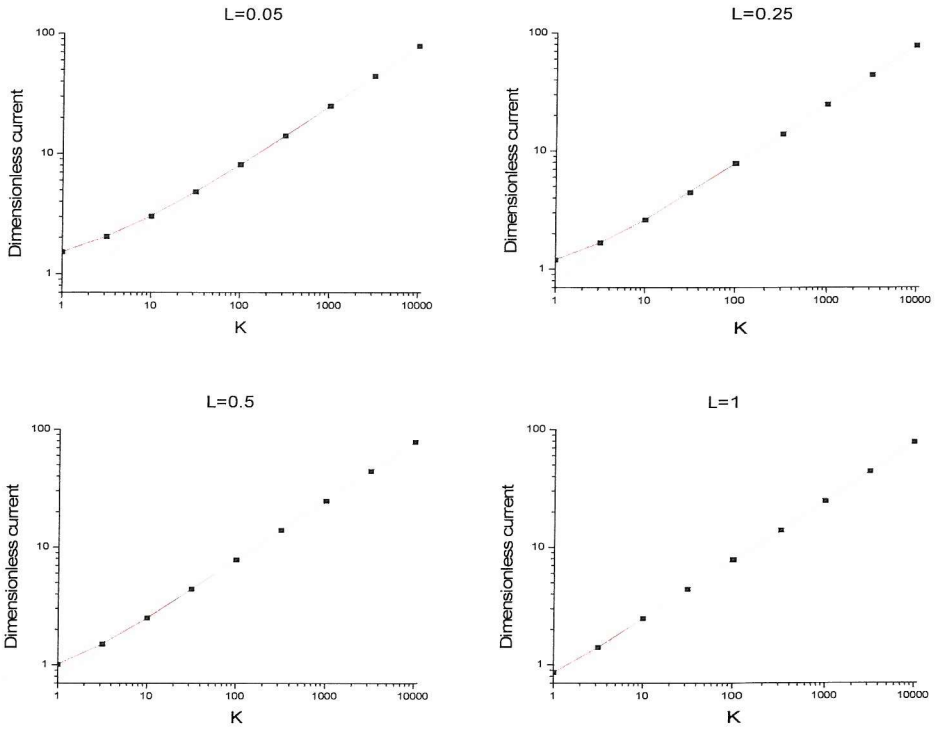

Figure 4.23: Double logarithmic plots of the simulated current (marked as ■) versus K for various recess depths. Also shown are equation (4.50) (in red) and the asymptotic expression (4.53) (in green).

three minutes to reach 1% accuracy. For all the simulations conducted we find the results as accurate as we would expect from the prescribed error tolerance and the reported accuracies of the approximate expressions.

4.5.4 Conclusion

Since the inlaid and recessed EC' microdisc case has been analysed extensively (see [23] and references therein), our simulations have not added anything valuable in terms of results, aside from serving as yet another test of formulae already given. The work in this section is more motivated by a desire both to validate the code with a mechanism involving homogeneous reactions, and to compare performance with [36]. The program passes the former test, clearly, and has the advantage of being in every case faster than the work cited. While a few seconds' difference for simple simulations like those undertaken here is clearly not a great factor, with more interesting, complex, simulations we would expect the speed increase to be more significant.

4.6 E Mechanism Channel Flow Microband

Apart from being a useful cell geometry for experimental electrochemistry, the channel flow microband arrangement is well suited for testing of convection simulation. The associated fluid dynamics problem has a simple exact solution and there are a variety of approximations to the mass transport problem. Harriman *et al.* give results for their adaptive finite element approach for the E mechanism, so a comparison of the algorithms' efficiency can be made in a convective context.

4.6.1 The Problem

The problem is one of convective-diffusive mass transport of species A to a microband electrode embedded in the side of a channel in which a Poiseuille

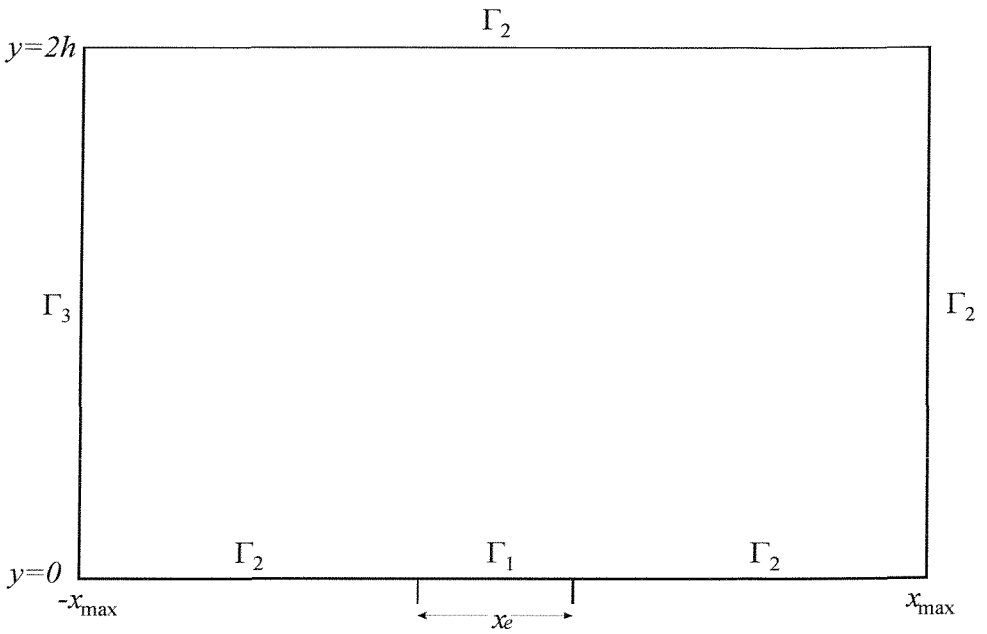

Figure 4.24: The channel flow microband domain. Material enters the system at the inlet on the left hand side, labelled Γ_3; there the normalised concentration u is unity. Reactant is consumed at the electrode, denoted by Γ_1, where $u = 0$. Elsewhere the $\frac{\partial u}{\partial n} = 0$.

flow regime [6,147] holds. At the microband a reaction occurs to form species B—one that we shall assume is reversible.

The cross-sectional domain is shown in Figure 4.24. Only the x component of the convective velocity field is non-zero, so the steady state convection-diffusion equations for species A and B are:

$$D_A \nabla^2 c_A - v_x \frac{\partial c_A}{\partial x} = 0 \,, \tag{4.56}$$

$$D_B \nabla^2 c_B - v_x \frac{\partial c_B}{\partial x} = 0 \,. \tag{4.57}$$

The velocity follows a parabolic profile defined by

$$v_x = v_0 \left[1 - \frac{(h-y)^2}{h^2} \right] \,, \tag{4.58}$$

where v_0 is the maximum velocity, found at the centre of the channel.

Reactant A enters the channel at the left hand inlet, so $c_A = c_A^*, c_B = 0$ there. Adopting a normalised concentration, $u(c_A)$, we impose $u = 0$ at the electrode, which may be interpreted in two ways.

If the diffusion coefficients D_A and D_B are equal then we can take

$$c_B = c_A^* - c_A \ , \tag{4.59}$$

and all potentials may be related to one simulation by taking

$$u = \frac{c_A - c_A^s}{c_A^* - c_A^s} \ . \tag{4.60}$$

With δ the ratio of surface concentrations,

$$\delta = \frac{c_A^s}{c_B^s} \ , \tag{4.61}$$

the surface concentration of A is given by

$$c_A^s = \frac{\delta c_A^*}{1+\delta} \ . \tag{4.62}$$

If the diffusion coefficients are not equal the simulation covers only the purely diffusion controlled case, where $c_A^s = 0$, and we take

$$u = c_A / c_A^* \ . \tag{4.63}$$

Both of these schemes lead to the bulk condition $u = 1$ at the inlet, and $\frac{\partial u}{\partial n}$ on the insulating walls of the channel. All of the boundary conditions are shown in Figure 4.24.

For the other quantities we adopt what appears to be the standard normalisation. The x and y coordinates are divided by x_e so that the electrode has a normalised size of one. Two dimensionless quantities are then defined:

$$p_1 = \frac{x_e}{h} \ , \qquad p_2 = \frac{4hv_0}{3D_A} \ . \tag{4.64}$$

The transport equation then becomes

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} - \left(\frac{1}{2} P_s y (2 - p_1 y) \right) \frac{\partial u}{\partial x} = 0 \ , \tag{4.65}$$

where $P_s = (3/2)p_1^2 p_2$ is the *shear rate Peclet number*, a global measure of the relative importance of convection.

The normalised current is given by

$$i_{norm} = \int_{\Gamma_1} \frac{\partial u}{\partial n} d\Gamma \ . \tag{4.66}$$

4.6.2 Approximate Solutions

Various different approximations exist for different rates of flow. Often they entail making the Lévêque approximation [148], where the convective velocity is taken to be approximately linear near the electrode—i.e. the quadratic term in the convection coefficient of equation (4.65) is neglected. By making the further approximation that mass transport in the x direction is purely convective, Levich [149] derived the simplest approximation to the current:

$$i_{levich} = 0.8075 P_s^{1/3} \ . \tag{4.67}$$

As a consequence of these assumptions, Levich's equation is applicable only at fairly high flow rates.

An alternative, for low flow rates, is that due to Ackerberg *et al.* [150]:

$$i_{ackerberg} = \pi g(P_s)[1 - 0.04633 P_s g(P_s)] \tag{4.68}$$

where

$$g(P_s) = \frac{1}{(\ln(4/\sqrt{P_s}) + 1.0559)} \ . \tag{4.69}$$

Finally, for moderate to high flow rates, Newman [151] gives a modified version of Levich's expression:[4]

$$i_{newman} = 0.8075 P_s^{1/3} + 0.7085 P_s^{-1/6} - 0.1984 P_s^{-1/3} \ . \tag{4.70}$$

The expression of Aoki *et al.* [152] would not appear to offer any advantage over those given above for any value of P_s, and so was not included in the comparisons below.

[4]Note that (4.67) and (4.70) are given incorrectly in [37] (as indeed is equation (16) in that work).

4.6.3 Simulation

The boundary conditions for this problem allow the use of $1 - u$ for the influence function. Thus a possible expression for the current using the higher order field $\hat{u}$ is

$$\hat{d} = \int_\Omega \|\nabla \hat{u}\|^2 - (1 - \hat{u})\frac{1}{2} P_s y(2 - p_1 y)\frac{\partial \hat{u}}{\partial x} d\Omega \ . \tag{4.71}$$

As ever, this type of expression carries the advantage that only one field need be solved for, but because we have convective terms the equation is not self-adjoint, and $1 - u$ is not the dual. To use the dual a second problem with convection in the opposite direction must be solved:

$$\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2} + \left(\frac{1}{2} P_s y(2 - p_1 y)\right)\frac{\partial v}{\partial x} = 0 \ ; \tag{4.72}$$

and the domain integral becomes

$$\hat{d} = -\int_\Omega \nabla \hat{u} \cdot \nabla \hat{v} + \hat{v}\frac{1}{2} P_s y(2 - p_1 y)\frac{\partial \hat{u}}{\partial x} d\Omega \ . \tag{4.73}$$

This presents an interesting performance tradeoff between the benefits of solving only one problem, and solving a two potentially requiring less refinement. In the apparent absence of any theory defining the optimal choice for v we can gather here some empirical data.

The channel microband simulation was run over a range of flow rates, with one of the sets of parameters used by Harriman *et al.* for comparison purposes:

$$D_A = 10^{-5} \text{ cm}^2\text{s}^{-1} \ , \tag{4.74}$$

$$x_e = 5 \times 10^{-4} \text{ cm} \ , \tag{4.75}$$

$$h = 0.02 \text{ cm} \ . \tag{4.76}$$

Various values of P_s were imposed by varying v_0. The dimensionless x extents given by Harriman *et al.* of 16 units in either direction from the edge of the electrode were also used (these were reported to be sufficient to approximate an infinite domain), and a 5% error tolerance imposed.

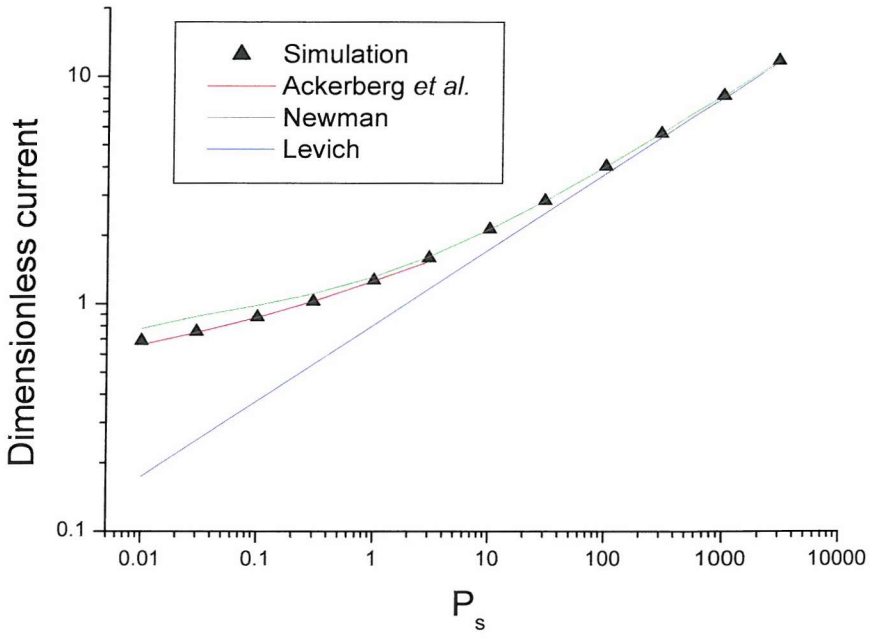

Figure 4.25: The simulated channel flow microband current over a range of flow rates. The approximate expressions of Levich (4.67), Ackerberg *et al.* (4.68) and Newman (4.70) are also shown. The simulation is seen to agree with one or other of these expressions, depending on their range of applicability.

Using these parameters it was found that, with the stabilisation scheme described in Chapter 2, a Jacobi preconditioned BiCGStab unsymmetric solver converged rapidly for all flow rates. The values for the current were close enough for the two forms of v that they are indistinguishable in the logarithmic plot of Figure 4.25, and they both largely agree with those of Harriman *et al.* and the approximate expressions. For larger shear rate Peclet numbers it should be noted that the non-dual form of v does give values closer to the approximate expressions, but see below. At low flow rates agreement is found with the expression of Ackerberg *et al.* (4.68); at higher

rates, the simulation agrees with Newman's approximation (4.70). As the shear rate Peclet number increases beyond 1000, the current clearly tends toward that predicted by Levich's expression (4.67).

A pronounced difference between the non-dual and dual v cases is found in the meshes produced, which is illustrated in Figures 4.26 and 4.27. In the first it is seen that the mesh refinement follows the path that one would expect for the material generated at the electrode. This is in contrast to the results of Harriman *et al.* and to the adjoint v case, where refinement is concentrated around the electrode, and more pronounced upstream, rather than downstream. Where flow rates are high—i.e. where the PDE is furthest from being self-adjoint—the effect of the $1 - u$ influence function is to cause much greater refinement downstream, where it isn't needed, and to greatly increase overall node count. Since the results for the current are correct in either case this is unwanted over-refinement, and points to the adjoint v being more efficient.

In the diffusion dominated case of $v_0 = 0.1$, giving $P_s = 0.25$, again at 5% accuracy, our algorithm out-performs that of Harriman *et al.* in both cases in respect of node counts. Where their algorithm required 2137 nodes, ours needed either 218 or 150 for non-dual and dual respectively. It might also be added that in the first case the simulated current is within 0.1% of equation (4.68)—closer than their estimate, but the accuracy of the approximate expression is not known.

On the other hand, in the convection dominated case, where $D = 10^{-5}$ cm^2 s^{-1}, $x_e = 10^{-3}$ cm and $v_0 = 50$ cm s^{-1} (giving $P_s = 500$), our simulation with $v = 1 - u$ requires only slightly fewer nodes—1558 versus 1644. The simulated current in this case was 6.63222, which is very close to the value of Newman's expression, 6.63562. This is a higher value than that cited in [37], which was 6.59956. If Newman's expression were known to be accurate to within three significant figures, then we could claim greater accuracy; but the various expressions' accuracy is not documented. The adjoint v simulation uses far fewer nodes—317—and produces a current of 6.80947,

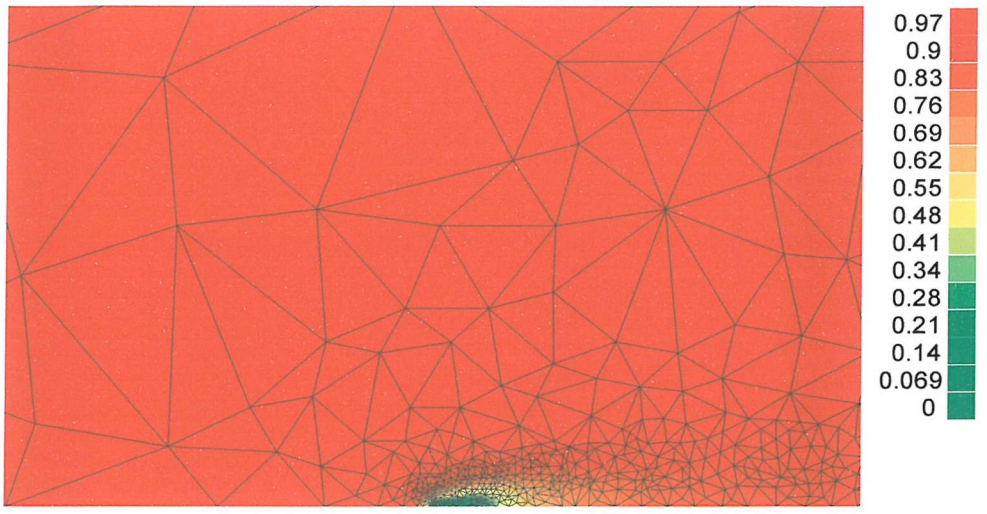

Figure 4.26: The mesh and concentration field for a typical channel flow microband simulation ($P_s = 30$), zoomed in on the area around the electrode. A 5% error tolerance was imposed on the current, and the non-adjoint v was used, giving 593 total nodes.

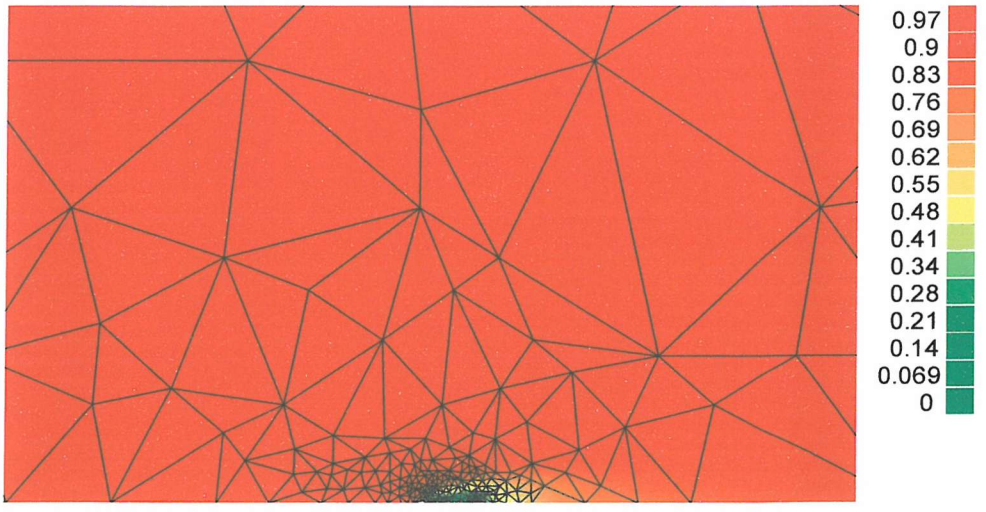

Figure 4.27: The mesh and concentration field for a typical channel flow microband simulation ($P_s = 30$), zoomed in on the area around the electrode. A 5% error tolerance was imposed on the current, and the adjoint v was used, giving 231 total nodes..

which is further from the the approximate expression, but within the 5% prescribed accuracy. The timings too are very different, with 18.2 s versus 2.7 s.

The slowest simulation run, at $P_s = 3000$, took 77 s for the $v = 1-u$ case, which while not unacceptable, is less impressive than the 7.3 s for adjoint v. Timings for lower flow rates were generally much lower, going below 1 s at the lower end of the P_s spectrum, and there the difference became less pronounced. Since the unsymmetric linear system solver used was almost certainly far from optimal, there is reason to hope the situation could be improved for both.

4.6.4 Conclusion

The channel flow microband case, with the simplest E mechanism, has illustrated the applicability of the adaptive FE algorithm developed in this work to convective cases. Clearly further testing in other situations would be desirable in order to be sure that the conclusions reached regarding stabilisation schemes and Krylov solvers generalise over the range of possible simulations, but nothing so far suggests that the algorithm fails where convection is present.

Comparison with the scheme of Harriman *et al.* [37] is made difficult by the paucity of information thereon. For instance, no idea is given of how the degree of stabilisation is varied over the range of flow rates (the "optimal" value of the stabilisation parameter given by Zienkiewicz and Taylor [74] would appear to perform well here). Nor are any timings given. From what information is given, it would seem that the algorithm used here is at least competitive with $v = 1-u$, and considerably more efficient with a v satisfying the adjoint equation.

Overall, the channel flow case is a successful one for the algorithm. That said, more investigation needs to be done on the nature of v, and better linear system solvers should also be evaluated. Convective problems are clearly

more challenging than purely diffusive ones, and more work is required to reach a satisfactory simulation method.

4.7 Conclusions

It is believed that the simulations described in this chapter show that the theory outlined in Chapter 3 does yield a useful and accurate simulation algorithm. Apart from the case of raised and recessed microbands, we have found agreement with the best available approximations; and there we argue that we have nonetheless produced accurate results.

As it stands, the program cannot simulate the majority of interesting cases, as it cannot handle transient situations or various more complex mechanisms. Nonetheless, the results of this chapter do appear to constitute a vindication of the theoretical approach taken in Chapter 3, and justify the programming effort required to produce a general simulation program of the type described there.

Having validated the adaptive algorithm over a range of problems where analytical or semi-analytical solutions exist, it is desirable to test it in an arena where simulation is essential. An example of such is SECM, to which we turn in the next chapter.

Chapter 5

SECM Simulations

An important area of electrochemistry demanding accurate simulation is that of scanning electrochemical microscopy (SECM). SECM is a broad field, and it is not possible to describe it in detail here. A book covering a variety of aspects of the technique is [64]. A review of the many SECM modes is given by Mirkin and Horrocks [153]. We consider only two of the various configurations in this work—those of negative and positive feedback.

For our purposes, an SECM tip may be considered a movable microelectrode embedded in a cylindrically symmetric insulating surround. In this chapter we simulate approach curves of a reactant-consuming microdisc tip moving towards either an insulating or conducting substrate. Since the simulation program is limited to two dimensions, we are confined to substrates that are cylindrically symmetric; here we consider those that are unvarying in the plane whose normal is parallel with the approach direction. This domain is complex enough to make analytical solution difficult,[1] but there have been various simulation studies. These have varied according to the geometric fidelity with which they have reproduced the experiment, with some ignoring the solution past the level of the electrode, and others assuming a cylindrical

[1]The only attempt at an analytical solution appears to be that of Bard *et al.* [154], but this is a restatement of the problem as an integral equation, and does not give an explicit function.

sheath shape. In this chapter a generally more realistic geometry is used.

