Parallel iterative methods in semiconductor device modelling

Coomer, Rob

Award date:
1994

Awarding institution:
University of Bath

Parallel Iterative Methods in Semiconductor Device Modelling

Submitted by

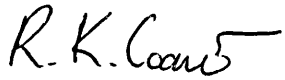
Rob Coomer

for the degree of PhD

of the

University of Bath

1994



R.K. Coomer

ProQuest LLC
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106-1346

Abstract

The discovery and subsequent development of semiconducting materials has been a subject area that has seen rapid changes in recent decades. The ability to numerically model this behaviour allows the efficient design of many new applications without the need for expensive test equipment. The system of equations modelling the electrical behaviour of a stationary semiconducting device is established in Chapter 1. In Chapter 2 discretisation schemes in both one and two dimensions are introduced. Existing iterative solution techniques (including Gummel's method) and also a novel continuation scheme are then discussed. In Chapter 3 the problem of solving the semilinear equation arising in the calculation of the electrostatic potential is addressed. A certain quasi-Newton method which computes sequences of upper and lower solutions, and converges quadratically from any starting upper and lower solution pair is introduced. In Chapters 4 and 5 the Lipschitz constant of the fixed point map for a version of Gummel's method is shown to be independent of h in one dimension and grows only logarithmically in $1/h$ (as the mesh diameter $h \to 0$) in two dimensions, provided the meshes are refined in a regular manner. Furthermore, in Chapter 4 results are provided which show that the computed potential exhibits sharp layers, interior to the domain of computation. In Chapter 6 domain decomposition preconditioned iterative methods for the types of linear systems arising in Chapter 2 are discussed. The effect of this type of domain decomposition technique on a certain class of model problems is also considered. It is shown that, in some of these types of

problems, it is possible to achieve acceptable convergence without the need to precondition. In Chapter 7 implementation issues arising from such an iterative method on a massively parallel architecture are examined. Finally, in Chapter 8, numerical results for the semiconductor system on the MasPar MP-1 are given.

Acknowledgements

During the course of my research I have been lucky enough to have the support of many friends and colleagues. In particular I would like to thank Dr. Ivan Graham for his enduring and excellent supervision. My gratitude should also be expressed to Prof. Alastair Spence, Dr. Peter Jimak and Dr. Kevin Parrott, all of whom have contributed ideas, comments and encouragement.

I am indebted to the Science and Engineering Research Council for their financial support during the three years of my studies. I would also like to thank Normalair-Garrett Limited who have invested time and money in supporting me through both my undergraduate and postgraduate training.

I must thank my parents and grandparents for their support and enthusiasm. I would also like to mention some of the characters that have made my three years as a postgraduate a time that I will look back on with fond memories. With Messrs. Currie and Webber I have learnt to ski, surf, scuba dive and order taxi cabs at any hour of the night or day. My thanks go to the many and varied inhabitants of 1W3.6, who have always allowed me the privilege of the window seat next to the radiator. Of the many hard-working postgraduates in the School of Mathematics, Rob Douglas and Gabriel Lord should be singled out as two colleagues who were never too busy for a cup of tea in the Physics Servery philosophy department. Finally, I would like to thank my girlfriend Helen for the (almost continuous) patience she has shown in waiting for me to complete this thesis.

Publication details

Two joint papers based on some of the work contained in this thesis have been submitted for publication [15], [14].

Contents

List of Figures

Chapter 1

Semiconductor device modelling

1.1 Introduction

The discovery and subsequent development of semiconducting materials has been a subject area that has seen rapid changes in recent decades. Nearly every aspect of modern life is affected by semiconducting technologies. It is the unique electrical properties of semiconducting material that make it such a useful component in electrical applications. Although these properties are readily used in a diverse range of products, the laws which govern these properties make the accurate prediction of semiconductor behaviour a highly non–trivial task. The ability to numerically model this behaviour (using semiconductor technology in the form of powerful computers in the process) allows the efficient design of many new applications without the need for expensive test equipment.

1.2 Physical and electrical properties

1.2.1 Semiconducting materials

Semiconductors typically start life as single crystals of pure silicon (Si) or germanium (Ge) which are subsequently processed to obtain desired electrical prop-

erties. At very low temperatures pure single crystals of semiconductor act as insulators. The atoms of silicon or germanium contain 4 outer shell *electrons*. Electrons are negatively charged. Each of these may form a covalent bond with any of the neighbouring atoms and hence silicon and germanium form regular tetrahedral, diamond–like structures. When the temperature is raised, energy is randomly distributed to the atoms and electrons in the lattice. The most energetic electrons, forming the covalent bonds between the atoms, will escape these bonds and become free to conduct electricity. Figure 1.1 demonstrates this effect. The remarkable feature of semiconductors is that the gaps left behind by the escaping electrons are free to move within the bonds. These gaps are called *holes* and have a positive charge equal in magnitude to the charge on an electron. Hence, although the freed electrons are able to conduct electricity in the spaces between the bonds, the holes they leave behind can also conduct electricity, quite independently, within the bonds. The net effect is that a semiconductor contains a positively charged cloud and a negatively charged cloud, both of which can conduct electricity (at least for a limited time). The electrons and holes are collectively known as *carriers*.

The phenomenon of releasing electrons and holes can be artificially accelerated by *doping* the crystal with impurities. There are two basic types of material that can be created in this way.

n–type material. By adding small amounts of pentavalent elements (i.e. elements with 5 outer shell electrons, like phosphor for instance), extra outer shell electrons can be introduced to the lattice. This type of material is negatively doped.

p–type material. By adding small amounts of trivalent elements (i.e. elements with 3 outer shell electrons, like aluminium for instance), a shortage of outer shell electrons can be produced in the lattice. This results in an excess of holes. This type of material is positively doped.

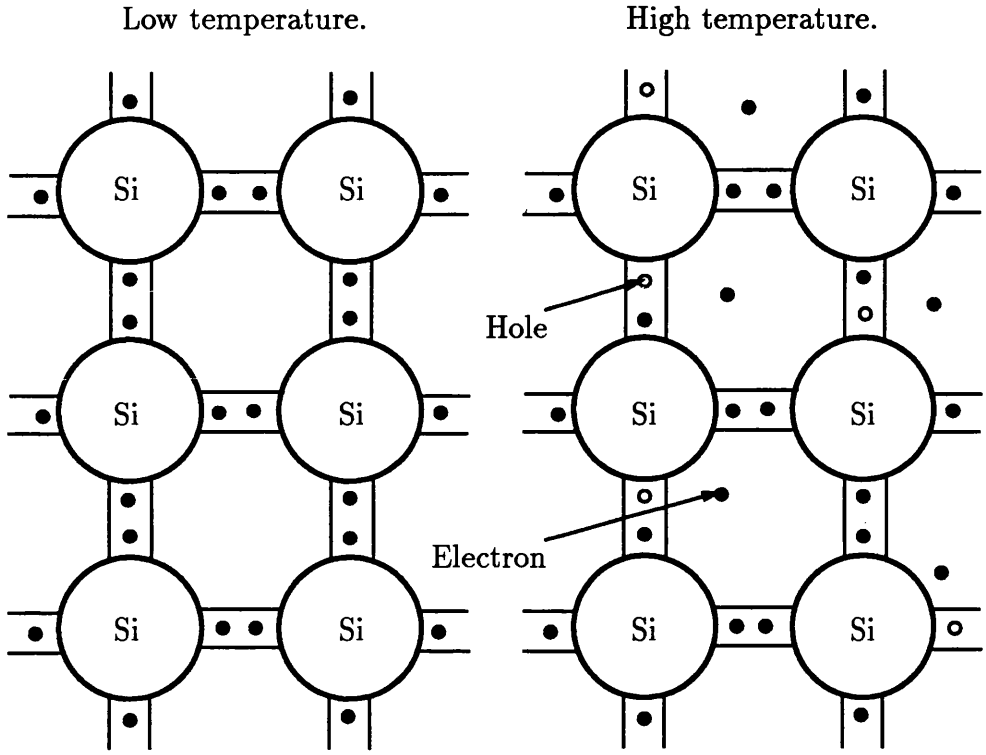


Figure 1.1: Silicon crystal lattice.

Semiconducting devices are therefore manufactured by doping different regions of silicon or germanium crystals with the appropriate impurities to achieve the desired electrical properties.

The thrust of this thesis is concerned with the simulation of the electrical properties of semiconducting devices and therefore assumes that the device has an existing, known doping profile. We shall now use the example of a simple p–n diode to illustrate the important physical processes which occur within all semiconducting devices.

1.2.2 Simple p–n diode

The simplest of semiconducting devices is a p-n junction. This is a single crystal of semiconducting material with a transition from p– to n–type material. We show this arrangement in Figure 1.2. Here there are metallic contacts attached at either end of the device.

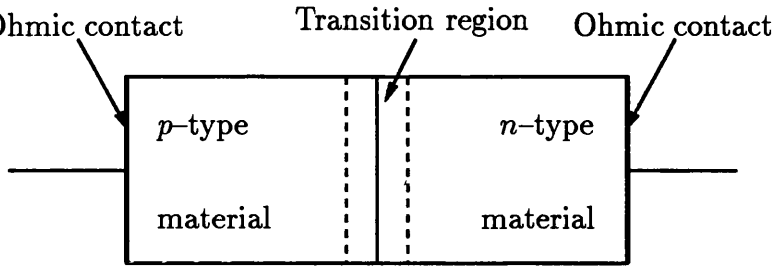


Figure 1.2: Simple p–n diode.

With zero applied potential, the p–n junction gives rise to two types of current.

Diffusion current The large density gradient of electrons at the doping interface means that they tend to diffuse from the n–type to the p– type region. Likewise there will be a hole diffusion process from the p–type to n–type material. Since conventional current flow is in the direction of hole movement, both the diffusive processes give rise to a *diffusion current* from the p– to n–type region.

Drift current The diffusion process creates an electric field across the junction which in turn causes a *drift current* of the carriers in the opposite directions.

In thermal equilibrium the diffusion and drift currents will balance. This p–n junction will exhibit diode properties once a electric potential is applied across the contacts. Figure 1.3 illustrates the two possible configurations.