5.1 The Problem

The aim is to quantify the current to the microdisc SECM tip as it moves towards the two types of substrate. Either reactant diffuses solely from the bulk to the tip, with the substrate an insulator, in which case we have *negative feedback*; or it is also regenerated at a conducting substrate, giving *positive feedback* [64].

A key approximation is usually made in SECM simulations: the tip approach is assumed to be slow enough to make the concentration field surrounding the tip approximate that of the steady state one that would be present if the tip were left in the given position *ad infinitum*. One can then solve the steady state diffusion problem for a range of positions of the tip, which is far simpler than considering either transient diffusion or the convective currents that would be caused by faster movement. Fortunately the steady state mode is often preferred for experimental reasons [153], and this approximation has been widely validated with other simulations.

For the insulating substrate case the same normalisation as with the E mechanism microdisc (see § 4.1.1 on page 134) can be used to extend simulation results to the range of reversible cases. However, for the conducting substrate case, as with the generator-collector case, there are two potentials to consider, and the full generality of cases cannot be represented by a single simulation. For simplicity's sake, therefore, in this case we assume diffusion control at both the tip and the substrate, and adopt the normalisation $u = c_A/c_A^*$. In this we follow Amphlett and Denuault [1] and many others. The concentration is therefore simply normalised by the bulk value as in § 4.4.1 on page 170.

The domain used for all microdisc simulations is shown in Figure 5.1, where the microdisc radius has been normalised to one. Initially we conduct

simulations with $\alpha = 0$, where the sheath is approximated as cylindrical, in order to afford comparison with the results of Amphlett and Denuault [1] (the less accurate earlier simulations with rectangular domains were effectively with $\alpha = \pi/2$). We explore the effect of this simplification later, but for now we follow the accepted practice.

Material diffuses from the bulk boundary, where $u = 1$, to the tip, where it is consumed, so $u = 0$. The sole difference between the two types of substrate is in the type of boundary condition on Γ_4—for a conducting substrate, material is produced here and is a significant determinant of the tip current. Both the cases under study are inverse dual, so we solve for only one field.

Clearly there is a relation with the inlaid microdisc case, and we do not expect the currents at the tip to be radically different from those to an equally sized microdisc with the same experimental conditions. For this reason the dimensionless current is often normalised by the inlaid microdisc value (note, though, that as explained below we use a different normalisation value). Two major distinctions, however, affect the results.

Firstly, the microdisc is facing, at a distance L, an insulating or conducting barrier, rather than an infinite expanse of solution. This will tend to lessen the current in the former case, and increase it in the latter. As the tip distance decreases this effect will become more dominant, until in the limit of $L = 0$ the current must be zero or infinite respectively. This effect has been investigated in a number of studies.

The second effect depends on RG, the ratio of the insulating surrounding sheath's radius to the microdisc's, and holds regardless of the substrate type. As $RG \to \infty$ the effect of the insulating surround clearly tends towards that around an inlaid microdisc. But for small values of RG with moderate L, diffusion around the corner becomes a significant factor, and can push the current above the value for an inlaid microdisc, potentially in spite of any insulating substrate's effect. This was described by Amphlett and Denuault [1], who based their conclusions on ADI finite difference simulation. They point out that, contrary to previous speculation [155], RG does noticeably

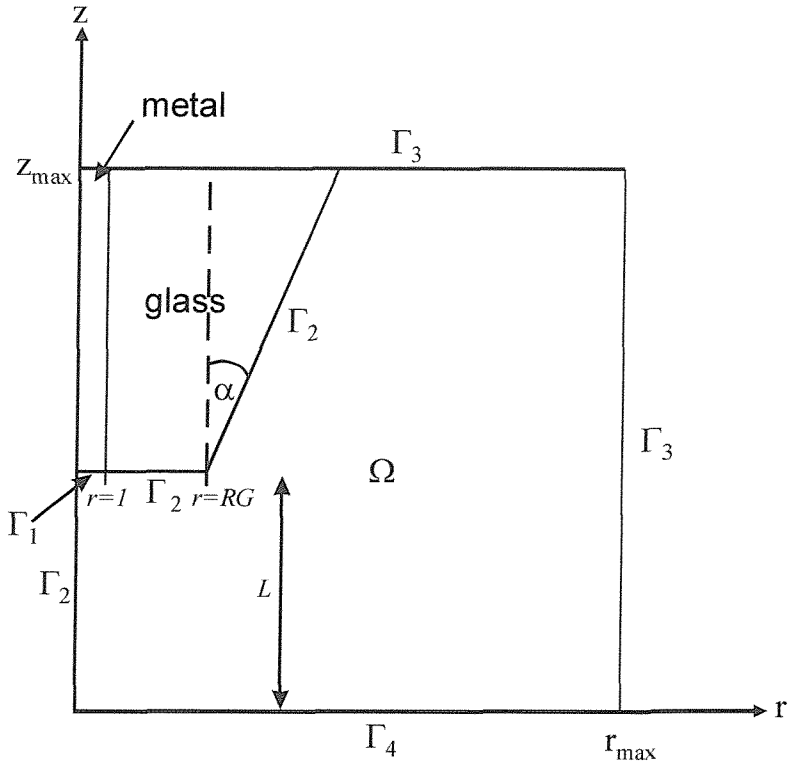

Figure 5.1: The microdisc tip SECM simulation domain. The dimensions are normalised according to the electrode radius. The tip electrode denoted by Γ_1 has $u = 0$ prescribed on it. The two insulating surround boundaries and the rotation axis denoted by Γ_2 have $\frac{\partial u}{\partial n} = 0$. The bulk boundary Γ_3 has $u = 1$. For an insulating substrate, $\frac{\partial u}{\partial n} = 0$ on Γ_4; for a conducting one, $u = 1$. The degree of tapering is defined by α.

affect the current even with a conducting substrate, although since diffusion from the bulk is less significant here the influence is clearly less pronounced. Fang and Leddy [156] also studied this phenomenon, as did Shao and Mirkin [157].

Since SECM experimental results are conveniently normalised by the limiting current as the tip tends towards an infinite distance from the substrate, and because this can differ significantly from the inlaid microdisc current, we normalise our results with respect to this, calculated by setting a tip distance of $L = 100$. This carries the advantage of allowing direct comparison with Amphlett and Denuault's results, as they also follow this practice. It is important to remember, then, that graphical presentation of the results in this form de-emphasises the impact of RG on currents; the difference in normalisation values should also be taken into account.

5.2 Comparison With Previous Simulations

Various researchers have simulated the system under study [158], and there are reasonably well accepted results with which to compare. The oldest and least sophisticated simulations ignored diffusion from the bulk around the sheath edge by cutting the domain off at the level of the tip, reducing the simulation domain to a rectangle. Since then, simulations have been conducted that do allow for this factor, but which approximate the sheath geometry as being that of a cylinder. Comparison with experimental data suggests that this latter approach is more accurate for small sheath radii (there is no difference for larger radii), but it remains an approximation to the true geometry, which is tapered. We begin by testing our simulation program with the cylindrical sheath geometry, before conducting a numerical investigation of the effect of this previously neglected tapering.

5.2.1 Approximate Solutions

Amphlett and Denuault [1] give expressions derived from fitting curves to ADI finite difference simulation results for various values of RG, for both insulating and conducting substrates. The simulation accuracy is presumably around 0.5%, as that was the accuracy obtained for the known microdisc case.

For an insulating substrate the expression is of the form

$$i = \frac{1}{k_1 + k_2/L + k_3 \exp(k_4/L)} \; . \tag{5.1}$$

The constants k_1, k_2, k_3, k_4 are given for a number of values of RG between 1002 and 1.11. A few sets of values are reproduced in Table 5.1 on the following page; we use the respective curves for evaluating our results. These curves have been validated against experimental data [159], and so are unlikely to be incorrect, even though the simulation domain did not incorporate tapering. A further validation is given by the agreement of the fitted curves produced for a few very similar values of RG by Shao and Mirkin [157].

The conducting substrate curve clearly has a different form, as it increases rather than decreases as $L \to 0$. The curve fitting expression for this case is

$$i = k_1 + k_2/L + k_3 \exp(k_4/L) \; . \tag{5.2}$$

Some values of the fitting constants are given in Table 5.2 on the next page.

5.2.2 Simulation

Since the expression (5.1) only fits the simulations from which it was derived to within 1%, and because the finite difference simulation accuracy itself is not exactly quantified, we use 1% simulation tolerance.

Based on the microdisc case simulated in the previous chapter (see Table 4.6 on page 151) we might expect the effect of domain size over $r_{max} = z_{max} = 50$ to be relatively small, and this is indeed found to be the case. For safety, and in order to accommodate the larger values of RG simply, we use $r_{max} = z_{max} = 500$. This does stress the meshing algorithm, somewhat: an

RG	k_1	k_2	k_3	k_4
100	0.27997	3.05419	0.68612	-2.7596
20.1	0.35541	2.0259	0.62832	-2.55622
10.2	0.40472	1.60185	0.58819	-2.37294
5.09	0.48678	1.17706	0.51241	-2.07873
1.51	0.90404	0.42761	0.09743	-3.23064

Table 5.1: The fitting constants given by Amphlett and Denuault [1] for a selection of values of RG for an SECM approach to an insulating substrate. They all are quoted as being at worst within 1% of the simulated finite difference values in the range $0.4 \leq L \leq 20$. (The peculiar values of RG are a practical illustration of the geometric inflexibility of finite difference.)

RG	k_1	k_2	k_3	k_4
10.2	0.72627	0.76651	0.26015	-1.41332
5.1	0.72035	0.75128	0.26651	-1.62091
1.51	0.63349	0.67476	0.36509	-1.42897

Table 5.2: Some of the fitting constants given by Amphlett and Denuault [1] for a conducting substrate approach curve. They all are quoted as being at worst within 0.5% of the simulated finite difference values in the range $0.1 \leq L \leq 20$.

example mesh generated with $r_{max} = z_{max} = 500, L = 5$, shown in Figure 5.2, illustrates how the adaptive meshing handles large domains containing relatively small features. The elongated elements are a cause for concern—see below.

The simulated insulating substrate approach curves for our values of RG are shown in Figures 5.3 to 5.7, along with the approximate expressions from (5.1). The matches are generally as good as we could expect from a 1% error tolerance, except at the extreme tip position of $L = 1$ in two cases. For $RG = 10.2, L = 1$ the error creeps up to 1.09%. More disturbing is the 4.19% error for $RG = 100, L = 1$.

While it would be comforting to ascribe these discrepancies to a fault in the finite difference simulation, the more likely explanation would appear to be the extremely elongated elements caused by having a very long, narrow channel between the SECM tip and the substrate (Figure 5.2 shows a less pronounced example): it has an aspect ratio of 100 in the $RG = 100$ case. Ideally the meshing algorithm would divide this up into nicely shaped elements, but it evidently fails to do a good enough job in this case, and the SPR-based error estimator consequently underestimates the error. A better meshing algorithm, as described in Chapters 3 and 6 would probably fix this.

In the meantime, one must rely on visual inspection of the mesh and checking for convergence with decreasing prescribed error tolerance in suspicious cases (essentially, those in domains exhibiting high aspect ratios). If, for instance, we decrease the error tolerance to 0.2% then we do obtain a current value within 1%. This does not, of course, mean that the error estimator is working as it should; but it does perhaps suggest that with more intelligent element subdivision cases such as these might well not be problematic, as they are now.

The conducting cases simulated caused no failures, and the values shown in Figures 5.8 to 5.10 are all within 1% of the values given by equation (5.2).

Overall, for 160 separate SECM simulations, the error bound fails in two cases, significantly in only one; and neither of these is catastrophic. Given

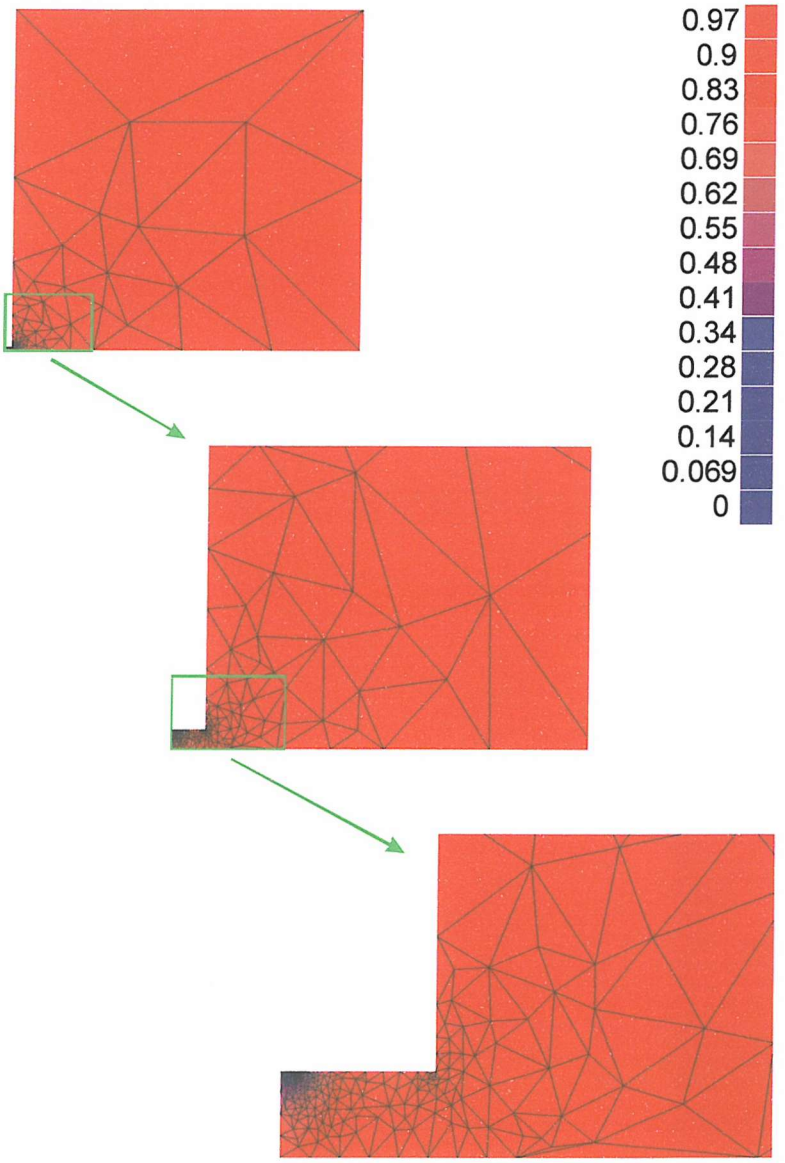

Figure 5.2: The mesh and concentration field for an insulating substrate simulation with $r_{max} = z_{max} = 500, L = 5, RG = 10.2$ and 1% current error tolerance, shown at three different magnifications. The large domain size relative to the tip does not cause too many problems efficiency-wise, with very large elements used in the bulk of the domain, but notice in the last picture the very elongated elements near the bottom. These did not affect the convergence, but see the discussion for their impact on accuracy.

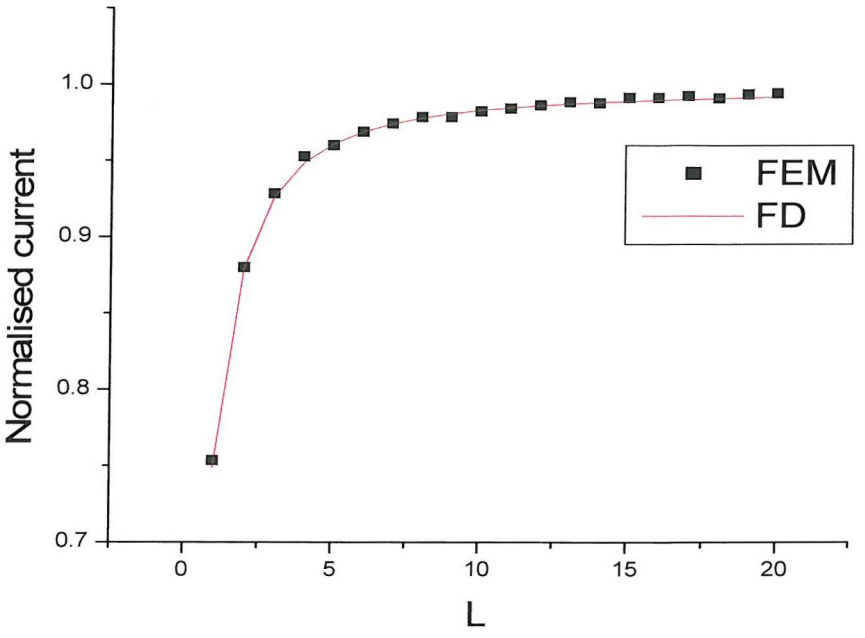

Figure 5.3: The insulating substrate SECM approach curve for $RG = 1.51$ (marked "FEM"). The normalisation value, from the tip at $L = 100$, was 1.16524. In this figure, and those following, is shown the approximate expression deriving from (5.1) or (5.2), marked as "FD".

the pre-existing experimental endorsement of the curve-fitted finite difference expressions, this is a relief. Broadly speaking, we have a reliable starting point for further investigations of SECM using our simulations. We must keep in mind, though, that the high aspect ratios caused by very short tip distances and large insulating surrounds can cause the error estimator to fail. Clearly this problem is likely to manifest itself in other contexts.

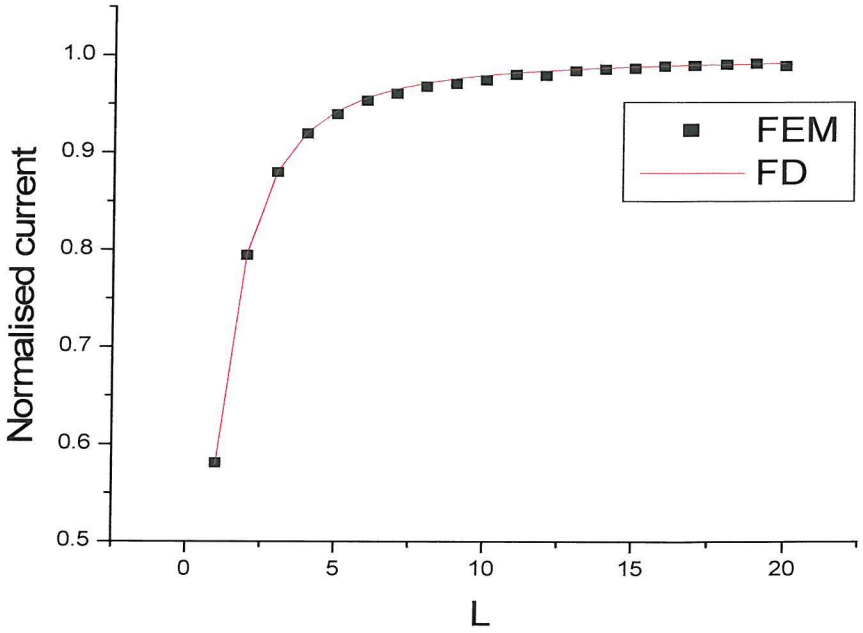

Figure 5.4: The insulating substrate SECM approach curve for $RG = 5.09$. The normalisation value was 1.04626.

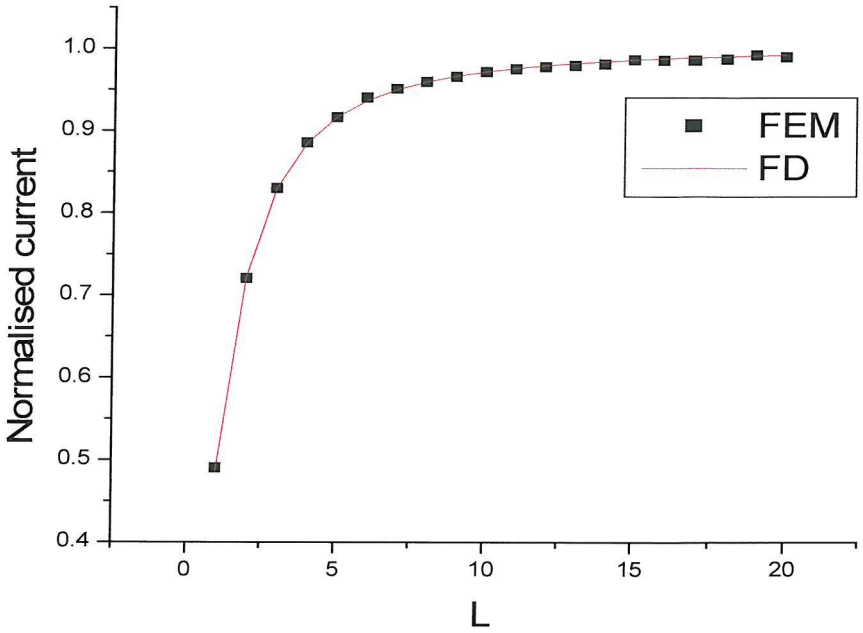

Figure 5.5: The insulating substrate SECM approach curve for RG = 10.2. The normalisation value was 1.0216.

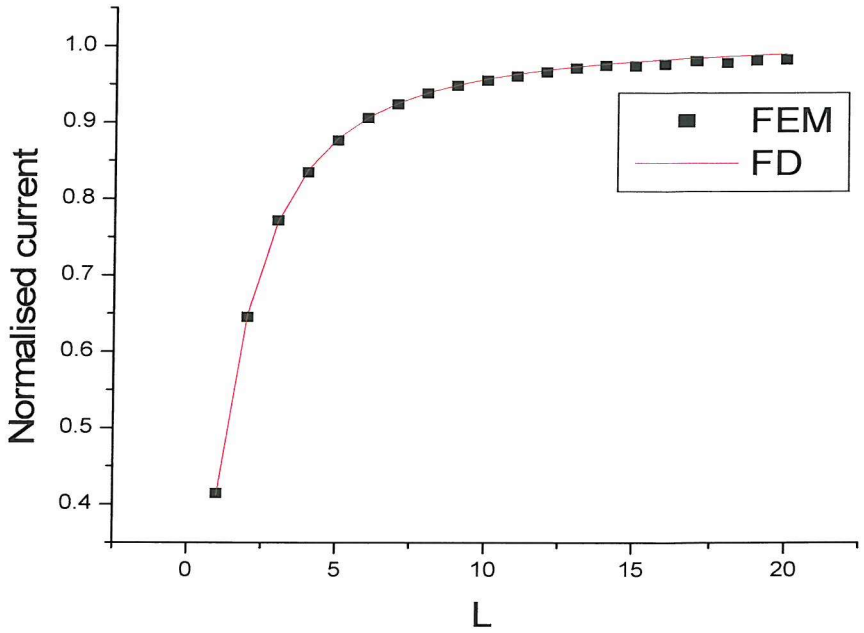

Figure 5.6: The insulating substrate SECM approach curve for $RG = 20.1$. The normalisation value was 1.01747.

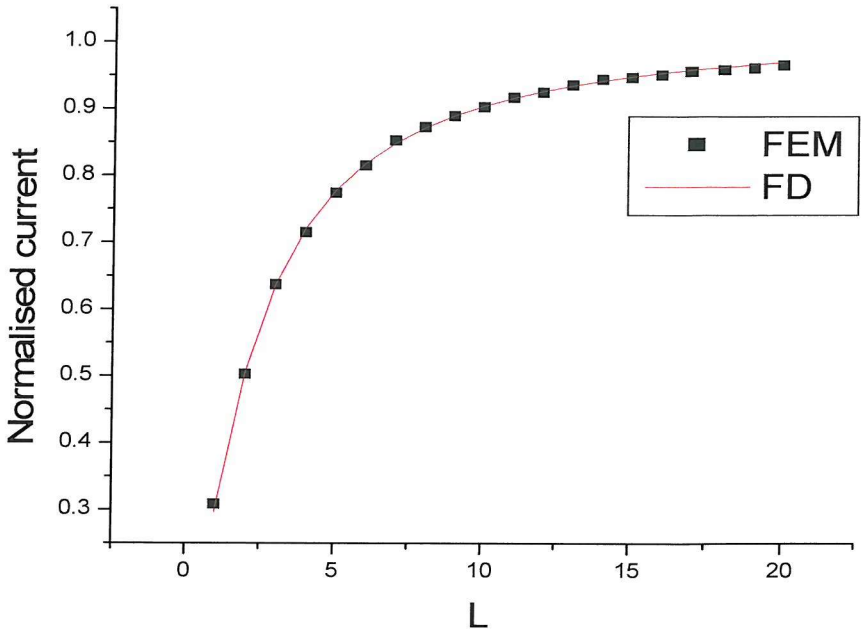

Figure 5.7: The insulating substrate SECM approach curve for $RG = 100$. The normalisation value was 1.00369.

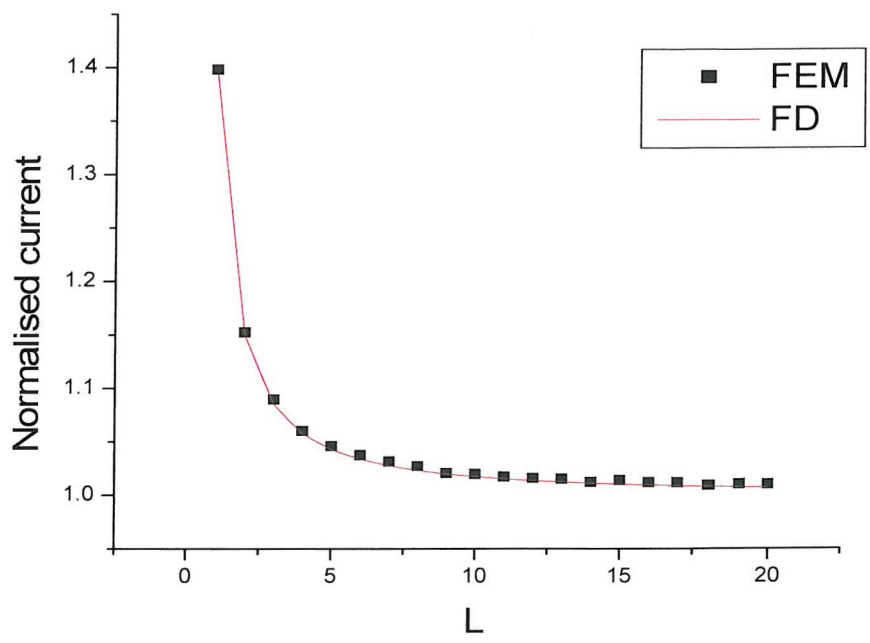

Figure 5.8: The conducting substrate approach curve for $RG = 1.51$ (marked "FEM"). The normalisation value was 1.16813.

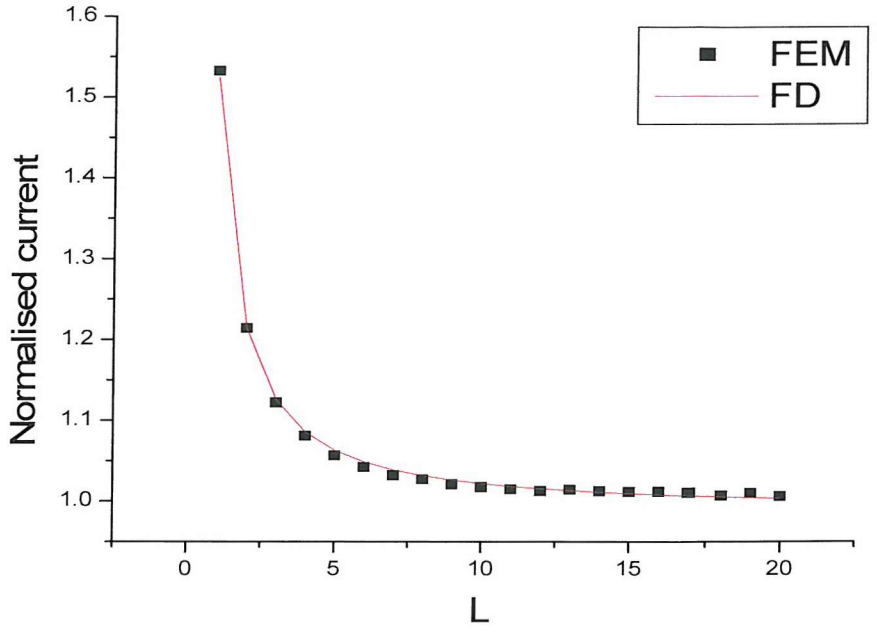

Figure 5.9: The conducting substrate approach curve for $RG = 5.1$ (marked "FEM"). The normalisation value was 1.04862.

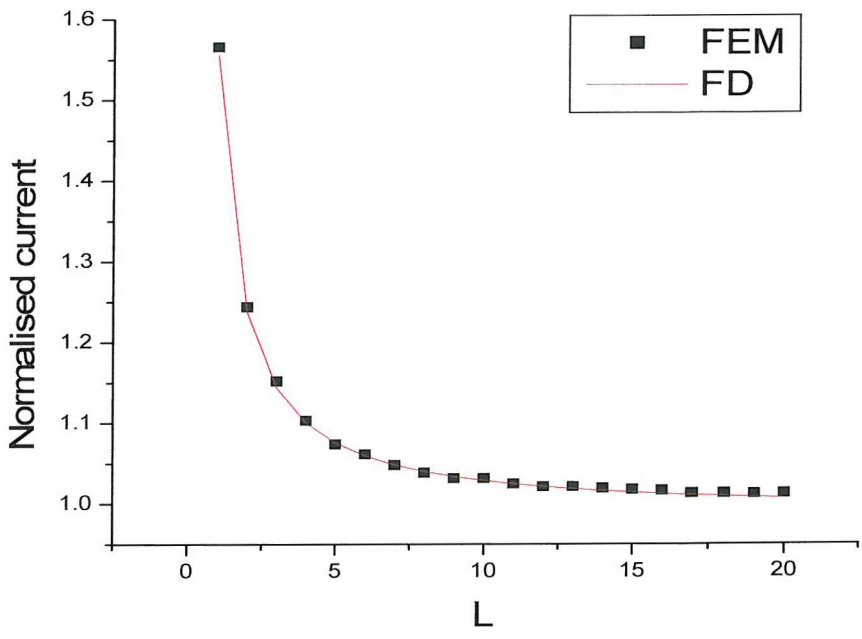

Figure 5.10: The conducting substrate approach curve for $RG = 10.2$ (marked "FEM"). The normalisation value was 1.02397.

5.2.3 Conclusion

The tests of microdisc approach curves with a cylindrical sheath are a further case where the simulation program is generally successful. The deficiencies at $L = 1$ are worrying, but for $RG \leq 10$, to which we now turn, they do not lead us to expect significantly incorrect results.

The fact that comparison with ADI finite difference simulation is available, of course, does tell us that our approach is not necessary for this particular case; but of course modifications can now be made easily to the domain and mechanism in order to conduct more interesting simulations. Since tapered surrounds have apparently not been studied before, we examine next the effect of this alteration to the domain geometry.

Regarding simulation of SECM in general, it is perhaps necessary to address the views of Mirkin [158], who advocates a commercial package called "PDEase2" [160]. This is a general PDE solving program which uses adaptive finite element. In [158] it is demonstrated that, for instance, microdiscs and SECM can be simulated using it. While it has capabilities beyond those of the program developed in this work (notably it can perform transient simulations), it is nonetheless less capable in the key areas focused on in this work. Most importantly, it would appear to lack a) domain integral current calculation and b) current controlled adaptivity (at least, if it has them, they are not mentioned in [158]). The meshing would also appear to be rather crude.

Instead of a percentage current error tolerance, an arbitrary accuracy parameter, presumably in some generic error norm, is specified. Thus, for the inlaid microdisc case, we find the specified error tolerance is 5×10^{-7} in order to yield a current accuracy of 0.5%. Mirkin suggests that "[f]inding a proper value of [the accuracy parameter]... requires some experimentation". The aim of this work has been to obviate, as far as possible, such experimentation.

No idea is given of the simulation time or the number of nodes, but from inspection it would seem the node count is in the thousands. Since the ele-

ment order is not specified, it is not easy to compare this performance with our simulations. It may be comparable if linear elements were used, but of course one would question why they were, given the far greater efficiency of quadratic ones. In the SECM case, 8659 elements were apparently used to simulate a typical case, but to an unspecified accuracy (comparable simulations with our program used at most around 1000 quadratic elements to achieve 0.5% accuracy).

In summary, the lack of information on the "PDEase2" program simulations makes it impossible to conduct a true comparison. It is fair to say, however, that it is, at least with the settings used in [158], around an order of magnitude less efficient with regard to element count (with any but a multigrid solver this implies an even greater difference in simulation time, asymptotically speaking) and incapable of refining to a preordained accuracy in the current. There seems little reason to use it for the cases covered by our program.

5.3 The Effect of Tapering

It does not seem that any simulations are reported in the literature of tapered SECM sheaths, in spite of the prevalence of this configuration in actual experiments. Clearly the case requires a simulation program capable of meshing regions without axially aligned boundaries, which prevents the simulation method favoured so far in electrochemistry, finite difference, from being easily employed. The modification to the simulation input file for our program however, is trivial, and there is no particular reason to believe that the results would be any less accurate having made it.

Clearly, since cylindrical geometry simulations agree with experiment, that approximation must be adequate for the cases where it has been tested. On the other hand, those simulations neglecting diffusion around the sheath edge are known to be deficient. Since these represent tapering angles of 0

and $\pi/2$ respectively (see Figure 5.1), one would expect to find a value of α in between these two extremes beyond which it affects the tip current to a noticeable degree. Since there is no difference between simulations with $\alpha = 0$ and $\alpha = \pi/2$ for large values of RG, we expect the dependence on α to only manifest itself for small RG; further, its influence must be greater the smaller RG is. We address here only the insulating substrate configuration where the effect is assumed to be more pronounced.

Results for four values of RG are shown in Figures 5.11 to 5.14, where the tip distance has been held constant at $L = 10$, and the tapering angle varied between 0 and $\pi/2$. The current has been normalised by the $\alpha = 0$ value at $L = 100$. An accuracy of 0.5% was selected in order to be sure of capturing the relatively subtle effects being probed.

A further illustration of the effect of tapering is given in Figures 5.15 to 5.17, where approach curves for the lower three values of RG are shown for the more experimentally plausible end of the α range. Evidently any tapering effect would be confined to the smallest possible values of RG. The tolerance for these curves was 1%, as detail beyond this level is hard to perceive with the scale used.

It is seen that the effect of α becomes more pronounced the smaller RG is, as expected, but in all cases the effect increases sharply towards $\alpha = \pi/2$. Perhaps the most interesting conclusion from these simulations, if they are to be believed, is that for the smallest sheath radii one anticipates a noticeable tapering effect, depending on experimental accuracy, even for small angles. For instance, with $RG = 1.5$, a tapering angle of 20° is predicted to alter the current by more than 1%, and an angle of 30° should reduce the current by over 2%. Unfortunately data on actual tapering angles used is apparently unavailable, so it is difficult to know if these results have a bearing on experiments that have been conducted, or whether they simply verify that for the combinations of α and RG used tapering may be neglected (as it clearly can be in most instances). Nonetheless, since the resolution of SECM imaging is known to be increased by the use of small sheath radii [1], it could be that

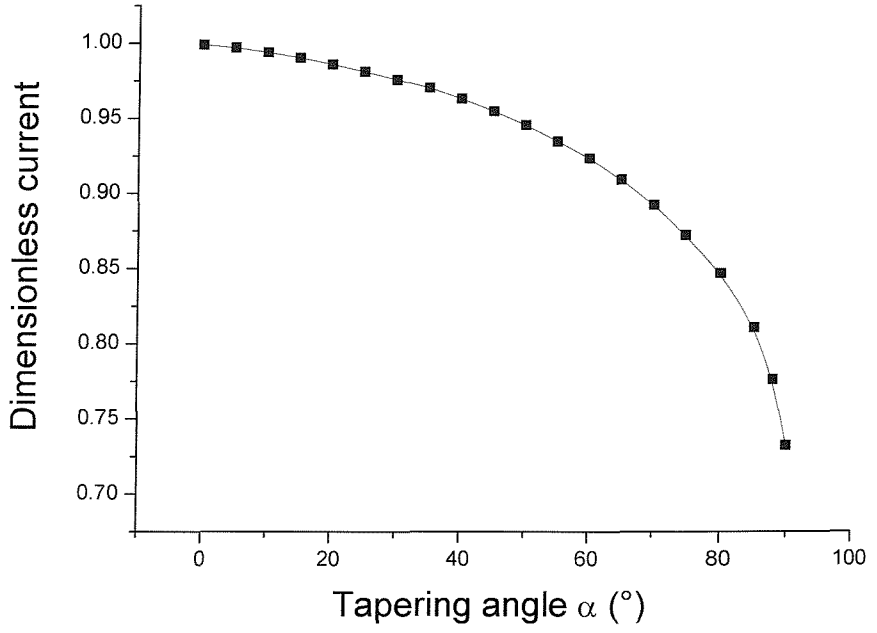

Figure 5.11: The effect on the dimensionless tip current (normalised by the value at $\alpha = 0$, which was 1.14407) of sheath tapering angle, for $RG = 1.5$. The tip was held at $L = 10$ while α was varied between 0 and $\pi/2$.

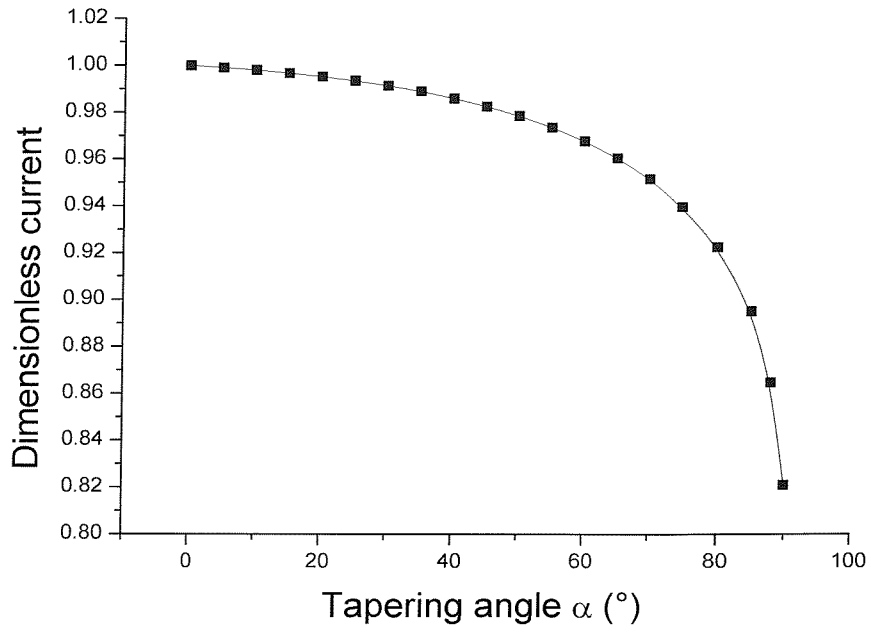

Figure 5.12: The effect on the dimensionless tip current (normalised by the value at $\alpha = 0$, which was 1.01982) of sheath tapering angle, for $RG = 5$.

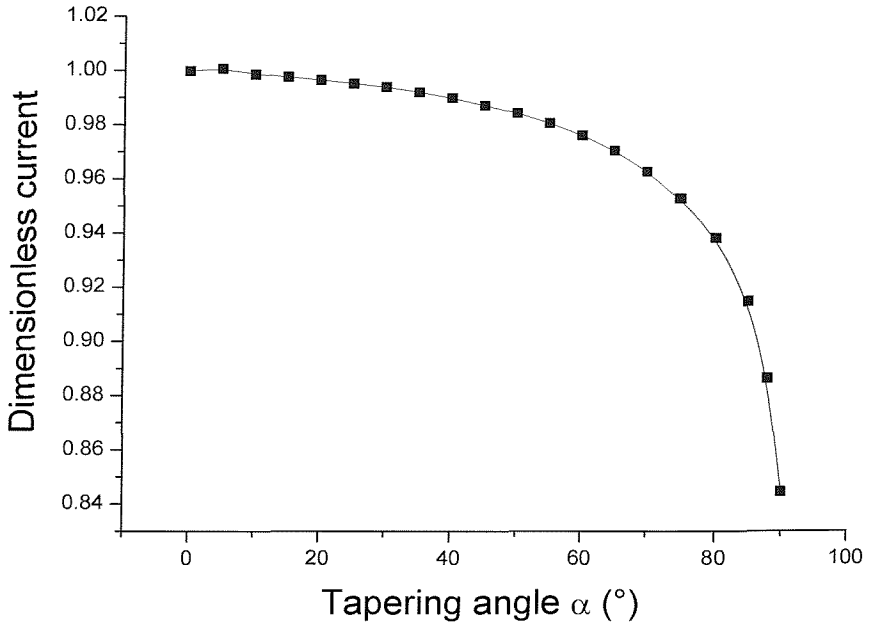

Figure 5.13: The effect on the dimensionless tip current (normalised by the value at $\alpha = 0$, which was 0.991768) of sheath tapering angle, for $RG = 10$. (Although the curve is not monotonic, the deviation from monotonicity, assuming the other points are correct, is within the error tolerance of 0.5%.)

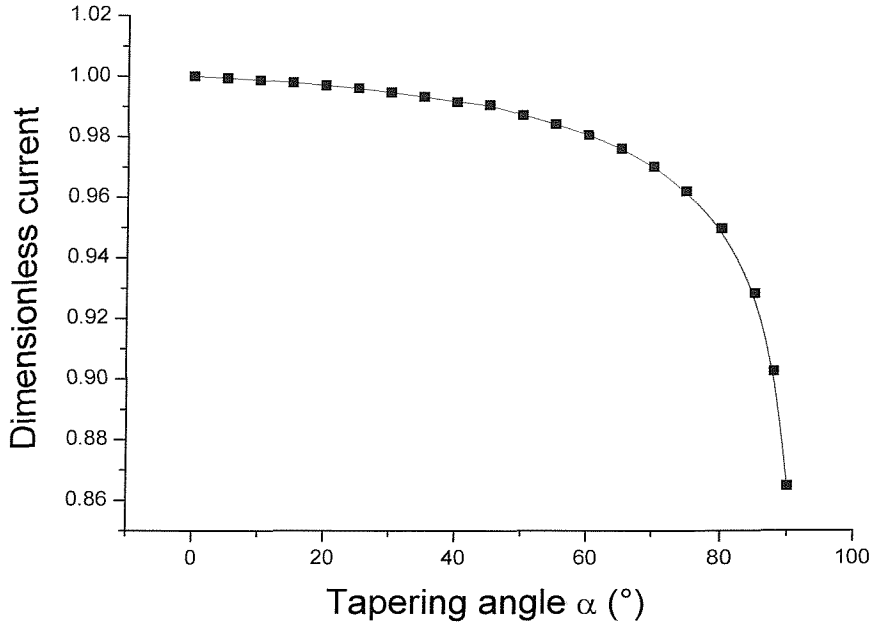

Figure 5.14: The effect on the dimensionless tip current (normalised by the value at $\alpha = 0$, which was 0.969356) of sheath tapering angle, for $RG = 20$.

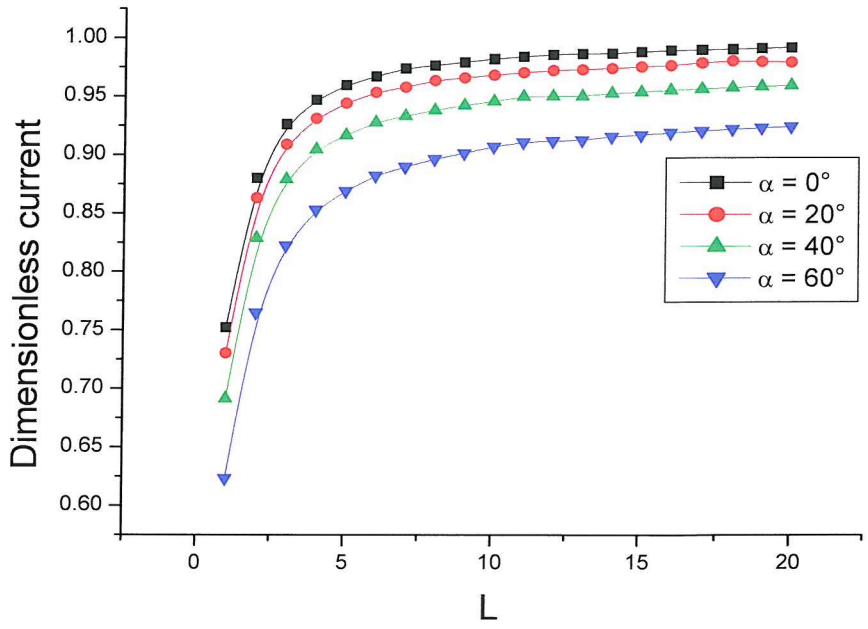

Figure 5.15: Approach curves of a tip with $RG = 1.5$ and various values of α to an insulating substrate.

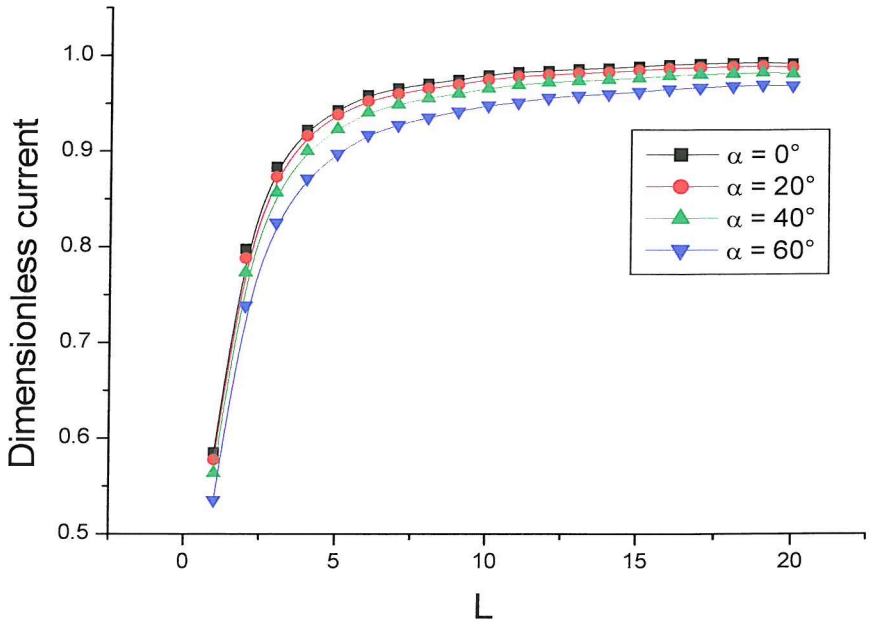

Figure 5.16: Approach curves of a tip with $RG = 5$ and various values of α to an insulating substrate.