1. **Forward bias** The application of a positive potential at the p–contact and a negative potential at the n–contact will cause both sets of carriers to converge on the transition region in the centre of the device. Here a *recombination* process will take place with the electrons and holes recombining back into the lattice. Each time this happens a hole and electron disappear together with the release of energy, for example, in the form of heat or light. This will allow large currents to flow in the device as recombination requires relatively little energy to occur. Clearly, in this configuration, recombination is a very important physical process.
2. **Reverse bias** Here a negative potential is applied to the p–contact and a positive potential to the n–contact. Hence the tendency is for the carriers to move towards their respective contacts. This cannot continue indefinitely, since

in order to have a steady flow of holes to the left these must be a supplied across the junction from the n–type material. As there are very few holes in the n–type material, nominally zero current flows. However, in practice, a very small current flows due to the process of *generation* of holes and electrons taking place at the transition region. This generation is caused by thermal energy and hence the reverse bias current should increase with temperature. The current should be independent of the magnitude of the reverse bias. However, an extremely large applied bias will cause a breakdown current to flow. This effect is usually not desired in practice.

At the semiconductor metal contacts very large apparent recombination rates are required. Hence although recombination/generation rates may be very small within the crystal, there is a need for infinite recombination/generation at the contacts. Such contacts are called *ohmic.*

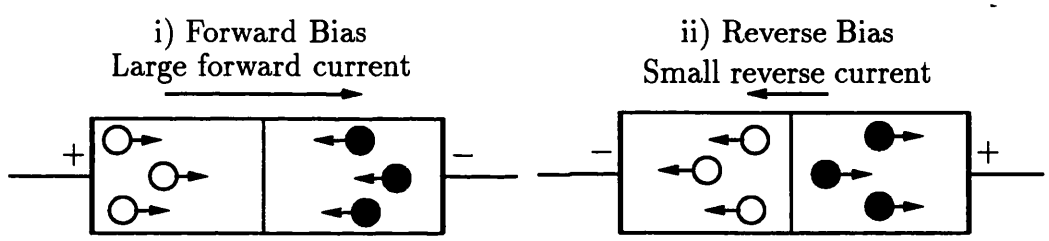


Figure 1.3: Current flow through p–n diode.

Clearly then, a semiconducting device should be modelled by a drift–diffusion system with an appropriate term to model the important physical process of recombination/generation. For a much more comprehensive study of the physical and electrical properties of semiconducting devices we refer the reader to [68], [55], [73]. The mathematical analysis of some of these models is available in [56].

1.3 Derivation of the model

Semiconductor devices occupy simply connected domains which are subsets of $\mathbb{R}^3$. They consist of a semiconducting part and in the case of a metal–oxide semiconductor (MOS), one or more thin adjacent oxide domains. Maxwell's equations state

$$\nabla \wedge H = J + \frac{\partial C}{\partial t}, \tag{1.3.1}$$

$$\nabla \wedge E = -\frac{\partial B}{\partial t}, \tag{1.3.2}$$

$$\nabla . C = \rho, \tag{1.3.3}$$

$$\nabla . B = 0, \tag{1.3.4}$$

where we have the convention

E – electric field vector,

C – displacement vector,

H – magnetic field vector,

B – induction vector,

J – conduction current density,

ρ – electric charge density,

$\boldsymbol{x} \in \mathbb{R}^3$ – independent space variable,

$t \geq 0$ – time variable.

The electric field and the electric displacement are related by

$$C = \epsilon E, \tag{1.3.5}$$

where ϵ is the permittivity of the medium. We assume ϵ to be time independent and spatially homogeneous. We shall also regard the material to be isotropic in which case ϵ can be regarded as a scalar. We shall use the following physical

constants,

$$\begin{aligned} q \text{ elementary charge} &= 1.602189 \times 10^{-19} \text{ As}, \\ \epsilon_v \text{ permittivity constant in a vacuum} &= 8.854188 \times 10^{-14} \text{ As/Vcm}, \\ c \text{ speed of light in a vacuum} &= 2.997925 \times 10^{10} \text{ cm/s}, \\ K_B \text{ Boltzmann's constant} &= 1.380662 \times 10^{-23} \text{ VAs/K}. \end{aligned}$$

Poisson's equation gives

$$B = \nabla \wedge A, \tag{1.3.6}$$

where A is the vector potential. Inserting (1.3.6) into (1.3.2) we obtain

$$\nabla \wedge \left(E + \frac{\partial A}{\partial t} \right) = 0. \tag{1.3.7}$$

Since a sufficiently smooth, vortex–free vector field, which is defined in a simply connected domain, is a gradient field, we have

$$E = -\frac{\partial A}{\partial t} - \nabla \psi, \tag{1.3.8}$$

for some scalar potential ψ. From (1.3.3), (1.3.5) and (1.3.8) we have

$$\epsilon \left(\frac{\partial}{\partial t} \nabla . A + \Delta \psi \right) = -\rho. \tag{1.3.9}$$

In order to make (1.3.9) invariant under the Lorentz transformation we set

$$\nabla . A = -\frac{1}{c^2} \frac{\partial \psi}{\partial t}. \tag{1.3.10}$$

This is known as the Lorentz convention and is adopted in, for example, [48]. Hence (1.3.9) becomes

$$-\frac{\epsilon}{c^2} \frac{\partial^2 \psi}{\partial t^2} + \epsilon \Delta \psi = -\rho. \tag{1.3.11}$$

If we assume that the speed of light, c, is large in comparison to the propagation velocities in the device, then the first term in (1.3.11) can be neglected. Hence we have

$$\epsilon \Delta \psi = -\rho. \tag{1.3.12}$$

In the semiconducting region the space charge density can be written as

$$\rho = q(p - n + d), \tag{1.3.13}$$

where

n – electron concentration,

p – hole concentration,

d – doping profile.

The oxide is assumed charge neutral, i.e. $\rho = 0$. (1.3.12) and (1.3.13) combine to create the *potential equation*

$$\epsilon \Delta \psi = q(n - p - d). \tag{1.3.14}$$

Now (1.3.1) and (1.3.3) imply

$$0 = \nabla . J + \frac{\partial \rho}{\partial t}. \tag{1.3.15}$$

We now split the conduction current density into that caused by electrons, J_n, and that caused by holes, J_p, i.e. $J = J_n + J_p$. We also assume the doping profile to be time invariant, i.e. $\partial d/\partial t = 0$. Then using these assumptions together with (1.3.13) and (1.3.15), we obtain

$$-\nabla . J_p - q\frac{\partial p}{\partial t} = \nabla . J_n - q\frac{\partial n}{\partial t}. \tag{1.3.16}$$

By setting both sides of (1.3.16) to qr we obtain

$$\nabla . J_n - q\frac{\partial n}{\partial t} = qr, \tag{1.3.17}$$

$$\nabla . J_p + q\frac{\partial p}{\partial t} = -qr. \tag{1.3.18}$$

By inspection of the left hand sides of (1.3.17), (1.3.18), we can see that r can be physically interpreted as the difference of the rate at which electron–hole carrier

pairs recombine and the rate at which they are generated. The net effect results in the appropriate flux of electrons and holes within the device. r is called the recombination/generation rate. Generation occurs when $r < 0$ and recombination occurs when $r > 0$. (1.3.17) is thus the *electron continuity equation* and (1.3.18) is the *hole continuity equation.*

Current relations We shall give a phenomenological derivation of the current relations. The two main sources of current flow in a device are diffusion and drift. We denote the diffusion current densities by J_n^{diff}, J_p^{diff} and the drift current densities by J_n^{drift}, J_p^{drift}. Hence

$$J_n = J_n^{diff} + J_n^{drift}, \tag{1.3.19}$$

$$J_p = J_p^{diff} + J_p^{drift}. \tag{1.3.20}$$

The diffusion flux densities are proportional to the gradients of the corresponding particle concentrations.

$$J_n^{diff} = qD_n \nabla n, \tag{1.3.21}$$

$$J_p^{diff} = -qD_p \nabla p. \tag{1.3.22}$$

The signs on the right hand sides are chosen such that the diffusion coefficients D_n, D_p are positive.

The drift current densities are defined as the products of the charge per particle, the corresponding carrier concentration and the average drift velocities v_n^d and v_p^d.

$$J_n^{drift} = -qnv_n^d, \tag{1.3.23}$$

$$J_p^{drift} = qpv_p^d. \tag{1.3.24}$$

Drift directions are assumed parallel to the electric field, holes drift in the same direction as the electric field and electrons drift in the opposite direction. At

moderate field strengths we can assume that the drift velocities are proportional to the electric field. Hence

$$v_n^d = -\mu_n E, \qquad v_p^d = \mu_p E. \tag{1.3.25}$$

(1.3.19), (1.3.21), (1.3.23) and (1.3.25) combine to give

$$J_n = qD_n \nabla n + q\mu_n n E. \tag{1.3.26}$$

Similarly we have

$$J_p = -qD_p \nabla p + q\mu_p p E. \tag{1.3.27}$$

Einstein showed the relationship

$$D_n = U_T \mu_n, \qquad D_p = U_T \mu_p, \tag{1.3.28}$$

where U_T is the thermal voltage. For a fuller discussion of Einstein's relations, (1.3.28), see, for example, [70]. This combined with (1.3.8) and (1.3.26) results in the *electron current relation*

$$J_n = q\mu_n (U_T \nabla n - n \nabla \psi). \tag{1.3.29}$$

Similarly we have the *hole current relation*

$$J_p = -q\mu_p (U_T \nabla p + p \nabla \psi). \tag{1.3.30}$$

Substitution of (1.3.29), (1.3.30) into (1.3.17), (1.3.18) yields the *transient* continuity equations. We will be concerned only with the steady state versions of these, namely

$$\mu_n \nabla.(U_T \nabla n - n \nabla \psi) = r, \tag{1.3.31}$$

$$\mu_p \nabla.(U_T \nabla p + p \nabla \psi) = r. \tag{1.3.32}$$

The three equations (1.3.14), (1.3.31) and (1.3.32) governing the electrical behaviour of a stationary semiconducting device, are very badly scaled. We are also confronted with the problem that the carrier concentrations, n and p, are

typically of the order of $10^{16}\mathrm{m}^{-3}$. The combination of these two facts makes the equations very difficult to solve numerically. Clearly the equations need to be scaled in some way. Also we could try to employ a change of variables in order to reduce the magnitude of the unknowns in the problem. In the next section we shall adopt the so–called *quasi–Fermi levels.* These are just one of a choice of possible variable changes. See [62] or [40, Chapter 3] for a survey of possible choices and discussion of their advantages and disadvantages. Furthermore we shall scale the governing equations in an analogous manner to [62].