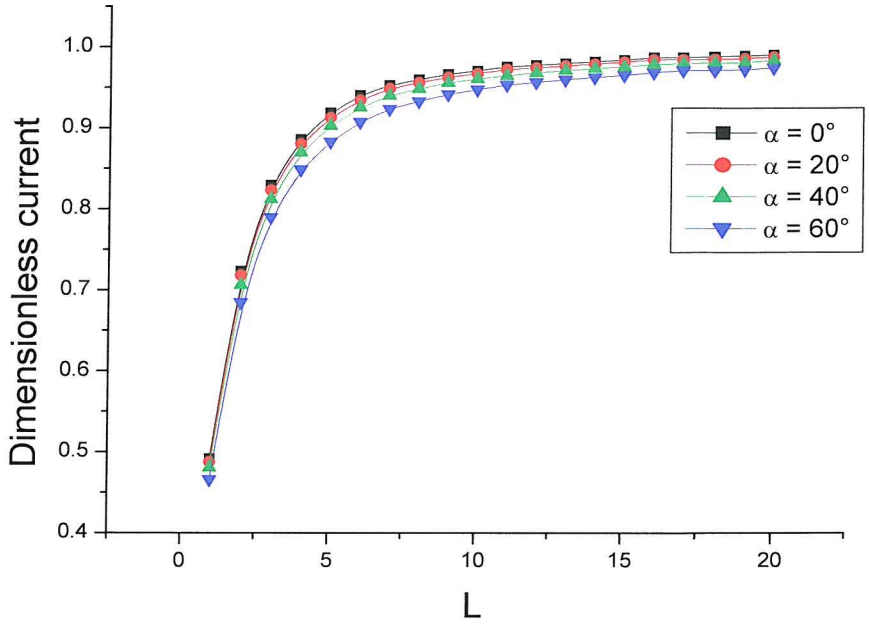

Figure 5.17: Approach curves of a tip with $RG = 10$ and various values of α to an insulating substrate.

RG	α	k_1	k_2	k_3	k_4
1.5	20°	0.65721	0.59318	0.35211	-1.08681
1.5	40°	0.69525	0.64534	0.33397	-1.14945
5	40°	0.62601	1.11231	0.37998	-2.42286
10	40°	0.63049	1.43748	0.37379	-3.40603

Table 5.3: Fitting constants for equation (5.1) for some tapered sheath insulating substrate approach curves.

the effect of tapering becomes more important. In that case a more thorough investigation of its effects would probably be advisable.

In lieu of that investigation, and in the knowledge that tapering may never be of concern to SECM practitioners, some fitted curves are given for the few experimentally conceivable cases simulated where the experimental current is predicted to deviate to an observable extent (taken here to be 1%) from the cylindrical sheath approximation. The form is the same as equation (5.1), and the parameters are given in Table 5.3. The fitting error is substantially less than the simulation tolerance, so the accuracy is, assuming the tolerance is achieved, 0.5% over the range $1 \leq L \leq 20$ (at least for the integer values of L simulated).

5.4 Conclusion

Unfortunately there has not been time to extend the simulation of SECM systems beyond the results given in this chapter. The results so far, though, are relatively encouraging. Where other reliable simulation data is available the program generally produces results differing from it by less than the prescribed tolerance. (The case of highly elongated domains, however, clearly requires addressing.) The results from simulation of the novel case of a tapered domain are in accordance with qualitative expectations.

Overall, SECM has been shown to be a viable arena in which to use the

adaptive finite element algorithm developed in this work.

Chapter 6

Conclusions

Although no claims are made of universal applicability, the results of Chapters 4 and 5 indicate that finite element in some form similar to that presented in this thesis has a future as an important part of electrochemical simulation. The particular algorithm described has some attractive features, the most important of which is automated generation of a mesh designed to meet the requirements of whichever problem is being solved. A program that does this is clearly a major leap forward over *a priori* meshing, whether over the domain for non-adaptive finite difference or element techniques, or on the boundary for boundary element. The methodology espoused herein, then, seems to be an inevitable part of the future of electrochemical simulation. In this chapter a summary of the particular theoretical novelties of this work is given.

The specific use of finite element has been argued for in Chapter 1, and in the light of our findings these contentions still hold. Finite difference simply lacks the geometric flexibility to model many two dimensional geometries, let alone those in three dimensions. In any case it can be be incorporated into the finite element framework as a special case.

The status of boundary element in future simulation work remains undefined. As we have already said, the basic form of boundary element, incapable of handling temporal or convective terms, seems to have little to recommend

it in a world of complex mass transport and transient effects. The dual reciprocity method, or other similar modifications, have the potential on paper to handle any conceivable situation, but they remain largely unproven, particularly for convective transport schemes. If they can be shown to work in the full range of realistic scenarios then the next step would be the development of an adaptive scheme, capable of refinement to achieve a specified current accuracy. This is not, at the present time, a well-researched area.

Assuming then that our approach has a future, it clearly needs improvements, both to allow it to solve simpler problems more efficiently and reliably, and in order to encompass a broader range of experimental conditions. Some of these have been alluded to already. For instance, the more sophisticated Ruppert [101] meshing procedure mentioned in Chapter 3 could, and probably should, be adopted. There are some computational issues related to this, which we do not describe here (clearly none is insurmountable, as the Ruppert algorithm was at one point a part of our simulation program). A related notion is of adopting an entirely different, e.g. quadtree-based, meshing strategy. This could be anything from preferable to essential in transient cases, where we have additional requirements, mentioned below in § 6.3.2 on page 240. Secondly, modification to allow simulation of multiple coupled fields would allow arbitrary first order mechanisms. This would seem a very worthwhile investment of effort, and the theory is a straightforward extension of that in Chapters 2 and 3.

Even with arbitrary first order mechanisms, the program would not handle quite a large number of realistic situations—notably transient ones—and we outline the extensions necessary to incorporate them in § 6.3 on page 238. On top of these, as geometric and mechanistic complexity grows, the system will probably need to be made faster. We discuss methods that could be used to achieve this in § 6.2 on page 232. Of course, if the program with these features is to be generally useful, it must be usable by non-specialists in the simulation field, and the less technical modifications required to achieve this clearly warrant mention; we sketch some of them in § 6.4 on page 248.

On an entirely different tack from the above future directions, the penultimate section addresses the possibility of error bounds on currents with a greater claim to the term "guaranteed"—an approach that might give a different order of confidence in simulation results.

6.1 Summary of Theoretical Advances

From a theoretical perspective the most important new aspect of this work is the use of Zienkiewicz's idea of a higher-order/lower-order solution comparison in tandem with Babuška and Miller influence function. The combination of lower order field interpolation with gradient patch recovery in a hybrid two-order method has also not been reported before, and it expands the range of Zienkiewicz's and Zhu's [125, 136–138] basic means of estimating error. While Harriman *et al.* [34–38] and Rannacher *et al.* [119], have used dual-based residual error estimation for functional evaluation, the more straightforward two-order comparison has not been used for this purpose (unless, perhaps, one counts the energy norm in engineering application).

In principle a wide variety of linear functionals expressible as domain integrals could be evaluated within estimated bounds using the new technique of this work, without any further mathematics. Given a domain-integral in need of accurate estimation a very simple algorithm relates this to the error estimator. A more advanced version of simulation program could incorporate this functionality to simplify the input file format. If an analogous algorithm exists for the residual-based tack of Harriman *et al.* it is not apparent, and certainly far more complicated.

Because the error estimation strategy does not hinge on the influence function being the dual of the primal solution it should allow a study to be made to determine the best form of the influence function. Unfortunately this has not yet been carried out, but the ideas in this work do offer some interesting possibilities for numerical experimentation. With previous esti-

mations strategies this has not been possible.

It has been noted that the meshing algorithm used to produce the results presented earlier was not ideal. Nonetheless, it does represent the most advanced used in electrochemistry so far, and generally functioned well. With some relatively minor tweaking to produce Ruppert's full algorithm it would almost certainly be more effective. Both variants seem more applicable to general problems than the simpler refinement algorithms used by previous goal-based adaptive refinement practitioners.

Finally, the implementation of expression parsing, while not novel generally in finite element, is apparently a step forward in goal-based adaptive finite element, where it is especially important. It allows the rapid employment of different error metrics, and if the program were expanded, notably to a truly general set of coupled equations, it might obviate programming for researchers wishing to use these methods.

6.2 Optimisations

We make the assumption that the basic error estimation theory presented in Chapter 3 is good enough to guide refinement to produce relatively efficient meshes. Clearly it is not perfect, or the error estimator effectivity index would be unity. There appears to be no obvious optimisation for this, although it is certainly conceivable that in the light of new theory, something better is devised. The difficulty with any modification is that a more accurate estimator (in the sense of avoiding over-refinement) is by definition less conservative, and is clearly more likely to underestimate the error, giving incorrect results—an outcome worse than inefficient production of the correct numbers. We have, in any case, already improved on the previous best effort in electrochemistry [35] by a reasonable margin in all cases except the channel flow microband at high flow rates, as documented in Chapter 4. Using the same element error estimator as is to guide refinement, however, a

number of possible speed improvements could be made.

For large problems—by this we mean systems generating large linear systems, whether through difficult geometry or mechanism—the linear system solver inevitably becomes the rate limiting factor; it rapidly dominates simulation times. There appear to be two primary algorithmic means of addressing this concern, ignoring for the moment non-linear mechanisms: better preconditioning or multigrid. These are somewhat interconnected, inasmuch as multigrid can be used as a preconditioner. A third path, always present, is machine level optimisation.

Another line of attack is to use higher order elements, which might allow the use of fewer DOF to attain the same accuracy. The great speed increase of quadratic elements over linear elements for all simulations run gives cause to wonder if, say, cubic elements could yield even faster simulations. We describe the fairly minimal changes these would require. A related idea, of using *hp* adaptive refinement, possibly in combination with an alternative finite element formulation, is also considered.

6.2.1 Preconditioning and Multigrid

The advantages of preconditioning are well known, and are discussed in Appendix D. So far we have implemented only two simple preconditioners: Jacobi and Symmetric Gauss Seidel. These carry the advantage of a near-zero processing cost per iteration, meaning virtually any reduction in the number of iterations is advantageous. They also cannot fail, in the sense that the preconditioning matrix always exists. Unfortunately they generally yield a relatively modest reduction in iteration count, and there are alternatives that are generally perceived to be more effective.

Implementing Incomplete Cholesky Factorisation—a common general purpose preconditioner—would not seem to be a particular challenge. It can involve a greater cost per iteration, and it necessitates more involved calculations before solution begins, but it is generally judged to reduce significantly

solution time. A primary difficulty, and one of the reasons for the great number of variants thereof [72], is that the incomplete factorisation need not exist. This would necessitate careful testing, and it is difficult to see how it could form part of a reliable purely automated simulation scheme. Nonetheless, incomplete Cholesky factorisation is probably the most common generic preconditioner used.

Two more problem-specific approaches are also being widely researched: domain decomposition and multigrid. The former, roughly speaking, breaks the problem geometry into smaller pieces, and solves them separately as a prelude to the entire problem. The latter uses an iterative solver (sometimes even the simple Gauss-Seidel solver, similar to the SOR solver mentioned in Appendix C—see [89, 90]) at a variety of levels of detail, based on the discovery that so-called smoothing procedures such as these are very and only effective at eliminating error components on the scale at which they are applied. In terms of implementation, multigrid requires a set of meshes at different levels of resolution, and moves up and down them to solve the linear system. This can be done either as a complete solver, or as a preconditioner. The mathematical details are given by Braess [32]. Computationally it is fairly involved, as it requires some form of hierarchical mesh storage.

Both of the techniques mentioned above have the potential to be considerably faster than the iterative solvers currently implemented, but they are very much active areas of research, and no one algorithm appears distinctly preferable. The advantages of multigrid seem to be particularly attractive for convective systems, where loss of matrix symmetry makes Krylov space methods less attractive. For non-linear mechanisms, on the other hand, Krylov solvers might have the advantage, given the possibility of adding additional terms to the minimisation process, thereby obviating additional Newton iterations. (This latter extension is discussed in § 6.3.4 on page 244.) All of this would require further research before more definite statements could be made.

6.2.2 Machine Level Optimisation

Aside from parallel implementations, which while bringing large potential speed gains (given the right hardware) also carry with them major technical challenges, a number of optimisations could be made even on serial machines. These incorporate a very limited form of parallel processing capability, generally termed *vector processing.*

The designers of modern processors, recognising the inherently repetitive nature of many numerical computations, have built into most of them capabilities to perform multiple arithmetic operations with a single instruction (sometimes called SIMD, for Single Instruction Multiple Data [83]), effectively in parallel (although the distribution of the processing among execution units can be complex). The present simulation program, however, is entirely written in C++, and is compiled using a compiler which generally does not use these special instructions.

The Pentium III processor, on which all of our simulations were run, has both integer and floating point SIMD capability. Clearly the former is irrelevant. Unfortunately, so is the latter for most scientific computing purposes, as it is confined to single precision. It would not be suitable for a Gaussian elimination solver, for instance, where numerical round-off error is a large concern [69]. It is conceivable, however, and limited tests have not contradicted this, that single precision might be adequate for the iterative solvers in the simulation program. In any case, on more advanced processors double precision SIMD instructions exist. Exploitation of particular machine capabilities would seem worthwhile, as the capabilities are widespread among desktop computers, and optimisation of a few key inner loops would probably yield noticeable speed improvements.

6.2.3 Higher Order Elements and *hp* Adaptive Refinement

Used globally, higher order elements in themselves do not in general present any significant challenge, this being one of the great advantages of finite element over finite difference. The shape functions are well documented [16], and formulae exist for generation of any order of element. Assuming the asymptotic orders of convergence expected in the absence of singularities, we would anticipate improved rates of convergence. On the other hand, there is clearly a limit to the advantage of higher order elements, as they do not allow such flexible distribution of DOF around the domain as lower order ones. With them nodes are also more tightly coupled, and the system matrix tends to have a higher condition number. There is presumably an optimal element order, and it may not be quadratic, but it is not necessarily the highest that can be implemented. The prevalence of lower order element simulations is testament to this.

In addition to the above points, two issues are worth raising:

Firstly, there are apparently no evaluations of superconvergent gradient refinement for very high element orders in the literature, and the superconvergent point locations are not explicitly given. There is some systematic theory governing these locations [16], but the difficulties with degenerate least squares problems noted with quadratic elements would increase. This is one difficulty with using arbitrary element orders, as demanded by *hp* adaptive methods.

Secondly, the condition number problems of higher order elements can be somewhat alleviated by use of hierarchical elements, which give the same results, but better conditioned linear systems (see *ibid.*). These would seem to be worth using regardless of element order, but apparently their advantages are not as pronounced with transient simulations, at least in mechanics applications (again, see *ibid.*).

As was noted in Chapter 3, p refinement has a theoretical rate of conver-

gence with increasing DOF potentially far greater than that of h refinement. Since p refinement is not generally regarded as alone suitable for adaptive finite element, much effort has been expended by researchers on a hybrid approach, combining the geometric flexibility of h refinement with the greater rate of convergence of p refinement, to form hp refinement. Simple ideas, such as performing h refinement to a certain error level, say 5%, and then increasing the element order to achieve 1% are described by Zienkiewicz and Taylor, but many seek a more general approach, where elements in the same mesh have different orders. It has been shown that this actually allows an exponential rate of convergence even in the presence of singularities, effectively by cordoning these off with h refinement while refining smooth areas away from singularities with p refinement—see e.g. [161].

Various difficulties exist with incorporating varied element orders in a mesh. In particular, satisfying the continuity requirements of the Galerkin formulation is a problem. Consider a quadratic triangle abutting a linear one. One element would have a node on their shared edge, whereas the other would not; and yet for the C^0 continuity demanded by the formulation they would have to have the same field value all along the edge.

Difficulties like these have driven theorists to alternatives with looser inter-element continuity requirements. In particular, the Discontinuous Galerkin method [74, 162] appears to be gaining popularity. This enforces continuity in a weak sense, and is thus well-suited to hp adaptive meshing. It also carries the advantage of apparently handling high flow rate convection without stabilisation techniques. All is not perfect, however, and one major downside, at least where using Krylov space solvers, is the loss of linear system symmetry, regardless of convective terms, with some versions of the formulation. More DOF are also needed for the same number of elements. It is unclear how the performance tradeoffs sum up, but this technique, or some other modification like it, might prove useful in the future.

6.3 Future Extensions

Numerous experimental conditions are not at present handled by the simulation program, varying in their difficulty of implementation. Unfortunately in many cases the changes necessary for expanding the scope of our program frequently impact on other decisions—for instance, convection is incompatible with a non-linear conjugate gradient solver. It is not possible to consider every permutation even theoretically, quite apart from performance in real situations. That said, we attempt to note any particularly salient interplay.

6.3.1 Infinite Domains

Perhaps the least elegant part of the current approach is the need in new cases to show convergence with expanding domain size. With automated mesh generation providing a theoretical bound on the error, the only human intervention in the solution process is this convergence testing. It would be desirable to eliminate this.[1]

Infinite electrochemical domains have so far only been simulated using boundary element, which has particular advantages for this. One means of extending finite element to problems of infinite geometry is to use this capability by coupling finite element to boundary element in a hybrid scheme. This might be one way of proceeding, but Zienkiewicz and Taylor [16] are quite categoric in their advocacy of an alternative, infinite elements, which they describe as "without doubt, the most effective and efficient treatment". They are also probably the simplest, so they form an attractive option.

The details of infinite elements are given in the finite element literature (see *ibid.* and references therein). Briefly, they operate by mapping a finite local coordinate space, similar to the local triangular coordinate system described in Chapter 2, to an infinite expanse in the domain. The field within

[1]The other case of geometric approximation, curved boundaries, is of course similar, but is well served by finite element texts, where curved elements are described in detail.

this space is then represented by functions asymptotically approaching the desired value (e.g. functions polynomial in $1/r$, where r is a measure of the distance from some part of the finite domain). We concentrate here on how they might be fitted into the overall adaptive scheme, which is apparently not a subject treated in the literature.

Clearly infinite elements cannot be subdivided geometrically. The h adaptive finite element scheme used for the normal elements therefore cannot be employed. This leaves two options. On the one hand, subdivision could be restricted to the finite part of the domain. This would mean the error estimator domain integral could only extend over those finite elements. Alternatively, a type of p refinement[2] could be implemented for the infinite elements, and the whole domain could be refined. This might seem preferable, but it does not eliminate user intervention; the size of the finite part of the domain would still need to be specified, unless of course this too were adaptively determined, which would introduce further issues.

It would seem therefore that the simplest viable means of using infinite elements would be to accept that the user must specify a finite domain size, encompassing any sharply varying parts of the concentration field, and that infinite elements of a fixed order would provide an asymptotic approach from the edge of that to infinity. Conceptually, this could be viewed as simply another type of boundary condition on the edge of the finite domain, although it would introduce extra degrees of freedom into the linear system behind the scenes.

Without testing, it is impossible to say how workable this approach would be. If it were successful it would not of course completely remove domain size from consideration when determining simulation accuracy. But it would presumably increase efficiency and reduce the impact of domain size, obviating very large domains that place greater demands on the meshing algorithm.

[2]The nomenclature does not seem strictly appropriate, since polynomials of spatial coordinates are not used as the infinite element shape functions, but it appears to be standard.

The implementation would not seem to have any particularly difficult aspects, and could apparently be fitted into the program as it exists fairly easily.

If the proposed scheme did not prove successful, other options could be considered, including p adaptive infinite element refinement and boundary element coupling. Clearly the latter, however, would bring all the attendant limitations of boundary element, and so would negate a large part of the usefulness of our approach: a large finite domain would seem preferable. Perhaps adaptive sizing of the finite domain could be useful, but no mention of such a thing has been seen in the finite element literature.

6.3.2 Transient Simulations

The lack of transient simulation capabilities is by far the greatest practical downside to the simulation program as it stands. The basic finite element theory for this is well-established, but implementing transient adaptive meshing brings with it difficulties. Most difficult of all, it would seem, and most important, is the extension of the current calculation scheme to transient currents.

Harriman *et al.* have followed a heuristic approach to solving this, with some success [140]. Reservations about the complexity and relative inefficiency of their simulations remain, however, and it seems possible that our version would retain some advantages if it could be made to work for transients. It has already been mentioned that the reduced restrictions on the influence function in our case could be an advantage here.

One very simple approach would be to fix on an influence function invariant with time, but this would probably handle singularities caused by initial conditions badly if it were efficient for later times—ideally it would reflect the shape of the primal solution. A better alternative could be to, say, solve the modified Helmholtz equation with varying values of k (not necessarily every time step). This could give a range of influence functions more or less

tightly confined around the electrode. We note that Harriman *et al.* actually solve this equation in their approach to the transient E mechanism microdisc problem, although the details are different.

A number of issues would remain if this primary problem could be solved. The transient error estimator would need to incorporate both the spatial discretisation error, as was done in the steady state case, and the temporal discretisation error. Not only would this guide the shape of each time step's mesh, but, in order to keep the current error below a predefined limit, the temporal step size would need to be adaptive too.

The sort of Taylor series time derivative approximations used in finite element lend themselves easily to constructing two different orders of approximation. The difficulty is that while these provide a *local* error estimate, an error will be accumulated with every step, and we desire to keep this error below our predetermined limit for all steps: we do not have the luxury, as with spatial coordinates, of globally adjusting the mesh over the whole range. If the step size is adaptive, the number of steps cannot be known ahead of time, and so it is difficult to know the acceptable error level for each one. One approach, of course, would be to use space-time finite elements, and solve for the entire time span at once. Unfortunately this is deeply inefficient, since it does not take advantage of the very regular shape of space-time elements, and it would have high memory requirements—this is of course why finite difference schemes tend to be used for FE time stepping.

Leaving aside error estimation in transient simulations, various other questions remain. The most significant concerns the computer implementation of adaptive meshing over time. Staying with a mesh enrichment strategy (see § 3.1 on page 82), there are some obvious problems to be solved. Firstly, some form of derefinement would need to be implemented in order to be assured of efficiency of each successive mesh: areas requiring a fine mesh at one time—for instance in the early stage of a Cottrell-type experiment near the electrode—would not necessarily require so many elements later in the simulation. Harriman *et al.* [34–38] implemented a derefinement strategy in

their steady state simulations. We achieved generally greater efficiency in our steady state simulations without derefinement, but for transients it would be necessary. There exist a number of approaches, documented in the literature, and it seems likely that one of them would fit easily into our overall scheme.

From a programming perspective, the most obvious difficulty with different successive meshes is efficient evaluation of the old mesh's concentration field values on the nodes of the new mesh, which is required for any time-stepping algorithm. This is non-trivial, because the new nodes introduced in a time step will not correspond to the ones in the previous time step, and searching for them within every element of the old mesh would be far too inefficient. Some form of data structure, locating new nodes in the old mesh, would need to be maintained as they were generated. In this context, it is worth noting that mesh regeneration would require even more radical measures to ensure efficiency, as no node in the mesh at $t + \Delta t$ would have a direct correspondence to any in the mesh at t.

About the only obvious part of implementation transient simulations is time derivative approximation, which is well documented. Here some of the issues raised with finite difference in Chapter 1 re-emerge. In order to assure stability, some form of fully or semi-implicit scheme is typically used, which requires solution of a linear system at each step. This is unavoidable, and it stresses linear system solver efficiency, but based on steady state simulation times we would not expect this particular problem to lead in itself to unacceptably long simulation times.

It is important to remember that transient simulation code constitutes a means of solving steady state problems too, simply by extended iteration. In some cases this might even be preferable, for instance where characteristic-based methods (see below) allow reliable solution of diffusive-convective problems. At the very least it could save programming effort.

6.3.3 Three Dimensional Simulations

Although perhaps less important than in many fields, three dimensional simulations would nonetheless be extremely useful in a number of situations. For instance, BEM practitioners [163, 164] have modelled tilted SECM tips in three dimensions, and there are any number of cases where cylindrical symmetry is destroyed by such imperfections.

Nearly every part of mathematical theory discussed in Chapters 2 and 3 generalises easily to three dimensions. The MWR formulation and our error estimation strategy carry over essentially unchanged; the area integrals simply become volume integrals. The addition of transients does not change this. The only alteration is in the type of element used, and tetrahedral three dimensional analogues of triangular quadratic elements are well documented. Instead of three assembly nodes and three edge nodes, they have four of each.

The real difficulty of three dimensional simulation is in the meshing. Crucially, Delaunay meshing, as used here for meshing two dimensional regions, does not generalise conveniently to three dimensions. Firstly, the three dimensional Delaunay triangulation does not necessarily exist, necessitating the addition of new vertices irrespective of other meshing requirements. More fundamentally, the minimum angle property does not generalise either, which brings into question the desirability of using three dimensional Delaunay triangulations in the first place. It means that the tetrahedra in a Delaunay triangulation would not necessarily constitute "good" elements.

Various attempts have been made to fix these problems, still using the Delaunay idea. Other quality constraints can be introduced to try to eliminate bad quality elements [94, 99]. It remains unclear, however, how effective these can be. It may be that an entirely different approach is required, for instance using octrees (described in a graphical context in [165]). These do carry with them the attractive properties of quadtrees, which have been used successfully for meshing in two dimensions [102]. It is not clear how efficient they could be made to be, given the reservations described in Chapter 3,

but they do hold potential benefits for efficient node searching in transient simulations.

An overarching problem in three dimensions, of course, in common with many other extensions to more complex systems, is the greater stress placed on the linear solver. Three dimensional domains will inevitably have far more degrees of freedom, and will therefore generate linear systems harder to solve. In this context, then, the faster linear solvers discussed elsewhere in this chapter might prove necessary for practical three dimensional simulations.

6.3.4 Higher Order Mechanisms

A large number of electrochemical mechanisms encompass second order homogeneous reaction terms. These render the algebraic system resulting from discretisation non-linear, and consequently necessitate a new solver.

Many electrochemical simulations carried out so far have adopted simplified, linearised, models of mechanisms in order to fit them into standard simulation schemes. Unfortunately the results are often only qualitatively congruent with experimental results (see e.g. [122]), and it would be desirable to simulate second order mechanisms properly. Since non-linear problems arise in many applications of the finite element method, a number of means of solving them are in existence.

Harriman *et al.* have simulated the non-linear EC_2E mechanism with current error adaptive finite element [38]. For this they used the multi-dimensional form of Newton's method in combination with a linear system solver. As with the familiar two dimensional version, this requires knowledge of the derivative of the function whose roots are being found, in the multi-dimensional case in the form of the Jacobian [13]. It also requires a starting guess, which Harriman *et al.* do not discuss.

The latter requirement necessitates some problem-specific thought. While Harriman *et al.* do not say this, a good choice for the EC_2E might well be the solution of the linear ECE mechanism. In other words, the various linearised

versions of mechanisms proposed as approximations, as in [122], could form a useful resource for the starting guesses of true non-linear solvers.

A second approach to the non-linear problem for purely diffusive problems is modification of the Conjugate Gradient scheme itself. Since this is an iterative minimisation process (of the linear system's quadratic form—see Appendix D), it is clearly possible to add other terms to the minimised function [30, 166]. In practice this means that one dimensional non-linear root finding must be performed for each conjugate gradient step, which is not generally too challenging to implement. This could prove an efficient way of proceeding, as it would not require two sets of nested multi-dimensional iterations, the one for the non-linear solution, and a further set of linear solver iterations for each one of these. So far this type of solver has not, apparently, been tried in electrochemistry, and it would seem to warrant investigation. That said, of the finite element authors cited in this work, apparently only Braess [32] mentions non-linear conjugate gradient, and most papers appear to favour Newton's method or other multi-dimensional iterations, so it may not be very effective in a finite element context.

Of course, if this approach were tried with non-linear mechanisms combined with convection, one of the generalised iterative solvers described above would have to be combined with a non-linear component, and there is no obvious mention of this in the literature. Clearly the Newton approach is the more easily generalised to convective cases, but it seems at least possible that non-linear Krylov space solvers could be more efficient.

6.3.5 Fluid Dynamics

The applicability of simulations with convective mass transport is limited, as things stand, to systems where a good approximation to the fluid dynamics problem is known for the cell geometry. The channel flow case is one of the rare ones where such a thing exists, essentially because the problem is one dimensional. Unfortunately, analytical solutions of the Navier-Stokes

equations governing hydrodynamics are even rarer than those of the diffusion equation. Experimental flow cell geometry is therefore somewhat constrained at the present time. If the channel flow microband geometry were modified, for instance, to have a recessed or protruding microband, as was simulated with purely diffusional mass transport in Chapter 4, only a crude approximation to the velocity field would be available, and it would be difficult to quantify the current under forced flow conditions.

The problem is not, or at least not principally, one of boundary conditions, as with our mass transport problems, but rather with the equations themselves. Even neglecting heat transfer and coupling between reactions and the fluid dynamics equations, the problem remains complex. For the most part we are concerned with fluids behaving like water—that is, viscous incompressible media. One dimensionless form [147] of the steady state equations in two dimensions is, ignoring external forces such as gravity:

$$\frac{1}{Re}\left(\frac{\partial^2 u_x}{\partial x^2}+\frac{\partial^2 u_x}{\partial y^2}\right)-\frac{\partial p}{\partial x}=u_x\frac{\partial u_x}{\partial x}+u_y\frac{\partial u_x}{\partial y}\,, \tag{6.1}$$

$$\frac{1}{Re}\left(\frac{\partial^2 u_y}{\partial x^2}+\frac{\partial^2 u_y}{\partial y^2}\right)-\frac{\partial p}{\partial y}=u_x\frac{\partial u_y}{\partial x}+u_y\frac{\partial u_y}{\partial y}\,, \tag{6.2}$$

$$\frac{\partial u_x}{\partial x}+\frac{\partial u_y}{\partial y}=0\,. \tag{6.3}$$

Here we must solve for the two velocity field components u_x and u_y, as well as the pressure field p. The dimensionless constant Re is known as the Reynolds number.

Various approximations exist for limiting cases. For very slow viscous flows, for instance, termed Stokes flows, the convection-type terms are deemed negligible, reducing the equations to the same form as those for incompressible solid mechanics. Unfortunately, these themselves are non-trivial to solve as incompressibility greatly complicates the process.

At the other extreme, where viscosity can be entirely neglected, termed Eulerian flow, the equations become purely convective. This is not in itself an advantage, but if the velocity field is irrotational [13], it can be written in

terms of a scalar potential which obeys Laplace's equation. All the machinery developed earlier in this work could then be brought to bear. Unfortunately this is not representative of the type of system under study. For instance, the parabolic flow profile in the channel flow cell is a result of the "no slip" condition imposed on the walls: the fluid is deemed to have zero velocity where in contact with the walls, on account of viscosity. Since electrodes generally reside in this region, viscosity cannot be neglected.

While no one approach appears universally validated, Zienkiewicz and Taylor [74] do suggest that the Characteristic Based Split method is widely applicable. (In any case, concerns such as shock capture in high speed gas flow are hardly relevant to our situation, so some failings are not of concern.) Roughly speaking, this separates the diffusion and convection parts of the equations, and solves them separately.

Since (6.1) and (6.2) are essentially of the diffusion-convection variety, we consider a simplified example of the same nature. The initial idea is that the convective-diffusion equation of the form

$$\frac{\partial u}{\partial t} = D\frac{\partial^2 u}{\partial x^2} - v_x\frac{\partial u}{\partial x} \tag{6.4}$$

can be rewritten in a purely diffusive form if the new coordinate $dx' = dx - v_x dt$ is defined:

$$\frac{\partial u}{\partial t} = D\frac{\partial^2 u}{\partial x'^2} \,. \tag{6.5}$$

Instead of moving the transported substance relative to the coordinate system, we do the converse.

The primary difficulty with this idea is that the coordinates of the mesh nodes would need to be rewritten in terms of the new coordinate, which would lead to ever greater elongation of elements, and all manner of programming problems. The realistic solution, then, is to project the new values back onto the old mesh, and continue as before. This still leaves the problem of non-linear convective terms, but various solutions have been proposed (*ibid.*).

Unfortunately error adaptivity seems far from well developed in this context, and ideally the influence of the fluid dynamics discretisation error on

the current would be considered, but there is the promise of an almost universally applicable simulation system if these problems could be solved.

6.4 Constructing a Generally Useful Simulation System

At present, our simulation program accepts input in a fairly general format, in terms of partial differential equations. This has the advantage of flexibility, but is not ideal for electrochemists unfamiliar with the details of simulation. More useful would be a program capable of handling a number of common geometries, with a number of common mechanisms, ideally with the potential for generalisation by the more technically minded user. The path to this goal, theoretical simulation difficulties notwithstanding, seems fairly clear.

The input files currently have a degree of parameterisation, in that they allow, say, the homogeneous rate constant, or the depth of a microdisc recess, to be specified in a single line, and the implications for the governing equations or the geometry taken into account. This leads to the idea of simulation template files, where a standard input file for, say, the E mechanism with a recessed microdisc, would be stored, and the user would supply the recess depth and the desired current accuracy to get their desired result. This would require a greater degree of interactivity on the part of the program, but boils down to standard graphical user interface programming. This approach has the advantage that the template files could be modified for deeper changes.

For further flexibility, with entirely arbitrary mechanisms and geometry, another level entirely of interface programming would be required. For the mechanism, some more chemist-friendly format than a PDE system would presumably need to be devised, and a program written to convert these to the mathematical description required; similarly for heterogeneous reactions and boundary conditions. Overall, given the parsing challenges already met, this does not seem overly daunting.

With regard to geometry, the difficulty of the task of a cell geometry designing interface would seem to revolve principally around the number of spatial dimensions. For two dimensions, this would amount to drawing a set of lines, and enforcing various constraints to ensure a well-formed boundary. For three dimensions, it would be altogether more complicated, and would entail many of the facilities of CAD design packages. Perhaps in that case some form of import facility from such programs would be a better strategy.

The time dimension would of course require its own treatment. Since the result of the simulation would be an entire curve rather than a single value (or sets thereof), this data would need to be manipulable and capable of export. None of this seems especially difficult.

6.5 "Guaranteed" Current Bounds

Apart from the error estimation theory in Chapter 3, there remains the unexplored possibility, illustrated by Fox [167], of bounding the current from above *and* below, using boundary element to establish the second bound. Obviously all the caveats associated with boundary element would hold, and the method would be restricted in its applicability, but it represents a qualitative jump from the *estimated* bounds that we deal with here, and which formed the basis of Harriman *et al.*'s work.

The idea of establishing upper bounds for quantities of interest using a Galerkin approach is not new, at least outside of electrochemistry. In quantum mechanics it has been used both analytically [168] and numerically [169] to establish upper limits for quantum energy states. FE bounds for strain energies in solid mechanics applications are also not novel [16]. The notion of upper and lower bounds using a combination of FE and BE, however, seems to be less widespread.

Essentially, Fox shows that while the Galerkin finite element approximation yields an upper bound on domain integral current functionals like those

in Chapter 3, the Trefftz boundary element approach gives a *lower* bound for the same quantity. He presents this in an analytical context, and illustrates with manual integration of trial functions. But there seems to be no fundamental reason why this could not be automated.

We have already alluded in Chapter 3 to some of the practical difficulties and limitations for the finite element side of things. Ignoring round-off error, there are still sources of approximation other than the finite element discretisation. Exact integration formula, rather than numerical integration, would be required for the linear system assembly and functional calculation. The iterative linear solver error would also need to be considered. Finally, any geometrical approximations, most relevantly the truncation of an infinite domain, would also cause difficulties. All of these suggest that no guarantee on a bound is ever possible, but certainly greater confidence would be possible where the error bound theory does not rest on the Taylor series arguments so far employed.

Presumably similar difficulties would exist for the boundary element lower bound generation. Exact integration formulae do exist for constant value boundary elements, however. A real difficulty would be systematic generation of functions satisfying the governing PDE, which reminds us of all the downsides of boundary element noted in § 1.3.3 on page 31. It is clear for these reasons, and others, that the method could never be applicable as widely as the algorithm we have used in Chapters 4 and 5.

All this being said, it might still be worth pursuing. Having both a lower and upper bound for a quantity of interest brings real advantages over an error estimate. If hard bounds could be established for currents using this method then disagreeing results could be classified as being within a certain percentage of the true value. A program could be envisaged employing adaptive finite element *and* boundary element in order to progressively bracket the desired quantity within a shrinking range. This would be as robust a means of establishing the current in some genuinely useful cases as could be imagined.

6.6 Summary

In this thesis the aim has been to move towards a more advanced simulation capability in electrochemistry. In this attempt we have introduced a number of new techniques to electrochemical finite element, including superconvergent patch recovery, and our own error estimation strategy. In combination with our choice of meshing algorithm, we believe that, although not covering every case that we wish for, the end product is a step forward.

Superconvergent patch recovery, in one form or another, would seem to be well worth utilising in any electrochemical finite element simulation. It can yield particularly accurate results in combination with a domain integral expression for the current. It should not be thought, however, that it is the only means of estimating the error. A less efficient, although possibly more robust strategy, would be to simply use two different orders of solution. The key idea is to use the widespread notion of different solution order comparison for current error estimation. That said, in our tests the hybrid SPR/twin order error estimation strategy has proven to be considerably more efficient than competitors.

The mode of implementation, we hope, has some advantages regardless of the technical specifics. The idea of supplying simply a cell geometry and a mechanism, and allowing the simulation program to automatically generate a mesh to solve it, is an attractive one. We believe this must be preferable to the *a priori* meshing into which so much effort has hitherto been put. Although Harriman *et al.* never make explicit the capabilities of their meshing algorithm, the implication is that the program used to generate their published results did not have the ability to mesh an arbitrary domain. We see no reason why the more general approach used here should not become the norm.

If it transpires, then, that none of the specific error estimation strategy pursued in this work is of longstanding utility, it seems fair to assume that the simulation methodology adopted here will prove fruitful. The next challenge

is to apply it to a wider span of cases, most particularly transient ones.

Appendix A

Inlaid Microdisc Solution Derivation

The E mechanism current to an inlaid microdisc is an important practical problem in electrochemistry. Fortunately an exact solution has been known for some time. A derivation is given here, the specificity of which is instructive when considering extension to more difficult cases; it makes a case for numerical methods where symmetry is not present.

The method used here—that of oblate spheroidal coordinates—is not the only one available. Carslaw and Jeager [170] employ an entirely different approach, using the Neumann integral theorem (*ibid.*) in combination with an obscure property of Bessel functions. While this may be shorter it seems less intuitively understandable.

As is mentioned below, Saito [171] apparently first derived the result in electrochemistry with essentially the same method as used here. Crank and Furzeland [172] also gave a solution, but with a different form for the concentration field. In this appendix the two are shown to be equivalent.

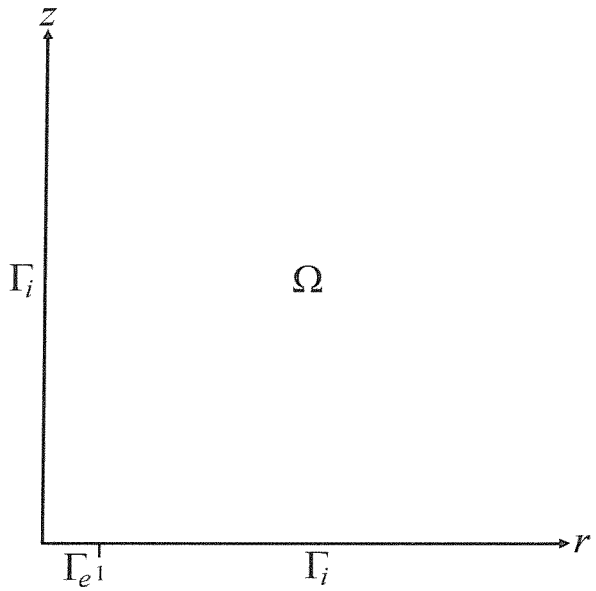

Figure A.1: The inlaid microdisc problem domain. On Γ_i, which comprises both the disc's insulating surround and the symmetry axis, we need $\frac{\partial u}{\partial n} = 0$. On Γ_e, the electrode, we need $u = 0$. We expect the dimensionless concentration u to tend towards unity far away from the electrode. The solution u covers the semi-infinite domain Ω.

The Problem

Adopting the normalisation of 4.1.1 on page 134, the challenge is to solve the Laplace equation,

$$\nabla^2 u = 0 \,, \tag{A.1}$$

with the boundary conditions shown in Figure A.1. Since the domain is cylindrically symmetrical, the equation is usually given in the form:

$$\frac{1}{r}\frac{\partial}{\partial r}\left(r\frac{\partial u}{\partial r}\right) + \frac{\partial^2 u}{\partial z^2} = 0 \,. \tag{A.2}$$

This of course derives from the Laplacian operator in cylindrical coordinates,

$$\nabla^2 u = \frac{1}{r}\frac{\partial}{\partial r}\left(r\frac{\partial u}{\partial r}\right) + \frac{1}{r^2}\frac{\partial^2 u}{\partial \phi^2} + \frac{\partial^2 u}{\partial z^2} \,, \tag{A.3}$$

with the assumption, since the boundary conditions are not angularly dependent, that $\frac{\partial u}{\partial \phi} \equiv 0$. The cylindrical coordinates are related simply to the Cartesian ones by

$$x = r\cos\phi\,, \qquad y = r\sin\phi\,, \qquad z = z\,. \tag{A.4}$$

In spite of the symmetric reduction achieved by assuming angular independence, the equation remains two dimensional. Worse, it is a mixed boundary value problem, with different types of condition (Neumann and Dirichlet) on the r axis. This precludes most traditional solution methods, including separation of variables and integral transforms, as they generally require the imposition of some unvarying condition along each constant-coordinate boundary. Textbooks offer little by way of advice for such problems (see, e.g., Ockendon *et al.* [17]).

A Better Coordinate System

In the *oblate spheroidal* coordinate system [145, 173], the problem ceases to be of mixed boundary type, which greatly simplifies it. Using the symbols of Lebedev [173] (there appears to be no standard), we define, in three dimensions,

$$x = \cosh\alpha\sin\beta\cos\phi\,, \tag{A.5}$$

$$y = \cosh\alpha\sin\beta\sin\phi\,, \tag{A.6}$$

$$z = \sinh\alpha\cos\beta\,. \tag{A.7}$$

Here $0 \leq \alpha < \infty$, $0 \leq \beta \leq \pi/2$ and $0 \leq \phi < 2\pi$, confining points to the upper half-plane. Note that in general oblate spheroidal coordinates (roughly translated as "squashed sphere coordinates") are parameterised according to the degree of "squashedness"—the length of half the line segment defined by $\alpha = 0$—but we have set this constant to one, which is convenient for our particular problem. As previously assuming angular independence, we obtain (taking $\phi = 0$ and relabelling x as r),

$$r = \cosh\alpha\sin\beta\,, \qquad z = \sinh\alpha\cos\beta\,. \tag{A.8}$$

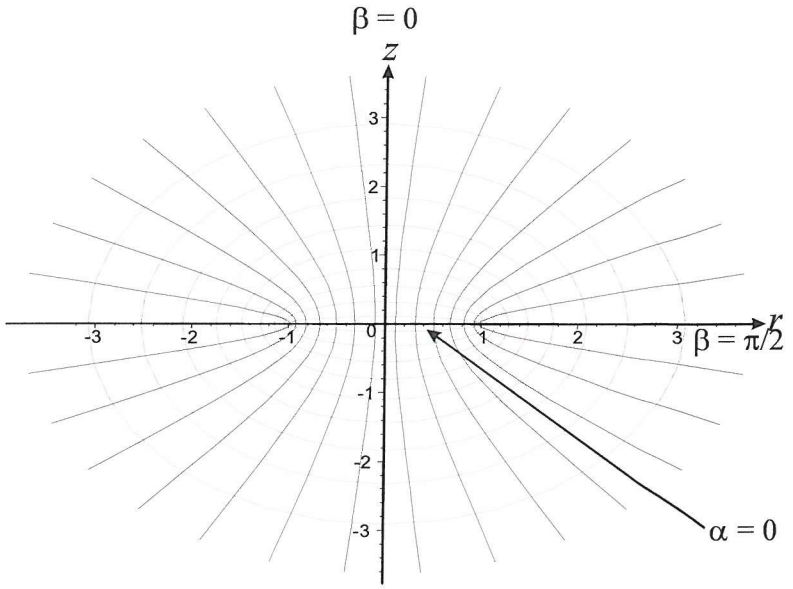

Figure A.2: A constant ϕ cross-section, showing lines of constant α and β, of the oblate spheroidal coordinate system used to solve the inlaid microdisc problem.

This cross-section is illustrated in Figure A.2, where lines of constant α and β are plotted in the $r-z$ plane.

Even after setting $\frac{\partial u}{\partial \phi} \equiv 0$, the Laplacian operator in this system (given in [173]) is rather more complicated than in the cylindrical system:

$$\nabla^2 u = \frac{1}{\cosh^2\alpha - \sin^2\beta}\left[\frac{1}{\cosh\alpha}\frac{\partial}{\partial\alpha}\left(\cosh\alpha\frac{\partial u}{\partial\alpha}\right) + \frac{1}{\sin\beta}\frac{\partial}{\partial\beta}\left(\sin\beta\frac{\partial u}{\partial\beta}\right)\right] . \tag{A.9}$$

In the particular case of Laplace's equation, however, the α and β coordinates may be disentangled to give

$$\frac{1}{\cosh\alpha}\frac{\partial}{\partial\alpha}\left(\cosh\alpha\frac{\partial u}{\partial\alpha}\right) + \frac{1}{\sin\beta}\frac{\partial}{\partial\beta}\left(\sin\beta\frac{\partial u}{\partial\beta}\right) = 0 . \tag{A.10}$$

This form of the equation is still more complicated than that in the cylindrical coordinate system, but this is the price paid for more convenient imposition of boundary conditions. In any case, the microdisc problem allows

further simplification. The Neumann conditions coincide with two different values of β (0 and $\pi/2$), but they are clearly satisfied by adopting a solution independent of β—i.e. $\frac{\partial u}{\partial \beta} \equiv 0$.

Most importantly, from the point of view of imposing boundary conditions, the microdisc boundary condition implies $u = 0$ at $\alpha = 0$. Consider the situation where $\alpha = 0$—then $r = \sin\beta$ and $z = 0$; in other words, $0 \leq r \leq 1, z = 0$, which implies that $\alpha = 0$ exactly coincides with, and only with, the microelectrode. The line between $r = 0$ and $r = 1$ is a degenerate ellipse, with a minor axis length of zero.

Thus, for the microdisc, we are left with solving an ordinary differential equation,

$$\frac{d}{d\alpha}\left(\cosh\alpha \frac{du}{d\alpha}\right) = 0 \,, \tag{A.11}$$

with the boundary conditions

$$u|_{\alpha=0} = 0 \,, \tag{A.12}$$

$$\lim_{\alpha\to\infty} u = 1 \,. \tag{A.13}$$

Since this is a linear differential equation, and it lacks a term in u, the general solution is easily found to be

$$u = c_1 + c_2 \tan^{-1}(e^{\alpha}) \tag{A.14}$$

(c_1 and c_2 are arbitrary constants). Imposing the boundary conditions gives

$$u = \frac{4}{\pi}\tan^{-1}(e^{\alpha}) - 1 \,, \tag{A.15}$$

and the problem is solved in oblate spheroidal space.