1.4 Scaling and the quasi–Fermi levels

We shall first scale the equations by the value

$$\tilde{d} := \max |d|.$$

The system then becomes

$$-\frac{\epsilon U_T}{q\tilde{d}}\Delta\left(\frac{\psi}{U_T}\right) = \frac{p}{\tilde{d}} - \frac{n}{\tilde{d}} + \frac{d}{\tilde{d}}, \tag{1.4.33}$$

$$\mu_n \nabla.\left(\nabla\left(\frac{n}{\tilde{d}}\right) - \frac{n}{\tilde{d}}\nabla\left(\frac{\psi}{U_T}\right)\right) = \frac{r}{\tilde{d}U_T}, \tag{1.4.34}$$

$$\mu_p \nabla.\left(\nabla\left(\frac{p}{\tilde{d}}\right) + \frac{p}{\tilde{d}}\nabla\left(\frac{\psi}{U_T}\right)\right) = \frac{r}{\tilde{d}U_T}. \tag{1.4.35}$$

(1.4.33)–(1.4.35) are then further scaled by the device diameter, l. We write

$$\tilde{\psi}(\boldsymbol{x}) := \psi(l\boldsymbol{x}), \quad \tilde{n}(\boldsymbol{x}) := n(l\boldsymbol{x}), \quad \tilde{p}(\boldsymbol{x}) := p(l\boldsymbol{x}).$$

We then redefine

$$\psi = \tilde{\psi}/U_T, \quad n = \tilde{n}/\tilde{d}, \quad p = \tilde{p}/\tilde{d}, \quad d = d/\tilde{d}.$$

The scaled equations are then (in terms of the new variables)

$$-\lambda^2\Delta\psi = p - n + d, \tag{1.4.36}$$

$$\mu_n \nabla.(\nabla n - n\nabla\psi) = \frac{l^2 r}{\tilde{d} U_T}, \tag{1.4.37}$$

$$\mu_p \nabla.(\nabla p + p\nabla\psi) = \frac{l^2 r}{\tilde{d} U_T}. \tag{1.4.38}$$

In (1.4.36), $\lambda = l^{-1}\sqrt{(\epsilon U_T/q\tilde{d})}$ is called the Debeye length. We now implicitly define the quasi–Fermi levels. Because they have a more restrictive range than n or p, they are in some sense more appropriate for the numerical treatment of (1.4.36)–(1.4.38). We write

$$n = \frac{n_i}{\tilde{d}} \exp(\psi - v), \tag{1.4.39}$$

$$p = \frac{n_i}{\tilde{d}} \exp(w - \psi). \tag{1.4.40}$$

This defines the electron and hole quasi–Fermi levels, v and w respectively. Also n_i is the *intrinsic concentration.* When a semiconductor is in thermal equilibrium there is a dynamic balance between the recombination and generation rates. Thus $r = 0$ holds and the equilibrium carrier concentrations, n_e, p_e are related by the mass–action law

$$n_e p_e = n_i^2.$$

Hence

$$\nabla n - n\nabla\psi = -\frac{n_i}{\tilde{d}} \exp(\psi - v)\nabla v,$$

$$\nabla p + p\nabla\psi = -\frac{n_i}{\tilde{d}} \exp(w - \psi)\nabla w,$$

and the equations, (1.4.36)–(1.4.38), become

$$-\lambda^2 \Delta\psi + \delta\{\exp(\psi - v) - \exp(w - \psi)\} = d, \tag{1.4.41}$$

$$-\nabla.(\exp(\psi - v)\nabla v) = \sigma\rho_v r, \tag{1.4.42}$$

$$\nabla.(\exp(w - \psi)\nabla w) = \sigma\rho_w r. \tag{1.4.43}$$

Here we have written $\rho_v = 1/\mu_n$, $\rho_w = 1/\mu_p$, $\delta = n_i/\tilde{d}$ and $\sigma = l^2/n_i U_T$.

The boundary conditions for (1.4.41)–(1.4.43) are given at ohmic contacts by the Dirichlet criteria

$$v = w = V_{appl}/U_T =: \alpha, \tag{1.4.44}$$

$$\delta\{\exp(\psi - v) - \exp(w - \psi)\} - d = 0, \tag{1.4.45}$$

where V_{appl} is the applied voltage at the contact. Regions of the boundary that are insulated will have Neumann conditions imposed on them.

The choice of modelling function for the recombination/generation rate is not clear–cut. In this thesis we shall only consider the *Shockley–Read–Hall* model which (in appropriately scaled form) is given by

$$r(\psi, v, w) = \frac{n_i}{\tau} \frac{\exp(w - v) - 1}{\exp(w - \psi) + \exp(\psi - v) + 2} \quad , \tag{1.4.46}$$

where $\tau = 10^{-6}$ (see for example [48], [49]).

The drift–diffusion equations (1.4.41)–(1.4.43) are three coupled, elliptic, non-linear partial differential equations (PDE's) in gradient form. Having introduced the system that we will be studying, we now discuss some of the recent work in this field and highlight some of the background to the methods we shall be working with.

1.5 Background

In this section we give a brief background to the work presented in this thesis and related topics. Since this thesis brings together quite a diverse range of subjects from the field of numerical analysis, no attempt is made here at a complete literature survey. We prefer instead to provide overviews of each of the subject areas in the relevant chapters.

There are various methods of discretisation for the system (1.4.41)–(1.4.43), which governs the steady state electrical behaviour of a semiconducting device. There are basically three choices: the finite difference method, the finite volume

(box) method and the finite element method. Along with the standard finite element method, there are several mixed finite element methods which have better current conservation properties [11], [10]. Although these approaches are different in their origins, they often lead to systems of discrete equations with similar qualitative properties. In this thesis we shall be predominantly concerned with the finite element method. In many practical applications the engineer is interested in the current flowing through through the device. For this reason we consider discretisations which in some sense exhibit current conservation properties. Such discretisations are discussed in detail in Chapter 2.

Once the equations have been discretised, there are many possible schemes for iterative solution of the resulting algebraic system. Some of these schemes are described in Chapter 2. In the literature, many of these iterative schemes have been appraised by carrying out an analysis of the corresponding scheme applied to the undiscretised equations (e.g. in most of the analysis of [40]–[43]). In this thesis we consider only iterative methods applied directly to the discrete equations, since this is what must happen in practice. By far the most commonly used is the one due to Gummel [28]. This is nothing more than a nonlinear block Gauss–Seidel iteration applied to (discretisations of) (1.4.41)–(1.4.43). Gummel's method in its continuous or discrete form has been well–studied and proved to converge for small enough applied bias ([37], [38], [40], [41], [42], [43]). In practice Gummel's method is still used far away from equilibrium, but usually in conjunction with continuation in some suitable parameter (e.g. the bias) – see, for example, [30]. Such methods compute solutions for a sequence of parameter values and restart with a smaller step size if divergence occurs. Clearly this can require a great many linear iteration steps leading to computation times which are much longer than those which may be reasonably required by a successful interactive design system. For one approach to accelerating Gummel's method see [64]. Other iterative schemes are based on (approximations to) Newton's method. These, in general, exhibit faster convergence than Gummel's method near the true solution,

but are more reliant upon an accurate starting value. For examples of these types of scheme we refer the reader to [4], [23], [63]. Iterative techniques for the semiconductor equations in one and two dimensions are discussed and analysed in Chapters 3, 4 and 5.

The efficient solution of these discretised problems has also been an area of active research over the past decade. This has been driven by the desire to solve complicated three–dimensional models on available machines in a reasonable execution time. Many of the ideas arrived at by studying the semiconductor equations have gone into the (scalar) elliptic equation solving package PLTMG ([2]). Moreover an entire suite of routines has been written by a consortium involving the research team at Rutherford Appleton Laboratories. This is called EVEREST and is specifically for the solution of semiconductor problems in up to three dimensions (see [22], [21], [29], [53], [31], [32]). The EVEREST package uses a continuation technique to solve the nonlinear systems arising from the discretised semiconductor equations. An analogous approach is used by PLTMG, which also adopts a hierarchical multigrid approach for the solution of the associated linear systems. For a background to the multigrid method we refer the reader to [33], [3]. Multigrid methods applied to semiconductor equations are given in [58], [16], [57], [36].

As an alternative approach, in Chapter 6 of this thesis we will explore the possibility of using domain decomposition methods to solve the resulting linear systems. The linear systems arising from the continuity equations present the greatest difficulties, since the exponential coefficients (which in a typical application vary at least between $10^{\pm 8}$) cause severe ill-conditioning. With the advent of parallel computers there has been much recent work in the field of domain decomposition. Some of the algorithms date back many years, but it has only been with this advancement in technology that they could be efficiently implemented. Our method will be an "additive Schwarz" type algorithm (see, e.g. [19], [20], [67], [65], [66], [46], [7]). Domain decomposition algorithms can be thought of

preconditioned iterative methods where the preconditioners are constructed from exact or approximate solves for the partial differential equation restricted to subregions or substructures. For a review of these techniques we refer the reader to [72], [18]. Different domain decomposition strategies will perform according not only to their theoretical properties, but also to the type of parallel architecture that they are implemented on. For this reason algorithms which may appear inefficient on one particular type of architecture may perform significantly better on another. In [27] and [44] comparisons are made of domain decomposition techniques for elliptic PDE's and their parallel implementations. Although, to the author's knowledge, there is no definitive text on domain decomposition currently available, both the conference proceedings [24], [12] and their successors are good sources for much of the background material relating to the theoretical and practical aspects of domain decomposition techniques. In Chapters 7 and 8 we discuss, in detail, the implementation of particular domain decomposition strategies on a massively parallel computer.

Finally in this introductory chapter we prepare the reader for what lies ahead.

1.6 What this thesis achieves

We shall firstly introduce discretisation schemes in both one and two dimensions. We then describe some of the existing iterative solution techniques (including Gummel's method) and also exhibit a continuation scheme of our own. Numerical results are shown to demonstrate the robustness of this technique. This thesis is then primarily concerned with the following questions.

- How are the semilinear equations arising in the calculation of the electrostatic potential inside each Gummel iterate to be solved? Because these equations may become singularly perturbed, standard Newton convergence theory predicts a convergence ball whose radius may be so small as to have

no practical meaning. In Chapter 3 we propose a certain quasi-Newton method which computes sequences of upper and lower solutions, and converges quadratically from any starting upper and lower solution pair. The required starting solutions are trivial to find. The convergence is also shown to be mesh-independent.

- How is the convergence of the (outer) Gummel iteration affected by refinement of the finite element mesh? In Chapters 4 and 5 we show that the Lipschitz constant of the fixed point map for our version of Gummel's method is independent of h in one dimension and grows only logarithmically in $1/h$ (as the mesh diameter $h \to 0$) in two dimensions, provided the meshes are refined in a regular manner.

- How do we explain the often surprisingly good performance of Gummel's iteration away from equilibrium? By restricting attention to a particular variant of Gummel's algorithm and a model one–dimensional problem, in Chapter 4 we are able to provide some results which, without being completely rigorous, help to explain this phenomenon. Further arguments allow us to show that the computed potential exhibits sharp layers, interior to the domain of computation, which are known to exist in the solution of the continuous problem ([9], [61]).