The Back-Transformation

In order to compare the solution with those given in the literature, we need (A.15) in cylindrical coordinate space. (As is seen presently, however, this is neither necessary nor desirable in order to find the current.) Achieving

this is a matter of finding α in terms of cylindrical r and z. Using standard trigonometric identities and (A.8) it is found that

$$(r+1)^2 + z^2 = \cosh^2\alpha \sin^2\beta + 2\cosh\alpha\sin\beta + 1 + \sinh^2\alpha\cos^2\beta \quad \text{(A.16)}$$

$$= \cosh^2\alpha + 2\cosh\alpha\sin\beta + \sin^2\beta \quad \text{(A.17)}$$

$$= (\cosh\alpha + \sin\beta)^2 \,. \quad \text{(A.18)}$$

Similarly,

$$(r-1)^2 + z^2 = (\cosh\alpha - \sin\beta)^2 \,. \quad \text{(A.19)}$$

So

$$\sqrt{(r+1)^2+z^2} + \sqrt{(r-1)^2+z^2} = 2\cosh\alpha \quad \text{(A.20)}$$

and

$$\alpha = \cosh^{-1}\left[\frac{1}{2}\left\{\sqrt{(r+1)^2+z^2} + \sqrt{(r-1)^2+z^2}\right\}\right] \,. \quad \text{(A.21)}$$

(In fact this is basically the Cartesian equation for an ellipse.) In the same vein,

$$\beta = \sin^{-1}\left[\frac{1}{2}\left\{\sqrt{(r+1)^2+z^2} - \sqrt{(r-1)^2+z^2}\right\}\right] \,. \quad \text{(A.22)}$$

Recalling the logarithmic definition of $\cosh^{-1}(t)$ we note that

$$\exp\left[\cosh^{-1}(t)\right] = t + \sqrt{t-1}\sqrt{t+1} \,, \quad \text{(A.23)}$$

and therefore that, if we define

$$t = \frac{1}{2}\left[\sqrt{(r+1)^2+z^2} + \sqrt{(r-1)^2+z^2}\right] \,, \quad \text{(A.24)}$$

then we can write down a form of u in cylindrical coordinates:

$$u = \frac{4}{\pi}\tan^{-1}\left(t + \sqrt{t-1}\sqrt{t+1}\right) - 1 \,. \quad \text{(A.25)}$$

Considering the various cases, we find that $t \geq 1$; (A.25) therefore simplifies slightly to

$$u = \frac{4}{\pi}\tan^{-1}\left(t + \sqrt{t^2-1}\right) - 1 \,. \quad \text{(A.26)}$$

This was apparently the form given by Saito [171], who, it would seem, used a conformal map to derive it. (The use of oblate spheroidal coordinates here is essentially equivalent to the conformal map $\alpha + i\beta = \sin^{-1}(r + iz)$, but since (A.2) is not invariant under a conformal map it seems more logical and convenient to view the transformation in terms of a new orthogonal coordinate system.)

The form of Crank and Furzeland [172] is not immediately attainable without some rearrangement, the least obvious part of which relies on a relatively obscure identity, apparently due to Charles Dodgson (better known as Lewis Carroll)[1]:

$$\tan^{-1}(p+r) + \tan^{-1}(p+q) - \frac{\pi}{2} = \tan^{-1} p \tag{A.27}$$

where

$$1 + p^2 = qr \; . \tag{A.28}$$

Consider the substitution

$$a = \sqrt{t^2 - 1} \; . \tag{A.29}$$

Then $t = \sqrt{a^2 + 1}$, and

$$u = \frac{2}{\pi}\left[2\tan^{-1}\left(\sqrt{a^2+1} + a\right) - \frac{\pi}{2}\right] \; . \tag{A.30}$$

If, in (A.27), we set $p = a$ and $q = r = \sqrt{a^2 + 1}$, satisfying the relation (A.28), we attain something relevant to our problem:

$$2\tan^{-1}(a + \sqrt{a^2+1}) - \frac{\pi}{2} = \tan^{-1} a \tag{A.31}$$

(note that this holds when a is positive, which is the case here). We therefore rewrite (A.25) as

$$u = \frac{2}{\pi}\tan^{-1} a = \frac{2}{\pi}\tan^{-1}\sqrt{t^2 - 1} \; . \tag{A.32}$$

[1] See [174]. The relation is less arbitrary when viewed in terms of arccotangents.

Still pursuing the equation of Crank and Furzeland, we use the standard identity,

$$\tan^{-1}\sqrt{\frac{1-x}{x}} = \frac{\pi}{2} - \sin^{-1}\sqrt{x}\,, \tag{A.33}$$

which holds for positive x:

$$u = \frac{2}{\pi}\tan^{-1}\sqrt{t^2-1} \tag{A.34}$$

$$= \frac{2}{\pi}\tan^{-1}\sqrt{\frac{1-1/t^2}{1/t^2}} \tag{A.35}$$

$$= \frac{2}{\pi}\left[\frac{\pi}{2} - \sin^{-1}\left(\frac{1}{t}\right)\right] \tag{A.36}$$

$$= 1 - \frac{2}{\pi}\sin^{-1}\left(\frac{1}{t}\right)\,. \tag{A.37}$$

The Crank and Furzeland expression is then, substituting from (A.24),

$$u = 1 - \frac{2}{\pi}\sin^{-1}\left(\frac{2}{\sqrt{(r+1)^2+z^2}+\sqrt{(r-1)^2+z^2}}\right)\,, \tag{A.38}$$

which is more neatly expressed, using the elementary relation $\sin^{-1}x = \pi/2 - \cos^{-1}x$, as

$$u = \frac{2}{\pi}\cos^{-1}\left(\frac{2}{\sqrt{(r+1)^2+z^2}+\sqrt{(r-1)^2+z^2}}\right)\,. \tag{A.39}$$

This derivation is not entirely rigorous in that the result does not hold for $z = 0$. This is of no great practical significance for our purposes. Of greater interest is the current, to which we turn now.

The Current

The dimensionless current is in principle given by

$$i = \frac{\pi}{2}\int_{r=0}^{r=1}\left.\frac{\partial u}{\partial z}\right|_{z=0} r\,dr\,, \tag{A.40}$$

but direct use of this definition with (A.39) does not yield the correct answer because $\frac{\partial u}{\partial z}$ is undefined in the region of integration (it is discontinuous there).

However, we can derive the current from the concentration field in oblate spheroidal coordinates.

The chain rule tells us that

$$\frac{\partial u}{\partial z} = \frac{\partial u}{\partial \alpha}\frac{\partial \alpha}{\partial z} = \frac{4e^{\alpha}}{\pi(1+e^{2\alpha})}\frac{\cos\beta\cosh\alpha}{\cos^2\beta\cosh^2\alpha + \sin^2\beta\sinh^2\alpha} . \tag{A.41}$$

Fortunately the region of interest is $\alpha = 0$, so we are left with

$$\left.\frac{\partial u}{\partial z}\right|_{\text{electrode}} = \frac{2}{\pi\cos\beta} = \frac{2}{\pi\sqrt{1-\sin^2\beta}} . \tag{A.42}$$

Clearly the expression for β in terms of r and z, (A.22), simplifies for $z = 0,\ r \leq 1$ to

$$u = \sin^{-1} r , \tag{A.43}$$

which can immediately be substituted to give

$$\frac{\partial u}{\partial z} = \frac{2}{\pi\sqrt{1-r^2}} . \tag{A.44}$$

The integration of (A.40) is then of standard type,

$$i = \int_{r=0}^{r=1} \frac{r}{\sqrt{1-r^2}} dr = 1 . \tag{A.45}$$

Conclusion

It is not suggested that the approach here is the best, and it is not the shortest, but the use of elementary methods does perhaps illustrate the difficulties of analytical solution of PDE problems. We have solved the simplest mechanism, in one of the simplest geometries, with minimal boundary conditions, and only in the steady state. And yet the amount of work required is fairly significant.

It is true that had we used the Neumann integral theorem, the effort might have been reduced somewhat. But this is a highly specific theorem, and the facts about Bessel functions on which that method rests are only to be found in specialised texts [175]. The approach used here is arguably more natural.

In any case, the more important objection is the necessity of symmetry to the approach. If the microdisc under consideration were part of an SECM tip, or even just recessed, the foundation of the solution method would crumble (spheroidal coordinates have been used with simulations, however, where they are generally thought of in terms of conformal maps [22]).

All of this being said, the method of oblate spheroidal coordinates might warrant further investigation in other situations of interest with the same geometry. The solution to the Helmholtz equation in oblate spheroidal coordinates, originally used to model wave scattering, can be adapted to the EC' mechanism case, as shown by Rajendran and Sangaranararyanan [24]. Apparently the method has not been applied to the transient problem, however, where by separation of variables some sort of solution would presumably be within reach.

Appendix B

Self-adjoint Problems

We define a self-adjoint problem as one where the governing partial differential equation operator is self-adjoint. For our purposes, the adjoint of a differential operator $\mathcal{L}$, denoted by $\mathcal{L}^*$, is defined such that, for two functions u and v over the domain Ω,

$$\int_\Omega u(\mathcal{L}v)d\Omega = \int_\Omega (\mathcal{L}^*u)v\Omega \,, \tag{B.1}$$

ignoring boundary terms. If $\mathcal{L} = \mathcal{L}^*$ then $\mathcal{L}$ is self-adjoint.

This abstract mathematical definition being difficult to picture, we proceed with an example. Consider the ODE

$$\frac{d^2u}{dx^2} + a\frac{du}{dx} + b = 0 \tag{B.2}$$

over an interval $0 \leq x \leq 1$. With a weight function v this becomes

$$\int_0^1 v\left(\frac{d^2u}{dx^2} + a\frac{du}{dx} + b\right)dx = 0 \,. \tag{B.3}$$

In exactly the same manner as the BEM derivation, we integrate (B.3) by parts twice (for FEM we would usually integrate by parts once). This yields

$$\int_0^1 u\left(\frac{d^2v}{dx^2} - a\frac{dv}{dx} + b\right)dx + \text{boundary terms} = 0 \,. \tag{B.4}$$

So the adjoint of the differential operator

$$\frac{d^2}{dx^2} + a\frac{d}{dx} + b \tag{B.5}$$

is

$$\frac{d^2}{dx^2} - a\frac{d}{dx} + b\,, \tag{B.6}$$

and (B.5) is only self-adjoint when $a \equiv 0$.

As it happens, the pattern generalises to operators in higher dimensional spaces, so the (modified) Helmholtz equation,

$$\nabla^2 u \pm ku = 0 \tag{B.7}$$

is self-adjoint since it only contains second order derivatives, whereas the steady-state diffusion-convection equation,

$$\nabla^2 u - \mathbf{v} \cdot \nabla \mathbf{u} = 0 \tag{B.8}$$

is not, on account of the first order derivatives in the convective term. In practice, the question of whether or not we are solving a self-adjoint problem hinges on the presence of convection.

Self-adjoint operators have a number of useful attributes. The only one that concerns us is that they give symmetric matrices (see Appendix C on the next page) with the Galerkin FEM formulation (this is a reflection of the existence of *variational* [16] formulations for such problems). In such cases simpler matrix solvers can be used.

Appendix C

Matrices

Numerical PDE solution methods generally replace the exact infinite PDE problem with one of finite dimension; the differential operator is replaced by a matrix, and the solution function by a finite vector. The result of this discretisation is usually a matrix equation of the form

$$\mathsf{A}\mathbf{x} = \mathbf{b}\,, \tag{C.1}$$

where A is a known square matrix, $\mathbf{b}$ a known vector, and $\mathbf{x}$ the unknown vector we wish to find. Sometimes, as with explicit finite difference, solving (C.1) is implicitly incorporated into the algorithm because the form of A is very simple. But often the end result of PDE discretisation is a matrix A and a righthand side $\mathbf{b}$ comprising a linear system in need of solving with standard methods.

Generally, solving (C.1) is the slowest part of the overall solution process, and consequently much effort has been expended in finding efficient solving algorithms. No one ultimate algorithm has been discovered for an entirely general A [69,72]. Fortunately the matrices produced by finite element often have special properties allowing faster specialised algorithms to operate. The area of linear algebra is huge and complicated. We attempt here to do no more than introduce the techniques used for solving our linear systems.

Firstly, self-adjoint problems yield, with the Galerkin formulation, *sym-*

metric matrices —i.e. $A_{ij} = A_{ji}$. Or, introducing the notation for the transpose of A (A with its columns and rows transposed), A^T, A is symmetric if

$$\mathsf{A}^T = \mathsf{A} \,. \tag{C.2}$$

This is obviously easily determined by inspection of the matrix elements. The transpose of the inverse is denoted thus:

$$\left(\mathsf{A}^{-1}\right)^T = \mathsf{A}^{-T} \,. \tag{C.3}$$

Secondly, such problems are often of a nature that they yield *positive definite* matrices. Positive-definiteness is most easily defined by saying that, for positive definite A,

$$\mathbf{x}^T\mathsf{A}\mathbf{x} > 0 \qquad \forall \mathbf{x} \text{ where } \|\mathbf{x}\| > 0 \,. \tag{C.4}$$

This is equivalent to stating that all of A's eigenvalues are positive [30, 90]. It is not a property easily determinable by inspection, but can be proved *a priori* for certain problems. For the self-adjoint problems solved in this work, the matrices are positive definite.

Finally, and most importantly, the finite element method always produces *sparse* matrices, where most elements are zero, because non-zero entries only occur where nodes are connected by element edges[1]; in fact, FE matrices have $\mathcal{O}(n)$ non-zero entries, where n is the number of nodes and rows/columns in A. (Consider a structured mesh, where each node is connected to six others: we would have six non-zero entries on n rows.) An ideal solving algorithm, therefore, would ignore as many of the zeros as possible. There are two basic ways of trying to achieve this: *sparse direct solvers* and *iterative* methods.

Direct solvers—generally variants on *Gaussian elimination*—utilise successive elementary manipulations, much like those used by humans, to eliminate matrix elements and ultimately determine $\mathbf{x}$. The latter repeatedly

[1]Note that this is a marked difference from boundary element, whose matrices are dense, since all boundary elements are connected to all others. (In fact BEM matrices usually lack the other advantageous properties discussed here too.)

apply some operation to a starting guess vector, and iterate towards the solution; the number of iterations (steps) required is not usually known *a priori*, and depends in part on the accuracy desired.

Direct solvers

Direct methods can be and are used in many finite element packages, but usually in special forms—raw Gaussian elimination is not generally efficient. It can be shown [69, 89, 90] that Gaussian elimination-type algorithms for general matrices take $\mathcal{O}(n^3)$ operations, where n is the number of columns in A. Since they require the storage of the full matrix, they also clearly require $\mathcal{O}(n^2)$ storage. Since Gaussian elimination can be applied to any matrix—dense, asymmetric, indefinite—this represents the performance baseline. We work to improve on it, as large problems would clearly quickly require unreasonable amounts of computer time and storage.

A particular direct solver only applicable to symmetric positive definite matrices is *Cholesky factorisation*. It only improves on the speed of Gaussian elimination by a constant factor, but proves important for other reasons—see Appendix D and § 3.2.3 on page 114. The most obvious aspect is that it utilises the symmetry of A to avoid duplicating work. Further, it avoids the *pivoting* required in general Gaussian elimination (*ibid.*), and so is simpler. The algorithm is described in texts. The important fact is that it results in the factorisation of A as

$$\mathsf{A} = \mathsf{L}\mathsf{L}^T , \tag{C.5}$$

where L is a lower triangular matrix—that is, all entries above the diagonal are zero; L^T is by extension an upper triangular matrix. With such a factorisation the system (C.1) becomes

$$\mathsf{L}\mathsf{L}^T\mathbf{x} = \mathbf{b}. \tag{C.6}$$

This is solved by first substituting $\mathbf{y} = \mathsf{L}^T\mathbf{x}$, to give

$$\mathsf{L}\mathbf{y} = \mathbf{b} \tag{C.7}$$

$$\mathbf{y} = \mathsf{L}^{-1}\mathbf{b}\,, \tag{C.8}$$

then solving

$$\mathsf{L}^T\mathbf{x} = \mathbf{y} \tag{C.9}$$

$$\mathbf{x} = \mathsf{L}^{-T}\mathbf{y} \tag{C.10}$$

The two matrix inverses are formal only—no inverse is calculated. Instead, because L and L^T are triangular matrices, they are very easy to solve by back-substitution: they unravel in $\mathcal{O}(n^2)$ operations.

The construction of the factorisation (C.5) is an $\mathcal{O}(n^3)$ operation in general, just like Gaussian elimination, so we have not greatly improved upon Gaussian elimination, or the generalised variant of Cholesky factorisation, *LU factorisation* (*decomposition*). LU decomposition factorises A into lower and upper triangular matrices L and U:

$$\mathsf{A} = \mathsf{L}\mathsf{U}\,. \tag{C.11}$$

The solution process then proceeds exactly as with Cholesky factorisation. This is probably the most commonly used form of Gaussian elimination, as once the $\mathcal{O}(n^3)$ factorisation has been performed any number of linear systems with the matrix A but different righthand sides can be solved in $\mathcal{O}(n^2)$ operations each. We do not use Cholesky factorisation for solving the finite element system matrix, but we do use it for the small matrices arising from SPR gradient smoothing (it is a way of solving least squares minimisation problems), and it also appears in preconditioning methods for conjugate gradients—see Appendix D.

None of the direct methods described so far take advantage of sparseness. This is the central challenge of using direct methods for finite element solutions. The basic problem is that it is intimately tied to the sparsity pattern

of the matrix in question. If the mesh has a regular connectivity pattern it will be reflected in the corresponding matrix, and can be exploited systematically in a direct solver. Commonly this boils down to predetermining the *bandwidth* of A, and treating it as *band diagonal*. (The bandwidth is the distance of the furthest non-zero entry from the diagonal. A matrix is termed *band diagonal* if this is significantly less than n.)

Unfortunately, during the Gaussian elimination process with pivoting, previously zero entries usually become non-zero (called "fill-ins"), increasing the bandwidth, and as a consequence drastically reducing the solver's efficiency [69]. Further, node re-numbering techniques are needed with unstructured meshes (like ours) to get maximum efficiency. Even with the best algorithms, the bandwidth for two dimensional problems scales with grid density, so with increased discretisation accuracy so come diminishing returns on computer time.

It may be possible to construct a competitive direct solver for our cases, but it seems unlikely. It would certainly be complex, and our unstructured meshes seem particularly ill-suited for it (the sort of approaches used are described by Golub and van Loan [69]). It is probably true to say that most finite element practitioners now use iterative solvers, as described below, and we follow the majority in this.

Iterative solvers

The attraction of iterative methods is that they do not manipulate the matrix A directly; thus they do not require it to be stored in its entirety—a great advantage with very sparse matrices like ours. They come in stationary and non-stationary forms, exemplified by *successive over-relaxation* and *conjugate gradient* respectively. In the first case, the same transformation is applied to the guess vector each time; in the second, information from previous steps is used to alter the operation for each iteration.

It is generally accepted that stationary iterative methods are inferior to

both direct and non-stationary iterative algorithms, so they are of little use for the actual solving process. However, it happens that stationary iterative methods reappear in the guise of *preconditioners* for conjugate gradient-type methods—see Appendix D.

Conjugate gradient solvers, which are applicable only to symmetric positive definite matrices, only require storage of and mathematical operations on the non-zero matrix entries—regardless of the matrix structure or bandwidth—and never manipulate the matrix, allowing very efficient implementation (see Appendix D). In Appendix D we find that conjugate gradient is approximately $\mathcal{O}(n^{3/2})$ for the problems solved in this work, which is considerably better than basic Gaussian elimination, and probably beyond what any sparse direct method could achieve[2]. More importantly, it avoids all the complexity of sparse direct solvers, essentially requiring only efficient matrix-vector multiplication to achieve near-optimal performance. We therefore use conjugate gradient as the main solver for the linear system generated by finite element assembly where convection is not present. For convective problems, other Krylov space solvers are used.

[2]Note though that the order of a method is not the only measure of its performance. Doubtless for small enough problems direct methods would be faster.

Appendix D

Krylov Space Solvers

It is probably fair to say that Krylov space solvers are not intuitively understandable. A specialised text [69,71,72] should be consulted on their details. Since, however, they form such a crucial part of our finite element solver, and because conjugate gradient is the simplest example thereof, a brief, far from rigorous, adumbration of it is given below. It seems, from the literature (e.g. [32]), true to say that only multigrid methods can out-perform conjugate gradient, and yet the implementation of it is startlingly simple, even with a preconditioner. Even the more complex non-symmetric solvers can be implemented relatively easily, and they are probably simpler than the most efficient direct methods. A brief description of those is given at the end of this appendix.

Solving as Minimisation

The starting point for conjugate gradient is the fact that, if A is symmetric and positive definite, solving (C.1) is equivalent to minimising the quadratic form (see [90]):

$$q(\mathbf{x}) = \mathbf{x} \cdot \mathsf{A}\mathbf{x} - 2\mathbf{x} \cdot \mathbf{b} \,. \tag{D.1}$$

This can be shown by considering $q(\mathbf{x}+\mathbf{e})$—the quadratic form of the solution $\mathbf{x}$ plus an error term $\mathbf{e}$:

$$q(\mathbf{x}+\mathbf{e}) = (\mathbf{x}+\mathbf{e}) \cdot \mathsf{A}(\mathbf{x}+\mathbf{e}) - 2(\mathbf{x}+\mathbf{e}) \cdot \mathbf{b} \tag{D.2}$$

$$= \mathbf{x} \cdot \mathsf{A}\mathbf{x} + \mathbf{x} \cdot \mathsf{A}\mathbf{e} + \mathbf{e} \cdot \mathsf{A}\mathbf{x} + \mathbf{e} \cdot \mathsf{A}\mathbf{e} - 2\mathbf{x} \cdot \mathbf{b} - 2\mathbf{e} \cdot \mathbf{b} \tag{D.3}$$

$$= q(\mathbf{x}) + \mathbf{x} \cdot \mathsf{A}\mathbf{e} + \mathbf{e} \cdot \mathsf{A}\mathbf{x} + \mathbf{e} \cdot \mathsf{A}\mathbf{e} - 2\mathbf{e} \cdot \mathbf{b} \tag{D.4}$$

Because $\mathsf{A}^T = \mathsf{A}$, $\mathbf{x} \cdot \mathsf{A}\mathbf{e} = \mathbf{e} \cdot \mathsf{A}\mathbf{x}$, so

$$q(\mathbf{x}+\mathbf{e}) = q(\mathbf{x}) + 2\mathbf{e} \cdot \mathsf{A}\mathbf{x} + \mathbf{e} \cdot \mathsf{A}\mathbf{e} - 2\mathbf{e} \cdot \mathbf{b} \,. \tag{D.5}$$

But, from (C.1), $\mathsf{A}\mathbf{x} = \mathbf{b}$, so

$$q(\mathbf{x}+\mathbf{e}) = q(\mathbf{x}) + \mathbf{e} \cdot \mathsf{A}\mathbf{e} \,. \tag{D.6}$$

Since A is positive definite, we know from (C.4) that the second term on the right is always positive, so q is minimised where $\mathbf{e} = \mathbf{0}$, and is greater everywhere else; the solution $\mathbf{x}$ is at its minimum.

CG Minimisation

A relatively obvious way of solving the minimisation problem is to start at some point and take steps following the steepest gradient of the quadratic form down to the minimum. This is termed the *method of steepest descent*, and is too slow for solving linear systems. The difficulty is that many steps in parallel or near parallel directions can be taken. The solution is to ensure successive directions are orthogonal, but this is impossible by the usual definition, because it would require knowing the solution to the problem being solved [166]. Instead conjugate gradient ensures steps are *A-orthogonal*, or *conjugate* (*ibid.*). Two vectors $\mathbf{c}$ and $\mathbf{d}$ are A-orthogonal if

$$\mathbf{c} \cdot \mathsf{A}\mathbf{d} = 0 \,. \tag{D.7}$$

Sets of orthogonal vectors, whatever the type of orthogonality, can be generated by the Gram-Schmidt process [30, 90]. Unfortunately this is both

numerically unstable and demanding of space and time: it requires storage of, and processing with, all past vectors to generate a new orthogonal vector. The conjugate gradient algorithm manages to avoid this by keeping just a handful of vectors and scalars updated at each step, and using a recurrence relation. The operations for iteration i are as below:

$$\mathbf{z}_{i+1} \Leftarrow \mathsf{A}\mathbf{v}_i \tag{D.8}$$

$$t_{i+1} \Leftarrow \frac{c_i}{\mathbf{v}_i \cdot \mathbf{z}_{i+1}} \tag{D.9}$$

$$\mathbf{x}_{i+1} \Leftarrow \mathbf{x}_i + t_{i+1}\mathbf{v}_i \tag{D.10}$$

$$\mathbf{r}_{i+1} \Leftarrow \mathbf{r}_i - t_{i+1}\mathbf{z}_{i+1} \tag{D.11}$$

$$d_{i+1} \Leftarrow \mathbf{r}_i \cdot \mathbf{r}_i \tag{D.12}$$

$$\mathbf{v}_{i+1} \Leftarrow \mathbf{r}_i + \frac{d_i}{c_i}\mathbf{v}_i \tag{D.13}$$

$$c_{i+1} \Leftarrow d_i \ , \tag{D.14}$$

with the starting values

$$\mathbf{r}_0 = \mathbf{v}_0 = \mathbf{b} - \mathsf{A}\mathbf{x}_0 \tag{D.15}$$

$$c_0 = \mathbf{r}_0 \cdot \mathbf{r}_0 \ . \tag{D.16}$$

It should be apparent that at each step the vector $\mathbf{r}_i$ holds the residual—the measure of the extent to which $\mathbf{x}_i$ fails to satisfy (C.1). The starting vector $\mathbf{x}_0$ is less important than might be thought: the method converges whatever the value, and although a good guess for $\mathbf{x}_0$ can speed up convergence, it is not particularly influential.

Speed

On the face of it, the primary computational cost per step is the matrix-vector product $\mathsf{A}\mathbf{v}_i$. Since this product is the only use of A, the matrix may be stored in the manner most efficient for its computation, and for storage space. With our type of sparse matrix, as discussed in Appendix C, we

can expect the matrix-vector product to be $\mathcal{O}(n)$, where n is the number of matrix rows. Thus it becomes of the same order as the vector operations, and we must be concerned about the speed of all of them.

Since all the steps taken by the conjugate gradient method are conjugate, and there exist only n conjugate directions in an n dimensional space, the method must theoretically terminate in n steps. The dimensionality n is, of course, equal to the number of rows/columns in A. The real situation is not quite as simple, however. On the downside, numerical imprecision destroys exact orthogonality, meaning we will *not* reach the exact solution in n steps. But this is more than counterbalanced by the realisation that, because we are using a minimisation process, we needn't seek an exact solution, but one within a certain tolerance. We therefore stop when $d = \mathbf{r} \cdot \mathbf{r} < \varepsilon$, for some small value ε (taken throughout this work to be 10^{-5}). Other termination criteria exist, notably ones based on the relative residual; these might well justify investigation.

It can be shown, in fact, that the number of steps to achieve a usable solution is roughly proportional to the square root of the matrix condition number κ [90], which broadly speaking measures how close to singular A is. So we know that CG is $\mathcal{O}(n\sqrt{\kappa})$. It can be shown that for many two dimensional problems $\kappa \propto n$ [166]. Thus CG is, for our purposes, around $\mathcal{O}(n^{3/2})$—far better than the $\mathcal{O}(n^3)$ obtainable from naïve Gaussian elimination methods or similar. Nonetheless, anything that can be done to reduce κ will speed up solving. This is achieved via *preconditioning* the matrix.

Preconditioned CG (PCG)

If, instead of solving

$$\mathsf{A}\mathbf{x} = \mathbf{b} \,, \tag{D.17}$$

an alternative system with the same solution,

$$\mathsf{M}^{-1}\mathsf{A}\mathbf{x} = \mathsf{M}^{-1}\mathbf{b} \,, \tag{D.18}$$

is solved, this is termed left-preconditioning (other possibilities include right or split preconditioning, where the preconditioning matrix appears on the right of A, or in two pieces either side of it, respectively). If M is in some way "close" to matrix A, then we might expect the system to be solved in fewer steps, as in the limit where $\mathsf{M} = \mathsf{A}$, $\mathsf{M}^{-1}\mathsf{A}$ is the identity matrix, and the system has a condition number of one. On the other hand, solving the preconditioned system generally requires formal multiplication by M^{-1} during each solver iteration, increasing the cost of each step. Therefore for the method to be effective M must be simple to invert—for instance it could be a diagonal matrix, or a triangular one. Numerous preconditioning approaches are described, both for CG and unsymmetric solvers, by Saad [72], and they remain an active area of research.

Where M is the diagonal matrix formed from the diagonal values of A it is known as the *Jacobi preconditioner*, on account of the relation with the stationary iterative Jacobi iterative solver. Trivial to implement, Jacobi preconditioning was incorporated into the finite element program, and gave a modest increase in speed.

A more effective preconditioner is generally known as symmetric successive over-relaxation (SSOR). By analogy with Jacobi iterations, it uses the symmetric version of successive over-relaxation (SOR) instead. This entails a split preconditioner of two triangular matrices, which is more time consuming than diagonal Jacobi preconditioning matrix, but special tricks [32, 72] bring the performance close to Jacobi levels. In the particular case of symmetric Gauss-Seidel (SGS), a particularly efficient implementation allows steps as almost as fast as without preconditioning. This too was implemented, and achieved another modest increase in speed.

There is evidently a tradeoff between reduction of the number of steps and the cost of each step: the closer M^{-1} is to A^{-1}, the fewer steps, but the greater the cost of each. The two aforementioned preconditioners essentially cost nothing per step, with clever implementations, and are always worth using. More complex preconditioning matrices can further dramatically reduce the

number of steps, but at the cost of ever greater complexity and processing per step. Of the generic options, not associated with any particular form of A, incomplete Cholesky factorisation (sometimes referred to as its more general variant, incomplete LU factorisation—ILU) is probably the most significant. As the name suggests, it approximately factorises A into upper and lower triangular matrices, the one the transpose of the other. Numerous versions exist, and they give varying speedups, but are less reliable than the simpler preconditioners above.

Many other approaches exist, most significantly *multigrid*, which can be a fast solver in its own right. We do not pursue them here, as the linear system solver performance with was found to be adequate for the problems under study, although of course with more complex problems in the future more advanced preconditioners might well prove necessary.

Non-Symmetric Systems

Numerous iterative solvers exist that attempt to translate some of the attractive properties of conjugate gradient to non-symmetric systems. Unfortunately it has been proven that no similarly elegant equivalent exists, in the sense that the recurrence relation that generates the series of search directions in CG has no analogue where the matrix is unsymmetric [176]. A large number of algorithms have been proposed, none of which functions well for all matrices. Two of the most common are biconjugate gradient (BiCG) (and its variants) and generalised minimum residual (GMRES).

In BiCG, sequences of search directions that are biorthogonal are are generated. This can be done with a simple recurrence relation, and if the sequence terminates then the solution has been found. Saad [72] notes that this in effect entails solving the transpose system in tandem with the one for A (in the case where A is symmetric they are the same thing, and the algorithm reduces to an inefficient form of conjugate gradient). The problem is that BiCG quite commonly breaks down without reaching a solution; it is not

guaranteed at all to converge. Indeed, it has irregular convergence, which is why variants such as stabilised biconjugate gradient (BiCGStab) exist. This can sometimes be more robust, but introduces new ways in which breakdown can occur, and there remain no guarantees.

GMRES is generally considered to be more robust than the biconjugate gradient-derived algorithms. This is not surprising, as it usually employs a far longer recurrence (e.g. fifty)—in other words it has a longer memory of previous search directions. Full GMRES requires storage of every previous search direction, and ensures orthogonality by looping through every one for each iteration. It is therefore far too slow, and in practice the recurrence is generally terminated after fixed number of vectors, to give Restarted GMRES (sometimes written as RGMRES, but since full GMRES is never used, the abbreviation is probably redundant). The result is a solver that has a relatively large cost per iteration, but which generally has a lower tendency to break down than faster methods.

BiCG, BiCGStab and GMRES were all implemented in the simulation program. As with conjugate gradient, these unsymmetric solvers benefit from effective preconditioners. It was found that, where SUPG stabilisation techniques were used, BiCGStab with the simple Jacobi preconditioner was able to solve most systems quickly, leading to overall simulation times of a few seconds. Doubtless it would benefit from more sophisticated preconditioners, however.

Appendix E

Simulation Input Files

All of the simulations in this thesis were conducted using one program embodying the theory of Chapter 3. This flexibility necessitated a relatively general input file format for describing each problem. This format is not documented here; instead examples of the files used to generate the results in Chapters 4 and 5 are given, in the hope that the format is self-explanatory enough to illustrate the program capability, as well as provide exact details about the simulation settings used. Since most sets of simulations only varied according to some parameter—a recess depth or kinetic rate constant, for instance—it should be possible to reconstruct the full range of input files relatively easily.

It should be born in mind that the program uses a number of default settings which can be assumed. Firstly, elements are quadratic. Secondly, SPR is used for gradient smoothing. Thirdly, for self-adjoint problems (that is, problems without convection), which yield symmetric linear systems, conjugate gradient is used to solve the linear systems. The default preconditioner for this is Symmetric Gauss-Seidel (equivalent to SSOR with a relaxation parameter of one) [71, 72]. For convective systems, stabilised biconjugate gradient (BiCGStab) (*ibid.*) is used with a Jacobi preconditioner, although this is not the default, and is explicitly set in the input file.

In the files that follow, each field and its derivatives come in two forms,

with and without a preceded tilde. The former is the lower order one ($\mathcal{O}(h^2)$ for quadratic elements); the latter is higher ($\mathcal{O}(h^3)$ for quadratic elements). By use of the difference between the two, in various forms, an error estimator is calculated, as described in Chapter 3. Throughout, subscripts of `x` or `y` indicate differentiation with respect to that coordinate; an `n` subscript, only applicable on the boundary, indicates differentiation with respect to the outward normal.

In the `AdaptiveControl` section of the file, an element error estimate is supplied as `elem_err`, the sum over all elements of which is denoted by `elem_err_sum` (calculated automatically). At each stage of the iterative refinement process, the global error estimate—which, depending on the error norm, can be or involve `elem_err_sum`—is compared with the global error tolerance (given as `glob_err_tol`) specified in the `Settings` section. If it is higher, refinement proceeds according to a comparison between each element's estimated error, and the user specified `elem_err_tol`. This framework is used in these files to refine the mesh according to the current error estimate described in Chapter 3. It could of course be used to refine according to other norms, and in fact the default is the L_2 gradient norm [16].

Inlaid E Mechanism Microdisc

The input file for this case is more complicated than some others, since the analytical solution (given here as `sol`) is imposed, and two fields are solved for: the primal problem, `u`, and the dual problem, `v`. Both are solved with a weighting of `x`—i.e. in the cylindrical coordinate system. The tests with linear elements in Chapter 4 were run with the line `elem_order linear` in the `Settings` section.

```
GoverningEqs
{
    u { (x*u_x)_x+(x*u_y)_y = 0 }
    v { (x*v_x)_x+(x*v_y)_y = 0 }
```

```
}

UserExprs
{
    pi := 3.14159265358979323846264338327795
    sol := (2/pi)*acos(2/(sqrt(y^2+(1+x)^2)+sqrt(y^2+(1-x)^2)))
}
BCDefs
{
    insulator { flux(u) = 0 flux(v) = 0 }
    collector { u = 0 v = 1 }
    generator { u = sol v = 0 }
}
Boundary
{
    axis:       0       0   insulator
                0       2   generator
                2       2   generator
                2       0   insulator
    electrode:  1       0   collector
}
Functionals
{
    b_current := -(pi/2)*int(electrode,u_n*x)
    ~b_current := -(pi/2)*int(electrode,~u_n*x)
    d_current := -(pi/2)*int(domain,(u_x*v_x+u_y*v_y)*x)
    ~d_current := -(pi/2)*int(domain,(~u_x*~v_x+~u_y*~v_y)*x)
}
Settings { glob_err_tol 0.05 }
AdaptiveControl
{
```

```
    elem_err := (pi/2)*abs(int(element,
                (~u_x*~v_x+~u_y*~v_y-u_x*v_x-u_y*v_y)*x))
    elem_err_tol := glob_err_tol/n_elems
    glob_err := elem_err_sum
}
```

Recessed E Mechanism Microdisc

The recessed E mechanism microdisc illustrates a so-called "inverse dual" problem, where only one field need be solved for, because we use a relatively large domain (`d`=200 units here), and impose $u = 1$ on the bulk boundary. It also shows the use of relative error estimation, as opposed to the less realistic absolute error estimation used by Harriman *et al.* [34,36–38]. The input file includes the approximate expressions described in §4.2.2, as it is convenient to present their values alongside the simulated current value.

By setting the recess depth, `L`, to a negative value raised microdisc simulations may be run. Alternatively, by slightly modifying the `Boundary` section, an inlaid microdisc in a finite domain may be simulated.

```
GoverningEqs
{
    u { (x*u_x)_x+(x*u_y)_y = 0 }
}
UserExprs
{
    d := 200
    L := 0.5
    pi := 3.14159265358979323846264338327 95

    A1 := 1.6843
    A2 := -1.3237
    A3 := 1.7116
```

```
    A4 := -0.7585
}
BCDefs
{
    insulator { flux(u) = 0 }
    collector { u = 0 }
    generator { u = 1 }
}
Boundary
{
    axis:       0       0   insulator
                0       d   generator
                d       d   generator
                d       L   insulator
                1       L   insulator
    electrode:  1       0   collector
}
Functionals
{
    ~d_current := (pi/2)*int(domain,(~u_x^2+~u_y^2)*x)

    bond_expr := pi/(4*L+pi)
    ferrigno_expr := exp(-0.96*L)
    bartlett_expr := 1/(1+A1*L+A2*L^2+A3*L^3+A4*L^4)
}
Settings { glob_err_tol 0.01 }
AdaptiveControl
{
    elem_err := (pi/2)*abs(int(element,(~u_x^2+~u_y^2-u_x^2-u_y^2)*x)/
                int(domain,(~u_x^2+~u_y^2)*x))
    elem_err_tol := glob_err_tol/n_elems
```

```
    glob_err := elem_err_sum
}
```

Generator-Collector Microband

The microband case mainly differs in being simulated in a Cartesian, as opposed to cylindrical, system (note the form of the PDE in `GoverningEqs`). It also has a slightly different arrangement of boundary conditions. The exact solution expression, as described in § 4.4.2 on page 173, is calculated here to display alongside the simulated current.

```
GoverningEqs
{
    u { u_xx+u_yy = 0 }
}
UserExprs
{
    d := 50
    w := 1
    g := 1
    k := g/(g+2*w)
}
BCDefs
{
    insulator { flux(u) = 0 }
    collector { u = 0 }
    generator { u = 1 }
}
Boundary
{
        -d-g/2-w    0   insulator
        -d-g/2-w    d   insulator
```

```
        d+g/2+w     d   insulator
        d+g/2+w     0   insulator
coll:   g/2+w       0   collector
        g/2         0   insulator
gen:    -g/2        0   generator
        -g/2-w      0   insulator
}
Functionals
{
    d_current := int(domain,u_x^2+u_y^2)
    ~d_current := int(domain,~u_x^2+~u_y^2)

    exact := EllipticK(sqrt(1-k^2))/EllipticK(k)/2
}
Settings { glob_err_tol 0.01 }
AdaptiveControl
{
    elem_err := abs(int(element,~u_x^2+~u_y^2-u_x^2-u_y^2)/
                int(domain,~u_x^2+~u_y^2))
    elem_err_tol := glob_err_tol/n_elems
    glob_err := elem_err_sum
}
```

Inlaid EC' Mechanism Microdisc

The EC' mechanism requires the solution of the modified Helmholtz equation, and so the main difference between this case and the E mechanism microdisc one is the change in the governing PDE. Another change is that, owing to the boundary conditions imposed, the problem is self-dual. The file here is easily modified to form one for the recessed microdisc variant. Note the difference between the `AdaptiveControl` section and that of the

previous files—there is more than one way to express the relations between local and global errors, and this one is slightly more efficient, since only one integral need be evaluated per element during the error estimation phase; the difference is one of relative element error versus absolute element error.

```
UserExprs
{
    pi := 3.14159265358979323846264338327 95
    d := 10
    k := 1000
}
GoverningEqs
{
    u { (x*u_x)_x+(x*u_y)_y-k*x*u = 0 }
}
BCDefs
{
    insulator { flux(u) = 0 }
    collector { u = 0 }
    generator { u = 1 }
}
Boundary
{
    axis:       0   0   insulator
                0   d   collector
                d   d   collector
    insulator:  d   0   insulator
    electrode:  1   0   generator
}
Functionals
{
```

```
    d_current := (pi/2)*int(domain,(u_x^2+u_y^2+k*u^2)*x)
    ~d_current := (pi/2)*int(domain,(~u_x^2+~u_y^2+k*~u^2)*x)

    asymptotic := (pi/4)*(sqrt(k)+1+1/(4*sqrt(k)))
    pade := (1+2.0016*k^(1/2)+1.8235*k+0.96367*k^(3/2)+0.307949*k^2+
                0.049925*k^(5/2))/(1+1.3650*k^(1/2)+0.8826*k+
                0.32853*k^(3/2)+0.063566*k^2)
}
Settings { glob_err_tol 0.01 }
AdaptiveControl
{
    elem_err := (pi/2)*abs(int(element,((~u_x^2+~u_y^2+~u^2*k)-
                                        (u_x^2+u_y^2+u^2*k))*x))
    elem_err_tol := (glob_err_tol/n_elems)*((pi/2)*int(domain,
                                    (~u_x^2+~u_y^2+k*~u^2)*x))
    glob_err := elem_err_sum/((pi/2)*int(domain,(~u_x^2+~u_y^2+k*~u^2)*x))
}
```

Channel Flow E Mechanism Microband

Clearly the essential difference with the channel flow microband case is the presence of convection. Apart from modifying the mass transport PDE, this gives a different form to the current domain integral, and in turn to the error estimation formula.

Given below is the non-adjoint version, which only solves for one field. The file is slightly longer than others due to the more involved normalisation scheme. The approximate expressions of Ackerberg *et al.*, Newman and Levich (see § 4.6.2 on page 193) are calculated for display alongside the simulated current.

```
UserExprs
{
```

```
    D := 1e-5
    v_0 := 1200
    x_e := 5e-4
    h := 0.02
    d := 1

    V_f := 4*d*h*v_0/3
    p_1 := x_e/h
    p_2 := V_f/(d*D)
    P_s := (3/2)*p_1^2*p_2
    v := (1/2)*P_s*y*(2-p_1*y)

    x_min := -16
    x_max := 17
    y_max := 2/p_1

    g := (ln(4/sqrt(P_s))+1.0559)^(-1)
    pi := 3.14159265358979323846264338327 95
}
GoverningEqs
{
    u { u_xx+u_yy-v*u_x = 0 }
}
BCDefs
{
    insulator { flux(u) = 0 }
    electrode { u = 0 }
    bulk { u = 1 }
}
Boundary
{
```

```
        x_min   0       bulk
        x_min   y_max   insulator
        x_max   y_max   insulator
        x_max   0       insulator
elec:   1       0       electrode
        0       0       insulator
}
Settings
{
    glob_err_tol 0.05
    nonsym_solver bicgstab
    nonsym_preconditioner jacobi
}
Functionals
{
    _P_s := P_s
    ~d_current := int(domain,~u_x^2+~u_y^2-(1-~u)*v*~u_x)
    ackerberg := pi*g*(1-0.04633*P_s*g)
    newman := 0.8075*P_s^(1/3)+0.7085*P_s^(-1/6)-0.1984*P_s^(-1/3)
    levich := 0.8075*P_s^(1/3)
}
AdaptiveControl
{
    elem_err := (pi/2)*abs(int(element,(~u_x^2+~u_y^2-(1-~u)*v*~u_x)-
                                    (u_x^2+u_y^2-(1-u)*v*u_x)))
    elem_err_tol := (glob_err_tol/n_elems)*abs(int(domain,
                                    ~u_x^2+~u_y^2-(1-~u)*v*~u_x))
    glob_err := elem_err_sum/abs(int(domain,
                    ~u_x^2+~u_y^2-(1-~u)*v*~u_x))
}
```

For comparison, the differing sections for the adjoint influence function form of the same problem is given below. Note the backward convection equation used for the influence function (since `v` is already in use for the flow velocity, `w` is used for this).

```
GoverningEqs
{
    u { u_xx+u_yy-v*u_x = 0 }
    w { w_xx+w_yy+v*w_x = 0 }
}
BCDefs
{
    insulator { flux(u) = 0 flux(w) = 0 }
    electrode { u = 0 w = 1 }
    inlet { u = 1 w = 0 }
    outlet { flux(u) = 0 w = 0 }
}
Boundary
{
        x_min   0       inlet
        x_min   y_max   insulator
        x_max   y_max   outlet
        x_max   0       insulator
elec:   1       0       electrode
        0       0       insulator
}
Functionals
{
    _P_s := P_s
    ~d_current := -int(domain,~u_x*~w_x+~u_y*~w_y+~w*v*~u_x)
    ackerberg := pi*g*(1-0.04633*P_s*g)
```

```
    newman := 0.8075*P_s^(1/3)+0.7085*P_s^(-1/3)-0.1984*P_s^(-1/3)
    levich := 0.8075*P_s^(1/3)
}
AdaptiveControl
{
    elem_err := (pi/2)*abs(int(element,
                        (~u_x*~w_x+~u_y*~w_y+~w*v*~u_x)-
                        (u_x*w_x+u_y*w_y+w*v*u_x)))
    elem_err_tol := (glob_err_tol/n_elems)*abs(int(domain,
                            ~u_x*~w_x+~u_y*~w_y+~w*v*~u_x))
    glob_err := elem_err_sum/abs(int(domain,
                        ~u_x*~w_x+~u_y*~w_y+~w*v*~u_x))
}
```

SECM Approach Curve

The input file for an SECM approach curve is not radically different from that for a microdisc. Given here is the input file for determining the current to the tip at a normalised distance `l`=100 from an insulating substrate. For an entire approach curve, the simulation is run over a range of values of `l`. The conducting substrate case requires the modification of one boundary condition. Note that the simulated current must be normalised by the "infinite" distance tip current in order to match the expression `amphlett_rg10_2` described in § 5.2.1 on page 205 (the slight difference in sheath radius is negligible).

```
GoverningEqs
{
    u
    {
        (x*u_x)_x+(x*u_y)_y = 0
    }
```

```
}
UserExprs
{
    d := 500
    pi := 3.14159265358979323846264338327795
    l := 100
    Rg := 10
}
BCDefs
{
    insulator { flux(u) = 0 }
    collector { u = 0 }
    generator { u = 1 }
}
Boundary
{
electrode:  0       l   collector
            1       l   insulator
            Rg      l   insulator
            Rg      d   generator
            d       d   generator
            d       0   insulator
            0       0   insulator
}
Functionals
{
    d_current := (pi/2)*int(domain,(u_x^2+u_y^2)*x)
    ~d_current := (pi/2)*int(domain,(~u_x^2+~u_y^2)*x)

    amphlett_rg10_2 := 1/(0.40472+1.60185/l+0.58819*exp(-2.37294/l))
}
```

```
Settings { glob_err_tol 0.01 }
AdaptiveControl
{
    elem_err := (pi/2)*abs(int(element,(~u_x^2+~u_y^2-u_x^2-u_y^2)*x))
    elem_err_tol := (glob_err_tol/n_elems)*((pi/2)*abs(int(domain,
                                            (~u_x^2+~u_y^2)*x)))
    glob_err := elem_err_sum/((pi/2)*abs(int(domain,(~u_x^2+~u_y^2)*x)))
}
```

Bibliography

[1] Amphlett, J.; Denuault, G. *J. Phys. Chem. B* **1998,** *102,* 9946.