- How can the linear systems which arise throughout the implementation of the algorithm in two dimensions be effectively solved in parallel? In Chapter 6 we shall discuss a preconditioned method which has conditioning independent of the jumps of the coefficients of the PDE across subdomain boundaries, and only growing logarithmically as the fine grid is refined relative to the coarse grid. Thus the degradation of performance of the inner iterates is no worse than that of the outer (Gummel) iterates as the mesh is refined. We also consider the effect of this type of domain decomposi-

tion technique on a certain class of model problems. These are often used as benchmark tests for preconditioning strategies. We have found that, in some of these types of problems, it is possible to achieve acceptable convergence without the need to precondition, whereas in others, preconditioning is essential for effective solution times. Theoretical results and numerical experiments are given.

- How should this algorithm be implemented on a massively parallel machine? One range of such machines (e.g. CM2, MasPar MP–1) have a large number of processors (typically some multiple of 1024), each with its own fast memory. These operate in SIMD lockstep, i.e. at any instant all processors are implementing the same instruction. Interchange of data between neighbouring processors is usually quicker than exchanges between random processors in the array. The slowest communication is between the array of processors and the outside world. Such machines naturally lead the user to work with a large number of subdomains each of which has a small number of nodes. In Chapter 7 we describe implementation issues arising from such an architecture, and how we have dealt with them. We develop the implementation ideas with the aid of some model numerical examples. Finally, in Chapter 8, we give some numerical results for the semiconductor system on the MasPar MP-1.

Chapter 2

Discretisation and solution methods

2.1 Introduction

In this chapter we shall outline various methods of discretising the semiconductor equations in both one and two dimensions. In many practical applications the engineer is interested in the current flowing through the device. Hence a discretisation that allows us to readily evaluate the current is considered desirable. We shall discuss schemes which in some sense exhibit current conservation properties. We then proceed to outline some of the existing solution techniques for such schemes. Finally we consider one of our own algorithms which provides a robust way of obtaining a solution even for large applied voltages. This algorithm is based upon the well-known method of continuation.

2.2 Discretisation in one dimension

We shall first consider the one dimensional problem. After employing the quasi-Fermi potentials introduced in Chapter 1, the equations modelling a semicon-

ducting device in steady–state become

$$-\lambda^2\psi'' + \delta\{\exp(\psi - v) - \exp(w - \psi)\} - d = 0, \tag{2.2.1}$$

$$-(\exp(\psi - v)v')' - \sigma\rho_v r(\psi, v, w) = 0, \tag{2.2.2}$$

$$-(\exp(w - \psi)w')' + \sigma\rho_w r(\psi, v, w) = 0, \tag{2.2.3}$$

on the domain $\Lambda = [0, 1]$, subject to the boundary conditions,

$$v(0) = \alpha_0 = w(0), \quad v(1) = \alpha_1 = w(1), \tag{2.2.4}$$

and with ψ chosen to satisfy the *zero space charge condition*

$$\delta\{\exp(\psi - v) - \exp(w - \psi)\} - d = 0 \quad \text{at} \quad x = 0, 1.$$

More explicitly, using (2.2.4),

$$\psi(0) = \sinh^{-1}\left(\frac{d(0)}{2\delta}\right) + \alpha_0, \quad \psi(1) = \sinh^{-1}\left(\frac{d(1)}{2\delta}\right) + \alpha_1. \tag{2.2.5}$$

We set $\beta_0 = \sinh^{-1}(d(0)/2\delta)$ and $\beta_1 = \sinh^{-1}(d(1)/2\delta)$. The parameters $\lambda, \delta, \rho_v, \rho_w$ and σ are those given in Chapter 1. Recall also that d is the (scaled) doping profile and the function r models the recombination and generation of holes and electrons in the device.

We can now see heuristically why the solutions of this system may exhibit layer behaviour. Consider for example the zero-current case ($\alpha_i = 0,\ i = 0, 1$) for a simple $p - n$ diode with equal doping levels. This is modelled by choosing d to have the values ± 1 in the n, p regions respectively. Then, from (2.2.5),

$$\psi = \pm\beta, \qquad \text{where} \quad \beta = \sinh^{-1}(1/(2\delta)) \tag{2.2.6}$$

at contacts in the n, p regions. When the parameter λ in (2.2.1) is small, (2.2.1) is singularly perturbed. Then ψ changes rapidly in a small region (layer) around the interface in d and remains essentially constant (and equal to its contact value) in the rest of the domain. When α_0, α_1 differ from zero, layers also arise in v, w.

Singular perturbation theory has been used to explain these phenomena in detail ([49]).

In one dimension our discretisation schemes will be with respect to the mesh

$$0 = x_0 < x_1 < \ldots \quad < x_{n+1} = 1. \tag{2.2.7}$$

Then, for $i = 1, \ldots, n+1$, we set $\quad h_i = (x_i - x_{i-1})$. We make the assumption that if $h = \max_i h_i$ then there exists a constant γ_1 independent of h such that

$$h_i \geq \gamma_1 h \quad \text{for} \quad i = 1, \ldots, n+1 \tag{2.2.8}$$

The condition (2.2.8) is often referred to as *quasi–uniformity* in the literature. We now discuss various approaches to discretising (2.2.1)– (2.2.3).

Finite difference discretisation. We first consider a finite difference discretisation of (2.2.1)-(2.2.3) with respect to the mesh (2.2.7). This can also be interpreted as a *finite volume* scheme. To motivate the scheme, first observe that (2.2.1)-(2.2.3) can each be written in the form

$$-j' = f \tag{2.2.9}$$

for some j and f. (In the case of (2.2.2), (2.2.3), j is the *current* of *electrons* and *holes* respectively.) Equation (2.2.9) is a simple *conservation law* and clearly

$$-j(1) + j(0) = \int_0^1 f. \tag{2.2.10}$$

If we now introduce the mid points $x_{i-\frac{1}{2}} = (x_i + x_{i-1})/2$ of subintervals, we can integrate (2.2.9) over each "cell" $[x_{i-\frac{1}{2}},\ x_{i+\frac{1}{2}}]$ to obtain the "local" conservation laws

$$-j(x_{i+\frac{1}{2}}) + j(x_{i-\frac{1}{2}}) \ = \ \int_{x_{i-\frac{1}{2}}}^{x_{i+\frac{1}{2}}} f, \quad i = 1, \ldots, n. \tag{2.2.11}$$

Then our numerical methods for (2.2.1)-(2.2.3) are discretisations of (2.2.11) of the general form

$$-J_{i+\frac{1}{2}} + J_{i-\frac{1}{2}} = \overline{h}_i\ f_i\ , \quad i = 1, \ldots, n. \tag{2.2.12}$$

Here $J_{i-\frac{1}{2}}$ is some approximation of $j(x_{i-\frac{1}{2}})$,

$$\overline{h}_i = (h_i + h_{i+1})/2\ , \quad i = 1, \ldots, n, \tag{2.2.13}$$

and $f_i = f(x_i)$, (i.e. we have used a one point quadrature rule rule to approximate the right hand side of (2.2.11)).

The scheme (2.2.12) satisfies a discrete version of (2.2.10), obtained by summing over i :

$$-J_{n+\frac{1}{2}} \ + \ J_{\frac{1}{2}} \ = \ \sum_{i=1}^{n} \overline{h}_i \ f_i.$$

This discretisation of (2.2.1)-(2.2.3) will yield a nonlinear system for the unknown vectors $\boldsymbol{\Psi}, \boldsymbol{V}, \boldsymbol{W} \in I\!R^n$, which approximate the solutions at the *interior nodes*:

$$\Psi_i \cong \psi(x_i);\ V_i \cong v(x_i),\ W_i \cong w(x_i),\ \ i = 1, \ldots, n.$$

Associated with these nodal values are their *piecewise linear interpolants* with boundary values given by (2.2.4) and (2.2.5) which we denote by Ψ, V, W. To help describe the system satisfied by $\boldsymbol{\Psi}$, $\boldsymbol{V}$, $\boldsymbol{W}$ it is convenient to introduce the following MATLAB style notation [59]. Let $\boldsymbol{\Psi} \in I\!R^n$. If $f : I\!R \to I\!R$, define $f(\boldsymbol{\Psi}) \in I\!R^n$ by $(f(\boldsymbol{\Psi}))_i = f(\Psi_i),\ \ i = 1, \ldots, n$. Let diag$\{\boldsymbol{\Psi}\}$ be the $n \times n$ matrix with $\boldsymbol{\Psi}$ on its main diagonal, and zeros elsewhere. If $\mathbf{a}, \mathbf{b}, \mathbf{c}$ are (column) vectors with perhaps different dimensions, then $[\mathbf{a}; \mathbf{b}; \mathbf{c}]$ denotes the column vector obtained by stacking $\mathbf{a}, \mathbf{b}$,and $\mathbf{c}$ one above the other.

It remains to describe how we approximate j. Firstly (2.2.1) is (2.2.9) with $j = \lambda^2\psi'$, and $f = d - \delta\{\exp(\psi - v) - \exp(w - \psi)\}$. We approximate $j(x_{i-\frac{1}{2}})$ by

$$J_{i-\frac{1}{2}} = \lambda^2(\Psi_i - \Psi_{i-1})/h_i. \tag{2.2.14}$$

Also (2.2.2) is (2.2.9) with $j = \exp(\psi - v)v'$ and $f = \sigma\rho_v r(\psi, v, w)$. We approximate $j(x_{i-\frac{1}{2}})$ by

$$J_{i-\frac{1}{2}} = \left(\frac{1}{h_i}\int_{x_{i-1}}^{x_i} \exp(\Psi - V)\right)\left(\frac{V_i - V_{i-1}}{h_i}\right). \tag{2.2.15}$$

(2.2.3) is approximated similarly.