[2] Bard, A.; Faulkner, L. *Electrochemical Methods;* John Wiley & Sons, Inc.: New York, 1980.

[3] Einstein, A. *Investigations on the Theory of Brownian Movement;* Dover: New York, 1956.

[4] Fick, A. *Philosophical Magazine* **1855,** *10,* 30.

[5] Rosner, D. *Transport Processes in Chemically Reacting Flow Systems;* Dover: New York, 2000.

[6] Acheson, D. *Elementary Fluid Dynamics;* Oxford University Press: Oxford, 1990.

[7] Glauert, M. *J. Fluid. Mech.* **1956,** *1,* 625.

[8] Homann, F. *ZAMN* **1936,** *16,* 153.

[9] Homann, F. *Forsschg. IngWes* **1936,** *7,* 1.

[10] Frössling, N. *Lunds. Univ. Avd.* **1940,** *2,* 35.

[11] Albery, W.; Bruckenstein, S. *J. Electroanal. Chem.* **1983,** *144,* 105.

[12] Southampton Electrochemistry Group, *Instrumental Methods in Electrochemistry;* Ellis Horward: Chichester, 1990.

[13] Greenberg, M. *Advanced Engineering Mathematics;* Prentice Hall: Upper Saddle River, second ed.; 1998.

[14] Byron, Jr., F.; Fuller, R. W. *Mathematics of Classical and Quantum Physics;* Dover: New York, 1992.

[15] Joos, G.; Freeman, I. *Theoretical Physics;* Dover: New York, 1986.

[16] Zienkiewicz, O.; Taylor, R. *The Finite Element Method Vol. 1: The Basis;* Butterworth-Heinemann: London, fifth ed.; 2000.

[17] Ockendon, J.; Howison, S.; Lacey, A.; Movchan, A. *Applied Partial Differential Equations;* Oxford University Press: Oxford, 1999.

[18] Angus, J. N. *Numerical Simulation of Complex Microelectrode Geometries,* Thesis, University of Southampton, United Kingdom, 2002.

[19] Nehari, Z. *Conformal Mapping;* Dover: New York, 1975.

[20] Osborne, A. *Complex Variables and Their Applications;* Addison-Wesley: Harlow, 1999.

[21] Milne-Thomson, L. *Theoretical Hydrodynamics;* Dover: New York, second ed.; 1996.

[22] Amatore, C.; Fosset, B. *J. Electroanal. Chem.* **1992,** *328,* 21.

[23] Galceran, J.; Taylor, S.; Bartlett, P. *J. Electroanal. Chem.* **1999,** *476,* 132.

[24] Rajendran, L.; Sangaranararyanan, M. *J. Phys. Chem. B* **1999,** *103,* 1518.

[25] Bond, A.; Luscombe, D.; Oldham, K.; Zoski, C. *J. Electroanal. Chem.* **1988,** *249,* 1.

[26] Rudolph, M.; Reddy, D. P. *Anal. Chem.* **1994,** *66,* 589A.

[27] Courant, R.; Friedrichs, K.; Lewy, H. *Math. Ann.* **1928,** *100,* 32.

[28] Courant, R.; Friedrichs, K.; Lewy, H. *IBM J. Res. Dev.* **1967,** *11,* 215 (English translation of 1928 paper).

[29] Britz, D. *Digital Simulation in Electrochemistry;* Springer-Verlag: Berlin, 1981.

[30] Press, W.; Teukolsky, S.; Vetterling, W.; Flannery, B. *Numerical Recipes in C;* Cambridge University Press: Cambridge, second ed.; 1992.

[31] Alden, J. *Computational Electrochemistry,* Thesis, University of Oxford, United Kingdom, 1998.

[32] Braess, D. *Finite Elements: Theory, fast solvers, and applications in solid mechanics;* Cambridge University Press: Cambridge, second ed.; 2001.

[33] Harriman, K.; Gavaghan, D.; Houston, P.; Süli, E. "Adaptive Finite Element Simulation of Steady State Currents at Microdisc Electrodes to a Guaranteed Accuracy", Technical Report NA99/19, Oxford University Numerical Analysis Group, 1999.

[34] Harriman, K.; Gavaghan, D.; Houston, P.; Süli, E. *Electrochemistry Communications* **2000,** *2,* 150.

[35] Harriman, K.; Gavaghan, D.; Houston, P.; Süli, E. *Electrochemistry Communications* **2000,** *2,* 157.

[36] Harriman, K.; Gavaghan, D.; Houston, P.; Süli, E. *Electrochemistry Communications* **2000,** *2,* 163.

[37] Harriman, K.; Gavaghan, D.; Houston, P.; Süli, E. *Electrochemistry Communications* **2000,** *2,* 567.

[38] Harriman, K.; Gavaghan, D.; Houston, P.; Süli, E. *Electrochemistry Communications* **2000,** *2,* 576.

[39] Qiu, F. L.; Fisher, A. C. *Electrochemistry Communications* **2000,** *2,* 738.

[40] Øksendal, B. *Stochastic Differential Equations: an introduction with applications;* Springer-Verlag: Berlin, fifth ed.; 1998.

[41] King, G. W. *Indust. and Eng. Chem.* **1951,** *43,* 2475.

[42] Fransaer, J. L.; Penner, R. M. *J. Phys. Chem. B* **1999,** *103,* 7643.

[43] Sagués, F.; Costa, J. M. *J. Chem. Educ.* **1989,** *66,* 502.

[44] Witten, T. A.; Meakin, P. *Phys. Rev. B* **1983,** *28,* 5632.

[45] Nyikos, L.; Pajkossy, T. *Electrochimica Acta* **1986,** *31,* 1347.

[46] Pajkossy, T.; Nyikos, L. *Electrochimica Acta* **1988,** *34,* 171.

[47] Pajkossy, T.; Nyikos, L. *Electrochimica Acta* **1988,** *34,* 181.

[48] Borosy, A. P.; Nyikos, L.; Pajkossy, T. *Electrochimica Acta* **1991,** *36,* 163.

[49] Voss, R. F.; Tomkiewicz, M. *J. Electrochem. Soc.* **1985,** *132,* 371.

[50] Fanelli, N.; Záliš, S.; Pospíšil, L. *J. Electroanal. Chem.* **1989,** *262,* 35.

[51] Fanelli, N.; Záliš, S.; Pospíšil, L. *J. Electroanal. Chem.* **1990,** *288,* 263.

[52] Trigueros, P. P.; Mas, F.; Claret, J.; Sagués, F.; Galceran, J.; Puy, J. *J. Electroanal. Chem.* **1993,** *348,* 221.

[53] Meakin, P.; Sapoval, B. *Phys. Rev. A* **1991,** *43,* 2993.

[54] Sapoval, B.; Gutfraind, R.; Meakin, P.; Keddam, M.; Takenouti, H. *Phys. Rev. E* **1993,** *48,* 3333.

[55] Vilaseca, E.; Trigueros, P. P.; Garcés, J. L.; Mas, F. *J. Electroanal. Chem.* **1998,** *458,* 55.

[56] Nagy, G.; Sugimoto, Y.; Denuault, G. *J. Electroanal. Chem.* **1997,** *433,* 167.

[57] Nagy, G.; Denuault, G. *J. Electroanal. Chem.* **1997,** *433,* 175.

[58] Heerman, L.; Tarallo, A. *J. Electroanal. Chem.* **1998,** *455,* 265.

[59] Cox, D.; Miller, H. *The Theory of Stochastic Processes;* Chapman & Hall: London, 1965.

[60] Muller, M. E. *Ann. Math. Statist.* **1956,** *27,* 569.

[61] Haji-Sheikh, A.; Sparrow, E. *SIAM J. Appl. Math.* **1966,** *14,* 370.

[62] Sabelfeld, K. K. *Monte Carlo Methods in Boundary Value Problems;* Springer-Verlag: Berlin, 1991.

[63] Ogawa, S.; Lécot, C. *Math. Comp. Sim.* **2003,** *62,* 487.

[64] Bard, A.; Mirkin, M., Eds.; *Scanning Electrochemical Microscopy;* Marcel Dekker: New York, 2001.

[65] Fagan, M. J. *Finite Element Analysis: Theory and Practice;* Prentice Hall: Harlow, United Kingdom, 1992.

[66] Fenner, R. T. *Finite Element Methods for Engineers;* Imperial College Press: London, 1975.

[67] Kreysig, E. *Advanced Engineering Mathematics;* John Wiley & Sons: New York, eighth ed.; 1998.

[68] Gerald, C.; Wheatley, P. *Applied Numerical Analysis;* Addison-Wesley: Reading, MA, 1999.

[69] Golub, G.; van Loan, C. F. *Matrix Computations;* John Hopkins University Press: Baltimore, third ed.; 1996.

[70] Demmel, J. W. *Applied Numerical Linear Algebra;* SIAM: Philadephia, PA, 1997.

[71] Barrett, R.; Berry, M.; Chan, T.; Demmel, J.; Donato, J.; Dongarra, J.; Eijkhout, V.; Pozo, R.; Romine, C. *Templates for the Solution of Linear Systems;* SIAM: Philadephia, PA, 1994.

[72] Saad, Y. *Iterative Methods for Sparse Linear Systems;* PWS Publishing Co.: Boston, 1996 (available online).

[73] Gresho, P.; Sani, R. *Incompressible Flow and the Finite Element Method Vol. 1: Advection-Diffusion;* Wiley: Chichester, 1998.

[74] Zienkiewicz, O.; Taylor, R. *The Finite Element Method Vol. 3: Fluid Dynamics;* Butterworth-Heinemann: London, fifth ed.; 2000.

[75] Donea, J.; Huerta, A. *Finite Element Methods for Flow Problems;* Wiley: Chichester, 2003.

[76] Gresho, P.; Sani, R. *Incompressible Flow and the Finite Element Method Vol. 2: Isothermal Laminar Flow;* Wiley: Chichester, 1998.

[77] Leonard, B. A survey of finite differences of opinion on the numerical muddling of the incomprehensible defective confusion equation. In *Finite Element Methods for Convection Dominated Flows*; Hughes, T., Ed.; ASME: New York, 1979.

[78] Ilinca, F.; Hétu, J.-F.; Pelletier, D. *Comput. Methods Appl. Mech. Engrg.* **2000,** *188,* 235.

[79] Stroustrup, B. *The C++ Programming Language;* Addison-Wesley: Reading, MA, third ed.; 1997.

[80] Kernighan, B.; Ritchie, D. *The C Programming Language;* Prentice Hall: New Jersey, second ed.; 1988.

[81] Josuttis, N. M. *The C++ Standard Library;* Addison-Wesley: Reading, MA, 1999.

[82] "IA-32 Intel Architecture Optimization: Reference Manual", Intel Corp. (available online).

[83] "IA-32 Intel Architecture Software Developer's Manual Vol. 1: Basic Architecture", Intel Corp. (available online).

[84] Bennett, J. *Introduction to Compiling Techniques: a first course using ANSI C, LEX and YACC;* McGraw Hill: London, second ed.; 1996.

[85] Chomsky, N. *IRE Transactions on Information Theory* **1956,** 113.

[86] Chomsky, N. *Information and Control* **1959,** 91.

[87] Bartlett, P.; Taylor, S. *J. Electroanal. Chem.* **1998,** *101,* 49.

[88] Babuška, I.; Szabo, B. *Int. J. Num. Meth. Eng.* **1982,** *18,* 323.

[89] Isaacson, E.; Keller, H. *Analysis of Numerical Methods;* Dover: New York, 1994.

[90] Kincaid, D.; Cheney, W. *Numerical Analysis;* Brooks/Cole: Pacific Grove, CA, second ed.; 1996.

[91] Carey, G.; Oden, J. *Finite Elements: Computational Aspects;* Prentice-Hall: Englewood Cliffs, NJ, 1984.

[92] Babuška, I.; Aziz, A. *SIAM J. Numer. Anal.* **1976,** *13,* 214.

[93] O'Rourke, J. *Computational Geometry in C;* Cambridge University Press: Cambridge, second ed.; 1998.

[94] Edelsbrunner, H. *Geometry and Topology for Mesh Generation;* Cambridge University Press: Cambridge, 2001.

[95] Fortune, S. *Algorithmica* **1987,** *2,* 153.

[96] Lawson, C. *Software for C^1 Surface Interpolation;* Academic Press: New York, 1977.

[97] Bowyer, A. *Computer Journal* **1981,** *2,* 162.

[98] Watson, D. *Computer Journal* **1981,** *2,* 167.

[99] Shewchuk, R. "Lecture Notes on Delaunay Mesh Generation", 1999 (University of California at Berkeley).

[100] Bank, R. *PLTMG: A Software Package for Solving Elliptic Partial Differential Equations. Users' Guide 7.0.;* SIAM: Philadephia, PA, 1994.

[101] Ruppert, J. *Results on Triangulation and High Quality Mesh Generation,* Thesis, University of California, 1992.

[102] Bern, M.; Eppstein, D.; Gilbert, J. *J. Computer and System Sciences* **1994,** *48,* 384.

[103] Loudon, K. *Mastering Algorithms with C;* O'Reilly: Sebastopol, CA, 1999.

[104] Babuška, I.; Rheinboldt, W. *SIAM J. Numer. Anal.* **1978,** *15,* 736.

[105] Babuška, I.; Rheinboldt, W. *Int. J. Num. Meth. Eng.* **1978,** *12,* 1597.

[106] Babuška, I.; Miller, A. *Int. J. Num. Meth. Eng.* **1984,** *20,* 1085.

[107] Babuška, I.; Miller, A. *Int. J. Num. Meth. Eng.* **1984,** *20,* 1111.

[108] Babuška, I.; Miller, A. *Int. J. Num. Meth. Eng.* **1984,** *20,* 2311.

[109] Babuška, I.; Strouboulis, T. *The Finite Element Method and its Reliability;* Oxford University Press: Oxford, 2000.

[110] Eriksson, K.; Johnson, C. *SIAM J. Numer. Anal.* **1991,** *28,* 43.

[111] Eriksson, K.; Johnson, C. *SIAM J. Numer. Anal.* **1995,** *32,* 706.

[112] Eriksson, K.; Johnson, C. *SIAM J. Numer. Anal.* **1995,** *32,* 1729.

[113] Eriksson, K.; Johnson, C. *SIAM J. Numer. Anal.* **1995,** *32,* 1750.

[114] Eriksson, K.; Johnson, C. *SIAM J. Numer. Anal.* **1998,** *35,* 1315.

[115] Rannacher, R. Error control in finite element computations. In *Proc. Summer School Error Control and Adaptivity in Scientific Computing*; Bulgak, H.; Zenger, C., Eds.; Kluwer Academic Publishers: Dordrecht, 1998.

[116] Rannacher, R. Duality techniques for error estimation and mesh adaptation in finite element methods. In *Adaptive Finite Elements in Linear and Nonlinear Solid and Structural Mechanics*; Stein, E., Ed.; Springer: New York, 2002.

[117] Rannacher, R. *Comput. Mechanics* **1998,** *21,* 123-133.

[118] Rannacher, R. *Comput. Meth. Appl. Mech. Eng.* **1999,** *176,* 333-361.

[119] Bangerth, W.; Rannacher, R. *Adaptive Finite Element Methods for Differential Equations;* Birkhäuser Verlag: Basel, 2003.

[120] Becker, R.; Rannacher, R. *Acta Numerica* **2001,** 1-102.

[121] Cantrell, C. *Modern Mathematical Methods for Physicists and Engineers;* Cambridge University Press: Cambridge, 2000.

[122] Amphlett, J. L. *Numerical Simulation of Microelectrodes,* Thesis, University of Southampton, United Kingdom, 2000.

[123] Babuška, I.; Strouboulis, T.; Upadhyay, C. *Int. J. Num. Meth. Eng.* **1997,** *40,* 2521.

[124] Hinton, C.; Campbell, J. *Int. J. Num. Meth. Eng.* **1974,** *8,* 461.

[125] Zienkiewicz, O.; Zhu, J. *Int. J. Num. Meth. Eng.* **1987,** *24,* 337.

[126] Zhang, Z. *Mathematics of Computation* **1996,** *65,* 1431.

[127] Zhang, Z. *Mathematics of Computation* **1998,** *67,* 541.

[128] Zhang, Z. *Mathematics of Computation* **2000,** *69,* 141.

[129] Zhang, Z. *Mathematics of Computation* **2002,** *71,* 1421.

[130] Zhu, Q. A survey of superconvergence techniques in finite element methods. In *Finite Element Methods, superconvergence post-processing, and a-posteriori estimates*; Krizek, M.; Neittaanmäki, P.; Sternberg, R., Eds.; Marcel Dekker: New York, 1998.

[131] Zhu, Q. *J. Comput. Math.* **2000,** *18,* 545.

[132] Schatz, A.; Sloan, I.; Wahlbin, L. *SIAM J. Numer. Anal.* **1996,** *33,* 877.

[133] Babuška, I.; Strouboulis, T.; Upadhyay, C.; Gangaraj, S. *Numerical Methods for PDEs* **1996,** *12,* 347.

[134] Appel, K.; Haken, W. The four-color problem. In *Mathematics Today*; Steen, L., Ed.; Springer: Berlin, 1978.

[135] Gullberg, J. *Mathematics From the Birth of Numbers;* W.W. Norton and Company: New York, 1997.

[136] Zienkiewicz, O.; Zhu, J. *Int. J. Num. Meth. Eng.* **1992,** *33,* 1331.

[137] Zienkiewicz, O.; Zhu, J. *Int. J. Num. Meth. Eng.* **1992,** *33,* 1365.

[138] Zienkiewicz, O.; Zhu, J. *Comp. Meth. Appl. Mech. Eng.* **1992,** *101,* 207.

[139] Abercrombie, S.; Denuault, G. *Electrochem. Comm.* **2003,** *5,* 647.

[140] Harriman, K.; Gavaghan, D. J.; Süli, E. *Electrochem. Comm.* **2003,** *5,* 519.

[141] Seddon, B.; Shao, Y.; Fost, J.; Girault, H. *Electrochimica Acta* **1994,** *39,* 2377.

[142] Kounaves, S.; Deng, W.; Hallock, R.; Kovacs, G.; Storment, C. *Anal. Chem.* **1994,** *66,* 418.

[143] Ferrigno, R.; Brevet, P.; Girault, H. *Electrochimica Acta* **1997,** *42,* 1895.

[144] Amatore, C.; Fosset, B. *Anal. Chem.* **1991,** *63,* 306.

[145] Abramowitz, M.; Stegun, I. A. *Handbook of Mathematical Functions;* Dover: New York, 1968.

[146] Phillips, C. *J. Electroanal. Chem.* **1990,** *437,* 255.

[147] Ockendon, H.; Ockendon, J. *Viscous Flow;* Cambridge University Press: Cambridge, 1995.

[148] Lévêque, M. *Ann. Mines. Mem. Ser.* **1928,** *12/13,* 201.

[149] Levich, V. *Physicochemical Hydrodynamics;* Prentice-Hall: Englewood Cliffs, N.J., 1962.

[150] Ackerberg, R.; Patel, R.; Gupta, S. *J. Fluid. Mech.* **1977,** *86,* 49.

[151] Newman, J. *Electroanalytical Chemistry* **1973,** *6,* 279.

[152] Aoki, K.; Tokuda, K.; Matsuda, H. *J. Electroanal. Chem.* **1987,** *217,* 33.

[153] Mirkin, M.; Horrocks, B. *Anal. Chim. Acta* **2000,** 119.

[154] Bard, A.; Fan, F.-R.; Mirkin, M. Electroanalytical Chemistry vol. 18. In ; Bard, A., Ed.; Marcel Dekker: New York, 1993.

[155] Kwak, J.; Bard, A. *Anal. Chem.* **1989,** *61,* 1221.

[156] Fang, Y.; Leddy, J. *Anal. Chem.* **1995,** *67,* 1259.

[157] Shao, Y.; Mirkin, M. *J. Phys. Chem. B* **1998,** *102,* 9915.

[158] Mirkin, M. Scanning Electrochemical Microscopy. In ; Bard, A.; Mirkin, M., Eds.; Marcel Dekker: New York, 2001; Chapter 5.

[159] Evans, S. A. G. *Mesoporous Platinum Microdisc Electrodes And The Detection Of Hydrogen Peroxide In Analytical Chemistry And Scanning Electrochemical Microscopy,* Thesis, University of Southampton, United Kingdom, 2003.

[160] "PDEase2. Version 2.5.2. Reference Manual and Application Notes", SPDE Inc.

[161] Melenk, J.; Schwab, C. *SIAM J. Numer. Anal.* **1998,** *35,* 1520.

[162] Cockburn, B.; Karniadakis, G.; Shu, C.-W. The Development of Discontinuous Galerkin Methods. In *Discontinuous Galerkin Finite Element Methods*; Cockburn, B.; Karniadakis, G.; Shu, C.-W., Eds.; Springer-Verlag: Berlin, 1999.

[163] Wittstock, O. S. G. *J. Phys. Chem. B* **2002,** *106,* 7499.

[164] Sklyar, O.; Ufheil, J.; Heinze, J.; Wittstock, G. *Electrochim. Acta* **2003,** .

[165] Foley, J.; van Dam, A.; Feiner, S.; Hughes, J. *Computer Graphics: Principles and Practice (in C);* Addison-Wesley: Reading, Massachusetts, second ed.; 1996.

[166] Shewchuk, R. "An Introduction to the Conjugate Gradient Method Without the Agonizing Pain", 1994 (University of California at Berkeley).

[167] Fox, C. *An Introduction to the Calculus of Variations;* Dover: New York, 1987.

[168] Pauling, L.; Wilson, E. B. *Introduction to Quantum Mechanics;* Dover: New York, 1985.

[169] Ram-Mohan, L. R. *Finite Element and Boundary Element Applications in Quantum Mechanics;* Oxford University Press: Oxford, 2002.

[170] Carslaw, H.; Jaeger, J. *Conduction of Heat in Solids;* Oxford University Press: Oxford, second ed.; 1959.

[171] Saito, Y. *Rev. Polarogr.* **1968,** *15,* 177.

[172] Crank, J.; Furzeland, R. *J. Inst. Maths. Applics.* **1977,** *20,* 355.

[173] Lebedev, N. *Special Functions and Their Applications;* Dover: New York, 1972.

[174] Lehmer, D. *American Mathematical Monthly* **1938,** *45,* 657.

[175] Watson, G. *A Treatise on the Theory of Bessel Functions;* Cambridge University Press: Cambridge, second ed.; 1966.

[176] Faber, V.; Manteuffel, T. *SIAM J. Numer. Anal.* **1984,** *21,* 352.

A Note on References

Republished books (for instance those now published by Dover Publications) are given in their modern forms.

Index