The system obtained by using these approximations to (2.2.1)–(2.2.3) can be neatly expressed once we have made the following definition. If Θ is any piecewise linear function with respect to the mesh (2.2.7), define

$$k_i(\Theta) = \frac{1}{h_i^2} \int_{x_{i-1}}^{x_i} \exp \Theta, \quad i = 1, \ldots, n+1.$$

Then, let $\tilde{K}(\Theta)$ denote the $n \times (n+2)$ matrix $\{\tilde{K}_{ij} : i = 1, \ldots, n,\ j = 0, \ldots, n+1\}$ given for $i = 1, \ldots, n$ by:

$$\begin{aligned} \tilde{K}(\Theta)_{i,i} &= k_i(\Theta) + k_{i+1}(\Theta), \\ \tilde{K}(\Theta)_{i,i-1} &= - k_i(\Theta), \\ \tilde{K}(\Theta)_{i,i+1} &= - k_{i+1}(\Theta), \end{aligned}$$

and, for all other i, j by

$$\tilde{K}(\Theta)_{ij} = 0.$$

Let $K(\Theta)$ be the $n \times n$ symmetric tridiagonal matrix with elements

$$K(\Theta)_{ij} = \tilde{K}(\Theta)_{ij}, \quad i, j = 1, \ldots, n.$$

Then our finite difference scheme takes the form

$$\lambda^2 \tilde{K}(0)[\beta_0 + \alpha_0; \boldsymbol{\Psi}; \beta_1 + \alpha_1] + \mathrm{diag}\{\overline{\boldsymbol{h}}\}[\delta(\exp(\boldsymbol{\Psi} - \boldsymbol{V}) - \exp(\boldsymbol{W} - \boldsymbol{\Psi})) - \boldsymbol{d}] = \mathbf{0}, \tag{2.2.16}$$

$$\tilde{K}(\Psi - V)[\alpha_0; \boldsymbol{V}; \alpha_1] - \sigma \rho_v \mathrm{diag}\{\overline{\boldsymbol{h}}\} r(\boldsymbol{\Psi}, \boldsymbol{V}, \boldsymbol{W}) = \mathbf{0}, \tag{2.2.17}$$

$$\tilde{K}(W - \Psi)[\alpha_0; \boldsymbol{W}; \alpha_1] + \sigma \rho_w \mathrm{diag}\{\overline{\boldsymbol{h}}\} r(\boldsymbol{\Psi}, \boldsymbol{V}, \boldsymbol{W}) = \mathbf{0}. \tag{2.2.18}$$

We now introduce a standard finite element scheme for (2.2.1)–(2.2.3), and show that a certain approximation of it also leads to (2.2.16)–(2.2.18).

Standard finite element discretisation. Recall that our problem, (2.2.1)–(2.2.3), is posed over the domain $\Lambda = [0, 1]$. We begin by introducing the appropriate spaces, $L_2(\Lambda)$ with inner product $(\cdot, \cdot)$ and norm $\| \cdot \|_{L_2(\Lambda)}$ and $H^1(\Lambda)$ with

seminorm $|\cdot|_{H^1(\Lambda)}$ and norm $\|\cdot\|_{H^1(\Lambda)}$. If $f, g \in (L_2(\Lambda))^2$ then

$$(f,g) = \int_0^1 fg.$$

Let $S_h(\Lambda)$ be the space of piecewise linear functions on Λ relative to the mesh (2.2.7). Similarly let $\Sigma_h(\Lambda)$ be the space of piecewise constant functions on Λ relative to the mesh (2.2.7). We define the usual hat function basis

$$\{\phi_i,\ i = 0, \ldots, n+1\} \quad \text{where} \quad \phi_i(x_j) = \delta_{ij}.$$

Throughout we identify a vector $\boldsymbol{x} \in I\!R^{n+2}$ with the function

$$X = \sum_{i=0}^{n+1} x_i \phi_i \in S_h(\Lambda).$$

We then set

$$S_h^0(\Lambda) = \{X \in S_h(\Lambda) : X(0) = X(1) = 0\}.$$

The finite element method for (2.2.1)-(2.2.3) would then consist of seeking $\Psi, V, W \in S_h(\Lambda)$ satisfying the boundary conditions (2.2.4), (2.2.5) and such that the following three equations

$$\lambda^2(\Psi', \phi_p') + (\delta\{\exp(\Psi - V) - \exp(W - \Psi)\} - d, \phi_p) = 0, \tag{2.2.19}$$

$$(\exp(\Psi - V)V', \phi_p') - (\sigma\rho_v r(\Psi, V, W), \phi_p) = 0, \tag{2.2.20}$$

$$(\exp(W - \Psi)W', \phi_p') + (\sigma\rho_w r(\Psi, V, W), \phi_p) = 0, \tag{2.2.21}$$

are satisfied for $1 \leq p \leq n$. This again yields a nonlinear system for the triple of unknowns (Ψ, V, W). Clearly the first term in (2.2.19) gives rise to the matrix–vector product in (2.2.16). Similarly the first terms in (2.2.20), (2.2.21) also correspond to the matrix–vector products in (2.2.17), (2.2.18) respectively. However the standard finite element discretisation of the zeroth order terms in (2.2.1)-(2.2.3) produces different approximations than those obtained by our finite difference scheme. For instance, if we consider the vector obtained from the second term in (2.2.19), then the ith element will depend on the $(i-1)$th, ith and $(i+1)$th

elements of $\boldsymbol{\Psi}$, $\boldsymbol{V}$ and $\boldsymbol{W}$. This differs from our finite difference discretisation where our one point quadrature rule means that the ith element depends only on the ith elements of $\boldsymbol{\Psi}$, $\boldsymbol{V}$ and $\boldsymbol{W}$. Hence the standard finite element discretisation of (2.2.1)–(2.2.3) has the general form (2.2.12), but f_i will differ from that of the finite difference method.

However we will now proceed to describe an approximate finite element scheme where our treatment of the zeroth order terms will again lead to a diagonal nonlinearity.

Approximate finite element discretisation We begin by introducing the trapezoidal rule:

$$\int_0^1 f \;\cong\; \frac{h_1}{2} f(x_0) + \sum_{i=1}^{n} \overline{h}_i f(x_i) + \frac{h_{n+1}}{2} f(x_{n+1}) \tag{2.2.22}$$

with $\overline{h}_i$ as in (2.2.13). This rule is exact for any $f \in S_h(\Lambda)$ and it induces the discrete bilinear form

$$\langle f, g \rangle = \frac{h_1}{2}(fg)(x_0) + \sum_{i=1}^{n} \overline{h}_i (fg)(x_i) + \frac{h_{n+1}}{2}(fg)(x_{n+1})$$

If we use this to approximate the zeroth order terms in (2.2.19)-(2.2.21), we obtain

$$\lambda^2(\Psi', \phi_p') + \langle \delta\{\exp(\Psi - V) - \exp(W - \Psi)\} - d, \phi_p \rangle = 0, \tag{2.2.23}$$

$$(\exp(\Psi - V)V', \phi_p') - \langle \sigma \rho_v r(\Psi, V, W), \phi_p \rangle = 0, \tag{2.2.24}$$

$$(\exp(W - \Psi)W', \phi_p') + \langle \sigma \rho_w r(\Psi, V, W), \phi_p \rangle = 0, \tag{2.2.25}$$

for all $1 \leq p \leq n$. Note that (2.2.23)–(2.2.25) yields an system identical to (2.2.16)–(2.2.18).

This approximation is useful in practice, since the zero-order terms are complicated nonlinear functions and the mass lumping produces the simplest possible diagonal approximation to them. It is also useful in theory since iterative schemes

for solving (2.2.23) (with V and W fixed) have useful monotonicity properties which are not present in (2.2.19).

Furthermore we shall also introduce a special piecewise–constant average for the exponential coefficients in the second order terms in (2.2.24) and (2.2.25). For any $Y \in S_h(\Lambda)$ we define $\overline{Y} \in \Sigma_h(\Lambda)$ by

$$\exp(\overline{Y}|_{I_i}) = \left(\frac{1}{h_i}\int_{x_{i-1}}^{x_i} \exp(-Y(x))dx\right)^{-1}, \quad \text{for each interval } I_i. \tag{2.2.26}$$

Then $\exp(\overline{Y})$ is the *harmonic average* of $\exp(Y)$ and can be calculated analytically. This yields an approximate finite element method which seeks $(\Psi, V, W) \in S_h(\Lambda)^3$ satisfying (2.2.4), (2.2.5) and

$$\lambda^2(\Psi', \phi_p') + \langle \delta\{\exp(\Psi - V) - \exp(W - \Psi)\} - d, \phi_p\rangle = 0, \tag{2.2.27}$$

$$(\exp(\overline{\Psi - V})V', \phi_p') - \langle \sigma\rho_v r(\Psi, V, W), \phi_p\rangle = 0, \tag{2.2.28}$$

$$(\exp(\overline{W - \Psi})W', \phi_p') + \langle \sigma\rho_w r(\Psi, V, W), \phi_p\rangle = 0, \tag{2.2.29}$$

for all $1 \leq p \leq n$. This scheme also has certain weak current–conservation properties, which we discuss in Section 2.3.1. Furthermore, if the zeroth–order terms in (2.2.28) and (2.2.29) are assumed to be zero, then the use of this harmonic average yields solutions to (2.2.28), (2.2.29) which are exact at the nodes. (This is known as the Allen–Southwell exponential fitting scheme in fluid dynamics literature and as the Scharfetter–Gummel discretisation in the semiconductor literature, although in these cases the exactness property is more usually presented for the equivalent convection–diffusion equations.) We now prove this result.

LEMMA 2.2.1 *Suppose $a : [0,1] \longmapsto \mathbb{R}$ is a given function with*

$$0 < m \leq \exp(a(x)) \leq M < \infty, \quad x \in [0,1].$$

Let

$$-(\exp(a)u')' = 0 \quad x \in (0,1) \tag{2.2.30}$$

with $u(0) = u_0,\ u(1) = u_1$. *Discretise (2.2.30) by the finite element method with respect to the mesh (2.2.7), using the harmonic average approximation for* $\exp(a)$ *given by (2.2.26). This results in the system*

$$(\exp(\overline{a})U', \phi_i') = 0, \quad i = 1, \ldots, n \tag{2.2.31}$$

with $U(x_0) = u_0,\ U(x_{n+1}) = u_1$. *The solution,* U, *to this system is exact at the nodes, i.e.*

$$U(x_i) = u(x_i),\ i = 1, \ldots, n.$$

Proof From (2.2.30) we have

$$(\exp(a)u') = c = \text{constant}$$

and hence

$$u' = c\exp(-a).$$

By integrating both sides of this equation over I_i we obtain

$$u(x_i) - u(x_{i-1}) = c\int_{x_{i-1}}^{x_i} \exp(-a(x))dx.$$

Then summing over i and utilising the boundary conditions satisfied by u gives

$$u_1 - u_0 = c\sum_{i=1}^{n+1}\int_{x_{i-1}}^{x_i} \exp(-a(x))dx = c\int_0^1 \exp(-a(x))dx. \tag{2.2.32}$$

In addition, (2.2.31) implies that for $i = 1, \ldots, n$

$$\int_{x_{i-1}}^{x_i} \exp(\overline{a}|_{I_i}) \left(\frac{U(x_i) - U(x_{i-1})}{h_i}\right) \frac{1}{h_i} - \int_{x_i}^{x_{i+1}} \exp(\overline{a}|_{I_{i+1}}) \left(\frac{U(x_{i+1}) - U(x_i)}{h_{i+1}}\right) \frac{1}{h_{i+1}} = 0.$$

Therefore, for $i = 1, \ldots, n$

$$\exp(\overline{a}|_{I_i}) \left(\frac{U(x_i) - U(x_{i-1})}{h_i}\right) - \exp(\overline{a}|_{I_{i+1}}) \left(\frac{U(x_{i+1}) - U(x_i)}{h_{i+1}}\right) = 0.$$

From this it follows that there is a constant k such that for $i = 1, \ldots, n+1$,

$$\exp(\overline{a}|_{I_i}) \left(\frac{U(x_i) - U(x_{i-1})}{h_i}\right) = k.$$

Hence

$$\frac{U(x_i) - U(x_{i-1})}{h_i} = k(\exp(\overline{a}|_{I_i}))^{-1} = k\left(\frac{1}{h_i}\int_{x_{i-1}}^{x_i} \exp(-a(x))dx\right),$$

and so

$$U(x_i) - U(x_{i-1}) = k\int_{x_{i-1}}^{x_i} \exp(-a(x))dx, \quad i = 1, \ldots, n+1.$$

After summing over i and using the boundary conditions we obtain

$$u_1 - u_0 = U(x_{n+1}) - U(x_0) = k\int_0^1 \exp(-a(x))dx \tag{2.2.33}$$

Comparing (2.2.32) and (2.2.33) it is obvious that $c = k$ and hence $u(x_i) = U(x_i)$ for all i. ■

We now introduce our discretisation of the governing equations in two dimensions. We shall consider only the standard finite element method and its approximation using quadrature on the zeroth order terms and harmonic averaging for the exponential coefficients in the second order terms.

2.3 Finite element discretisation in two dimensions

After the quasi–Fermi change of variables, the equations modelling a two–dimensional semiconducting device can be expressed as

$$-\lambda^2\Delta\psi + \delta\{\exp(\psi - v) - \exp(w - \psi)\} - d = 0, \tag{2.3.34}$$

$$-\nabla.(\exp(\psi - v)\nabla v) - \sigma\rho_v r(\psi, v, w) = 0, \tag{2.3.35}$$

$$-\nabla.(\exp(w - \psi)\nabla w) + \sigma\rho_w r(\psi, v, w) = 0. \tag{2.3.36}$$

We consider this system on a convex polygonal domain $\Omega \in \mathbb{R}^2$ with boundary $\partial\Omega$, subject to mixed boundary conditions given as follows. At the "Ohmic contacts" $\partial\Omega_D = \cup_i \partial\Omega_{D_i}$ (where the $\partial\Omega_{D_i}$ are a finite number of closed nonempty

subsets of $\partial\Omega$), we impose piecewise constant Dirichlet boundary conditions on v and w:

$$v|_{\partial\Omega_{D_i}} = w|_{\partial\Omega_{D_i}} = \alpha_i = \text{ constant}, \quad \text{for each } i. \tag{2.3.37}$$

This corresponds to the imposition of an applied voltage $\alpha_i U_T$ at each of the contacts $\partial\Omega_{D_i}$. Dirichlet conditions for ψ at the contacts are then obtained by requiring that the space charge there should be zero. This is expressed as

$$\delta\{\exp(\psi - v) - \exp(w - \psi)\} - d = 0, \quad \text{on } \partial\Omega_D, \tag{2.3.38}$$

or, equivalently,

$$\psi|_{\partial\Omega_{D_i}} = \alpha_i + \sinh^{-1}\left(\frac{d|_{\partial\Omega_{D_i}}}{2\delta}\right) \quad \text{for each } i. \tag{2.3.39}$$

On $\partial\Omega_N := \partial\Omega \backslash \partial\Omega_D$ the device is assumed to be insulated. That is we have the homogeneous Neumann conditions:

$$\frac{\partial\psi}{\partial n} = \frac{\partial u}{\partial n} = \frac{\partial v}{\partial n} = 0 \quad \text{on } \partial\Omega_N. \tag{2.3.40}$$

We shall triangulate Ω and discretise (2.3.34) - (2.3.36) using linear finite elements. To facilitate the domain decomposition methods in Chapter 6, our triangulation is obtained by first dividing Ω into convex quadrilateral subdomains or "substructures" $\Omega^{(i)}$ such that $\overline{\Omega} = \cup_i \overline{\Omega^{(i)}}$ and $\Omega^{(i)} \cap \Omega^{(j)} = \emptyset$ when $i \neq j$. Each substructure vertex is assumed to belong to no more than four substructures, and the intersection of the closure of any two substructures is either empty or consists entirely of a common edge and the two associated vertices. We divide each substructure into two triangles to obtain a "coarse grid" with maximum mesh diameter H. We then further refine the coarse grid to form a "fine grid" with maximum mesh diameter h. For theoretical purposes we assume throughout that both coarse grid and fine grid are conforming, and that as the mesh is refined both grids are *regular* and satisfy an *inverse assumption* (in the sense of [13, p.124, p.140]). Again, such meshes are often referred to as quasi–uniform. We also assume that the fine grid is of *weakly acute type*, (i.e. each angle of its triangles

is no greater then $\pi/2$). Finally we assume that the points in $\partial\Omega_D \cap \overline{\partial\Omega_N}$ (i.e. the *collision points* between Dirichlet and Neumann boundary conditions) are vertices of substructures and hence are nodes of both the coarse and fine grids.

We introduce the usual spaces $L_2(\Omega)$ with inner product $(\cdot,\cdot)$ and norm $\|\cdot\|_{L_2(\Omega)}$, and $H^1(\Omega)$ with semi-norm $|\cdot|_{H^1(\Omega)}$ and norm $\|\cdot\|_{H^1(\Omega)}$. If $\boldsymbol{f},\boldsymbol{g} \in (L_2(\Omega))^2$ then $(\boldsymbol{f},\boldsymbol{g}) := \int_\Omega \boldsymbol{f}\cdot\boldsymbol{g}$, with $\cdot$ denoting the usual dot product on $\mathbb{R}^2$. Let $S_h(\Omega)$ denote the space of all piecewise linear functions on Ω subordinate to the fine grid. The usual basis functions for $S_h(\Omega)$ are $\{\phi_p\}$ where ϕ_p is 1 at the pth node and zero elsewhere. Let Σ_h denote the space of all *piecewise constant* functions subordinate to the fine grid. The usual finite element method for (2.3.34) – (2.3.36) would consist of seeking $\Psi, V, W \in S_h$ satisfying (2.3.37), (2.3.39) and such that the equations

$$\lambda^2(\nabla\Psi,\ \nabla\phi_p) + (\delta\{\exp(\Psi - V) - \exp(W - \Psi)\} - d,\ \phi_p) = 0, \qquad (2.3.41)$$

$$(\exp(\Psi - V)\nabla V,\ \nabla\phi_p) - (\sigma\rho_v r(\Psi, V, W), \phi_p) = 0, \qquad (2.3.42)$$

$$(\exp(W - \Psi)\nabla W, \nabla\phi_p) + (\sigma\rho_w r(\Psi, V, W), \phi_p) = 0, \qquad (2.3.43)$$

are satisfied for all nodes $p \notin \partial\Omega_D$. (Here $(\cdot,\cdot)$ denotes the usual L_2 inner product of scalar or vector-valued functions on Ω.)

We shall again consider a slightly modified scheme analogous to (2.2.27)–(2.2.29). Firstly we shall use "mass lumping" for the zero-order nonlinear terms. As mentioned earlier, the diagonal nonlinearity obtained by doing this will be exploited in the monotone schemes introduced in Chapter 3.

The mass lumping can be achieved by approximating the usual finite element method using the nodal quadrature rule:

$$\int_\Omega f \ \approx\ \sum_T \frac{1}{3}\mathcal{A}(T)\sum_{p_T} f(p_T). \qquad (2.3.44)$$

The outer sum is over all triangles T of the fine grid, the inner sum is over the three nodes p_T of T and $\mathcal{A}(T)$ denotes the area of T. This rule is exact when

$f \in S_h(\Omega)$ and it is well known that its use in approximating the standard linear finite element method for elliptic problems yields no degradation of accuracy in the energy norm [13, Theorem 4.1.6]. The rule (2.3.44) may be equivalently written

$$\int_\Omega f \approx \sum_p w_p f(p), \tag{2.3.45}$$

where the sum is over all nodes p of the fine grid and each weight w_p is simply one third the sum of the areas of all the triangles which meet at node p. This induces the discrete bilinear form

$$\langle f, g \rangle = \sum_T \frac{1}{3} \mathcal{A}(T) \sum_{p_T} (f\, g)(p_T) = \sum_p w_p f(p) g(p), \tag{2.3.46}$$

which is easily seen to be an inner product on $S_h(\Omega)$. We use this to approximate the second terms in (2.3.41)–(2.3.43).

Then our finite element method for (2.3.34) – (2.3.36) consists of seeking $\Psi, V, W \in S_h(\Omega)$ satisfying (2.3.37), (2.3.39) and such that

$$\lambda^2(\nabla\Psi,\ \nabla\phi_p) + \langle \delta(\exp(\Psi - V) - \exp(W - \Psi)) - d,\ \phi_p \rangle = 0, \tag{2.3.47}$$

$$(\exp(\Psi - V)\nabla V,\ \nabla\phi_p) - \langle \sigma\rho_v r(\Psi, V, W), \phi_p \rangle = 0, \tag{2.3.48}$$

$$(\exp(W - \Psi)\nabla W, \nabla\phi_p) + \langle \sigma\rho_w r(\Psi, V, W), \phi_p \rangle = 0, \tag{2.3.49}$$

are satisfied for all nodes $p \notin \partial\Omega_D$. Since ϕ_p is zero except the pth node, the zeroth order terms in (2.3.47)-(2.3.49) involve only the nodal values of Ψ_p, V_p, W_p, i.e. they are "diagonal nonlinearities".

Our second modification to (2.3.41)–(2.3.43) is to introduce an averaging technique for the coefficients in the continuity equations. This is analogous to the approximation we make in one dimension. The averaging technique was originally published by Brezzi, [11] (see also [10]), for the Slotboom variables. This can be viewed as a certain hybrid finite element scheme which exhibits weak current conservation properties.

For any $Y \in S_h(\Omega)$, we define $\overline{Y} \in \Sigma_h(\Omega)$ by

$$\exp(\overline{Y}|_T) = \left\{ \frac{1}{\mathcal{A}(T)} \int_T \exp(-Y) \right\}^{-1}, \quad \text{for each triangle } T. \tag{2.3.50}$$

Clearly (2.3.50) is the two–dimensional analogue of (2.2.26). Hence (2.3.47)–(2.3.49) is modified to the following problem. We seek $\Psi, V, W \in S_h(\Omega)$ satisfying (2.3.37), (2.3.39) and such that

$$\lambda^2(\nabla\Psi,\ \nabla\phi_p) + \langle \delta(\exp(\Psi - V) - \exp(W - \Psi)) - d,\ \phi_p \rangle = 0, \tag{2.3.51}$$

$$(\exp(\overline{\Psi - V})\nabla V,\ \nabla\phi_p) - \langle \sigma\rho_v r(\Psi, V, W), \phi_p \rangle = 0, \tag{2.3.52}$$

$$(\exp(\overline{W - \Psi})\nabla W, \nabla\phi_p) + \langle \sigma\rho_w r(\Psi, V, W), \phi_p \rangle = 0, \tag{2.3.53}$$

are satisfied for all nodes $p \notin \partial\Omega_D$.

Finally in this section we study our proposed two–dimensional scheme, (2.3.51)–(2.3.53), in the context of a linear model problem.

2.3.1 Hybrid finite element methods and the harmonic average

This subsection is an aside looking at some of the properties of the harmonic averaging technique and how its use can be viewed as a hybrid finite element method. We introduce these ideas with the aid of the linear model problem

$$-\nabla.(a\nabla u) = f \tag{2.3.54}$$

in a convex polygon Ω with mixed boundary conditions

$$u = g \quad \text{on} \quad \partial\Omega_D, \quad \partial\Omega_D \neq \phi, \tag{2.3.55}$$

$$\frac{\partial u}{\partial n} = 0 \quad \text{on} \quad \partial\Omega_N. \tag{2.3.56}$$

We make the following assumptions.

A1. Ω is triangulated by a regular triangulation, such that at least one node lies on $\partial\Omega_D$. The resulting space of piecewise linear functions is denoted $S_h(\Omega)$. Similarly the subspace of $S_h(\Omega)$ with zero boundary values on $\partial\Omega_D$ is denoted $S_h^0(\Omega)$.

A2. $f \in L_2(\Omega)$

A3. $g = g|_{\partial\Omega_D}$ where $g \in H^1(\Omega)$, (i.e. g is the trace of an $H^1(\Omega)$ function restricted to $\partial\Omega_D$).

A4. $0 < a_{min} \leq a(\boldsymbol{x}) \leq a_{max} < \infty, \boldsymbol{x} \in \overline{\Omega}$

Suppose we are interested in approximations to the flux $a\nabla u$ which somehow model the conservation law (2.3.54). One strategy is to introduce some (as yet undetermined) piecewise constant functions $a_h, f_h \in \Sigma_h$ and to consider the (approximate) finite element method which computes $u_h \in S_h$ satisfying $u_h \doteq g$ on $\partial\Omega_D$ and

$$\int_\Omega a_h \nabla u_h . \nabla \phi = \int_\Omega f_h \phi, \quad \phi \in S_h, \quad \phi = 0 \text{ on } \partial\Omega_D. \tag{2.3.57}$$

Consider the piecewise constant function

$$\boldsymbol{J}_h := a_h \nabla u_h \in (\Sigma_h)^2 \tag{2.3.58}$$

which in some sense approximates the flux $(a\nabla u)$. It has certain weak conservation properties with respect to the (discontinuous piecewise linear) particular integral of (2.3.54) given on each triangle T by

$$\boldsymbol{J}_f(\boldsymbol{x}) = -\frac{1}{2}\left\{\boldsymbol{x} - \mathcal{A}(T)^{-1}\int_T \boldsymbol{y}\,\mathrm{d}\,y\right\}\{f_h|_T\}, \quad \boldsymbol{x} \in T. \tag{2.3.59}$$

Clearly then for each T we have

$$-\nabla . \boldsymbol{J}_f = f_h \quad \text{on } T, \tag{2.3.60}$$

and

$$\int_T \boldsymbol{J}_f = \boldsymbol{0}. \tag{2.3.61}$$

Hence it follows that for any $\phi \in S_h$ with $\phi = 0$ on $\partial\Omega_D$

$$\sum_T \int_T (\boldsymbol{J}_h + \boldsymbol{J}_f).\nabla\phi = \sum_T \int_T \boldsymbol{J}_h.\nabla\phi = \sum_T \int_T f_h\phi. \tag{2.3.62}$$

Now using divergence theorem and (2.3.60) we obtain

$$\sum_T \int_{\partial T} (\boldsymbol{J}_h + \boldsymbol{J}_f).\boldsymbol{n}\phi = 0, \tag{2.3.63}$$

where for each T, $\boldsymbol{n}$ is the unit outward normal to its boundary ∂T. Taking $\phi = \phi_p$ in (2.3.63) shows the *local conservation property* that the averages of the flux $\boldsymbol{J}_h + \boldsymbol{J}_f$ along all edges meeting at any node p is zero.

Furthermore, there is another completely different way of interpreting $\boldsymbol{J}_h$. Observe that the divergence theorem and (2.3.63) imply

$$\int_\Omega \boldsymbol{J}_h.\nabla\phi = -\sum_T \int_{\partial T} \{\boldsymbol{J}_f.\boldsymbol{n}\}\phi, \quad \phi \in S_h, \quad \phi = 0 \text{ on } \partial\Omega_D. \tag{2.3.64}$$

Moreover (2.3.58) and (2.3.61) conspire to give

$$\int_\Omega a_h^{-1}\{\boldsymbol{J}_h + \boldsymbol{J}_f\}.\boldsymbol{\tau} - \int_\Omega \nabla u_h.\boldsymbol{\tau} = 0, \quad \boldsymbol{\tau} \in (\Sigma_h)^2. \tag{2.3.65}$$

Equations (2.3.64), (2.3.65) constitute a *hybrid* discretisation of the mixed reformulation of (2.3.54) which seeks a pair $(\boldsymbol{J}, u)$ such that $u = g$ on $\partial\Omega_D$, $(\boldsymbol{J} + \boldsymbol{J}_f).\boldsymbol{n} = 0$ on $\partial\Omega_N$, $(\boldsymbol{J} + \boldsymbol{J}_f)$ is continuous on Ω and

$$-\nabla.(\boldsymbol{J} + \boldsymbol{J}_f) = f, \tag{2.3.66}$$

$$a^{-1}\boldsymbol{J} + a^{-1}\boldsymbol{J}_f - \nabla u = \boldsymbol{0}. \tag{2.3.67}$$

Then writing this in weak form, replacing a, f by a_h, f_h and seeking a solution in $(\Sigma_h)^2 \times S_h$ yields (2.3.64), (2.3.65). This hybrid method is different from the usual mixed method, where more continuity is imposed on the space where $\boldsymbol{J}$ is sought and less on the space where u is sought. This method is proposed for a specific semiconductor equation (slightly different from our applications in this thesis) in [11].

Comparing (2.3.67) with (2.3.65) suggests that we should choose $a_h \in \Sigma_h$ in (2.3.65) so that a_h^{-1} is a good approximation to a^{-1}. The obvious way to do this is to set

$$a_h = \overline{a} := \left\{ \frac{1}{\mathcal{A}(T)} \int_T a^{-1} \right\}^{-1}, \quad \text{on each } T. \tag{2.3.68}$$

This is a more general form of (2.3.50), and appears in the hybrid methods of [11] (see also [10]). However, there a is only replaced by a_h in the first term of (2.3.67). Their method then corresponds to (2.3.57) but with a modified right-hand side.

Besides the connection with mixed methods, there are a number of other justifications for employing the harmonic average approximation (2.3.68). Firstly, as we saw in Lemma 2.2.1, if the one-dimensional analogue of (2.3.54) with $f = 0$ is solved by the finite element method, then the use of the harmonic average yields a scheme which is *exact* at the nodes. This one–dimensional scheme is used in [16].

Secondly, for two-dimensional problems of the form (2.3.54), harmonic averaging of coefficients (along element sides) leads to a piecewise constant approximation of the flux with accuracy which depends only on the smoothness of the flux and of the forcing term f [50]. Such methods are appropriate when the flux is smoother than a or u, as is (empirically) the case in semiconductor modelling. Related observations are made in the earlier work [1], where the use of the harmonic averaged coefficient (in one dimension) is shown to be equivalent to a "generalised finite element method" which is found to be more robust to jumps in a than the standard method.

Hence we propose to discretise (2.3.54) by (2.3.57) with a_h given by (2.3.68). In our application (equations (2.3.35),(2.3.36)) a_h can be easily computed. For f_h in (2.3.57) the natural thing would be to use a standard average. However, this is not feasible in our application because of the complicated form of the zeroth order terms . Thus, in our theoretical studies, we approximate the average using

the quadrature rule (2.2.12):

$$f_h = \frac{1}{3}\sum_{p_T} f(p_T) \quad \text{on each } T. \tag{2.3.69}$$

It is a moderately easy exercise in finite element error analysis to show that method (2.3.57) yields a solution u_h which (as in the usual finite element method) satisfies a quasi–optimal H^1–error estimate under the smoothness assumptions (**A2**)–(**A4**) on a , f and g. For completeness we now include the details of this exercise. The standard weak form of (2.3.54)-(2.3.56) is to seek $u \in H^1(\Omega)$ with $u = g$ on $\partial\Omega_D$ such that

$$(a\nabla u, \nabla\phi) = (f, \phi), \quad \phi \in H^1(\Omega), \quad \phi = 0 \ \text{ on } \ \partial\Omega_D. \tag{2.3.70}$$

The *standard finite element method* is to find $U \in S_h(\Omega)$, $U = g$ on $\partial\Omega_D$ such that

$$(a\nabla U, \nabla\phi) = (f, \phi), \quad \phi \in H^1(\Omega), \quad \phi = 0 \ \text{ on } \ \partial\Omega_D. \tag{2.3.71}$$

The *harmonic average finite element method* is to seek $U \in S_h(\Omega)$, $U = g$ on $\partial\Omega_D$ such that

$$(\overline{a}\nabla U, \nabla\phi) = (f, \phi), \quad \phi \in H^1(\Omega), \quad \phi = 0 \ \text{ on } \ \partial\Omega_D, \tag{2.3.72}$$

with $\overline{a}$ given by (2.3.68). Due to assumption (**A4**) we have

$$0 < a_{max}^{-1} \leq a(x)^{-1} \leq a_{min}^{-1}, \quad x \in \overline{\Omega},$$

and so

$$a_{min} \leq \overline{a}(x) \leq a_{max}, \quad x \in \Omega. \tag{2.3.73}$$

We first need to show that (2.3.71) and (2.3.72) have unique solutions and then bound the norm of these solutions. Throughout the following two lemmas, C will denote a generic constant, independent of h.

LEMMA 2.3.1 *Under the assumptions (**A1**)–(**A4**), the weak formulation (2.3.70) and the finite element problems (2.3.71) and (2.3.72) have unique solutions $u \in H^1(\Omega)$, $U^1 \in S_h(\Omega)$ and $U^2 \in S_h(\Omega)$ respectively. These satisfy the*

energy estimate

$$\max\{\|u\|_{H^1(\Omega)}, \|U^1\|_{H^1(\Omega)}, \|U^2\|_{H^1(\Omega)}\} \leq a_{min}^{-1}(\|f\|_{L_2(\Omega)} + C(a_{min} + a_{max})|g|_{H^1(\Omega)}).$$

Proof Note that u solves (2.3.70) if and only if $u = u_o + g$ with $u_0 \in H^1(\Omega)$, $u_0 = 0$ on $\partial\Omega_D$ and

$$(a\nabla u_0, \nabla\phi) = (f, \phi) - (a\nabla g, \nabla\phi) \quad \text{for all } \phi \in H^1(\Omega),\ \phi = 0 \text{ on } \partial\Omega_D. \quad (2.3.74)$$

Now introduce the space

$$H_0^1 = \{u \in H^1(\Omega) : u = 0 \text{ on } \partial\Omega_D\},$$

and equip H_0^1 with the norm $\|.\|_{H^1(\Omega)}$. Then (2.3.74) may be written : Find $u_0 \in H_0^1$ such that

$$b(u_0, \phi) = L(\phi), \quad \text{for all } \phi \in H_0^1, \quad (2.3.75)$$

where

$$b(u, \phi) = (a\nabla u, \nabla\phi), \quad (2.3.76)$$

$$L(\phi) = (f, \phi) - b(g, \phi). \quad (2.3.77)$$

Then using our assumption (**A4**), $b(.,.)$ is a symmetric, continuous, elliptic bilinear form on H_0^1 with

$$|b(u, \phi)| \leq a_{max}\|u\|_{H^1(\Omega)}\|\phi\|_{H^1(\Omega)},$$

and

$$b(u, u) \geq a_{min}\|u\|_{H^1(\Omega)}^2.$$

Moreover L is continuous with

$$|L(\phi)| \leq (\|f\|_{L_2(\Omega)} + a_{max}|g|_{H^1(\Omega)})\|\phi\|_{H^1(\Omega)}.$$

Then by [39, §(2.1)], the problem (2.3.75) has a unique solution u_0 with

$$\|u_0\|_{H^1(\Omega)} \leq a_{min}^{-1}(\|f\|_{L_2(\Omega)} + a_{max}|g|_{H^1(\Omega)}).$$

Hence (2.3.70) has a unique solution u with

$$\begin{aligned}\|u\|_{H^1(\Omega)} &\leq \|u_0\|_{H^1(\Omega)} + \|g\|_{H^1(\Omega)} \\ &\leq a_{min}^{-1}(\|f\|_{L_2(\Omega)} + a_{max}|g|_{H^1(\Omega)}) + \|g\|_{H^1(\Omega)} \\ &\leq a_{min}^{-1}(\|f\|_{L_2(\Omega)} + C(a_{max} + a_{min})|g|_{H^1(\Omega)}),\end{aligned}$$

where the second inequality comes via Poincarés inequality (see, for instance, [34, page 114]). Hence we have the result for the solution of (2.3.70).

We now consider only the problem (2.3.71). The proof for the problem (2.3.72) is analogous once remark (2.3.73) has been taken into account. Let G denote the interpolant of g as in [69, Theorem 3]. Then

$$|g - G|_{H^1(\Omega)} \leq C|g|_{H^1(\Omega)},$$

so that

$$|G|_{H^1(\Omega)} \leq C|g|_{H^1(\Omega)}.$$

Hence U^1 solves (2.3.71) if and only if $U^1 = U_0^1 + G$ with $U_0^1 \in S_h(\Omega)$, $U_0^1 = 0$ on $\partial\Omega_D$ and

$$(a\nabla U_0^1, \nabla\phi) = (f, \phi) - (a\nabla G, \phi). \tag{2.3.78}$$

By applying the theory of [39] in $S_h^0(\Omega) \subset H_0^1$ this problem has a unique solution U_0^1 with

$$\|U_0^1\|_{H^1(\Omega)} \leq a_{min}^{-1}(\|f\|_{L_2(\Omega)} + a_{max}|G|_{H^1(\Omega)}).$$

Hence (2.3.71) has a unique solution $U^1 \in S_h(\Omega)$ with

$$\begin{aligned}\|U^1\|_{H^1(\Omega)} &\leq a_{min}^{-1}(\|f\|_{L_2(\Omega)} + (a_{max} + a_{min})|G|_{H^1(\Omega)}) \\ &\leq a_{min}^{-1}(\|f\|_{L_2(\Omega)} + C(a_{max} + a_{min})|g|_{H^1(\Omega)}).\end{aligned}$$

■

LEMMA 2.3.2 *Under the assumptions* **(A1)**-**(A4)**, *the solution* U^1 *of (2.3.71) satisfies the quasi–optimal error estimate*

$$|u - U^1|_{H^1(\Omega)} \leq \left(\frac{a_{max}}{a_{min}}\right) \inf_{\substack{\Phi \in S_h(\Omega) \\ \Phi = g \text{ on } \partial\Omega_D}} |u - \Phi|_{H^1(\Omega)} \tag{2.3.79}$$

where u *is the weak solution of (2.3.70). If, in addition,*

$$\sup_T \left\{ \|\frac{\partial a}{\partial x_1}\|_{L_\infty(T)}, \|\frac{\partial a}{\partial x_2}\|_{L_\infty(T)} \right\} \leq M \tag{2.3.80}$$

then the solution U^2 *of (2.3.72) satisfies*

$$|u - U^2|_{H^1(\Omega)} \leq$$
$$CM\frac{a_{max}^2}{a_{min}^4}(1 + a_{min} + a_{max})\left(1 + \frac{a_{max}}{a_{min}}\right)h + \left(\frac{a_{max}}{a_{min}}\right) \inf_{\substack{\Phi \in S_h(\Omega) \\ \Phi = g \text{ on } \partial\Omega_D}} |u - \Phi|_{H^1(\Omega)}. \tag{2.3.81}$$

Proof By the previous lemma and (2.3.73), both (2.3.71) and (2.3.72) have unique solutions U^1, U^2 respectively with

$$\max\left\{\|U^1\|_{H^1(\Omega)}, \|U^2\|_{H^1(\Omega)}\right\} \leq a_{min}^{-1}(\|f\|_{L_2(\Omega)} + C(a_{max} + a_{min})|g|_{H^1(\Omega)}).$$

Moreover for the solution U^1 of (2.3.71), using (2.3.70) and (2.3.71), we have

$$(a\nabla(u - U^1), \nabla\phi) = 0, \quad \phi \in H_0^1. \tag{2.3.82}$$

Hence if $\Phi \in S_h(\Omega)$, $\Phi = g$ on $\partial\Omega_D$ we have , using (2.3.82)

$$\begin{aligned}(a\nabla(u - U^1), \nabla(u - U^1)) &= (a\nabla(u - U^1), \nabla(u - \Phi)) \\ &\leq a_{max}|u - U^1|_{H^1(\Omega)}|u - \Phi|_{H^1(\Omega)},\end{aligned}$$

and so

$$|u - U^1|_{H^1(\Omega)}^2 \leq \left(\frac{a_{max}}{a_{min}}\right)|u - U^1|_{H^1(\Omega)}|u - \Phi|_{H^1(\Omega)},$$

which proves the required estimate.

For the solution of (2.3.72) we have

$$|\tilde{u} - U^2|_{H^1(\Omega)} \leq \left(\frac{a_{max}}{a_{min}}\right) \inf_{\substack{\Phi \in S_h(\Omega) \\ \Phi = g \text{ on } \partial\Omega_D}} |\tilde{u} - \Phi|_{H^1(\Omega)},$$

where $\tilde{u}$ is the exact solution of the weak problem: Find $\tilde{u} \in H^1(\Omega)$ with $\tilde{u} = g$ on $\partial\Omega_D$ and such that

$$(\overline{a}\nabla\tilde{u}, \nabla\phi) = (f, \phi), \quad \phi \in H_0^1.$$

Note that since $\overline{a}$ is h independent, so is $\tilde{u}$. Hence by the triangle inequality

$$|u - U^2|_{H^1(\Omega)} \leq |u - \tilde{u}|_{H^1(\Omega)} + \left(\frac{a_{max}}{a_{min}}\right) \inf_{\substack{\Phi \in S_h(\Omega) \\ \Phi = g \text{ on } \partial\Omega_D}} |\tilde{u} - \Phi|_{H^1(\Omega)}. \tag{2.3.83}$$

To prove (2.3.81) we now have to bound $|u - \tilde{u}|_{H^1(\Omega)}$. Assuming (2.3.80), we know that $u - \tilde{u} \in H_0^1$ and

$$\begin{aligned} (a\nabla u, \nabla\phi) &= (f, \phi) \\ (\overline{a}\nabla\tilde{u}, \nabla\phi) &= (f, \phi), \end{aligned}$$

for all $\phi \in H_0^1$. Hence

$$(a\nabla u - \overline{a}\nabla\tilde{u}, \nabla\phi) = 0,$$

for all $\phi \in H_0^1$. Hence

$$(a\nabla(u - \tilde{u}), \nabla\phi) = ((\overline{a} - a)\nabla\tilde{u}, \nabla\phi).$$

Putting $\phi = u - \tilde{u}$ and using the Cauchy–Schwartz inequality implies

$$a_{min}|u - \tilde{u}|^2_{H^1(\Omega)} \leq \|\overline{a} - a\|_{L_\infty(\Omega)}|\tilde{u}|_{H^1(\Omega)}|u - \tilde{u}|_{H^1(\Omega)}.$$

Hence

$$|u - \tilde{u}|_{H^1(\Omega)} \leq a_{min}^{-1}\|\overline{a} - a\|_{L_\infty(\Omega)}|\tilde{u}|_{H^1(\Omega)}. \tag{2.3.84}$$

Now we also know by analogy with the previous lemma

$$\begin{aligned} |\tilde{u}|_{H^1(\Omega)} &\leq a_{min}^{-1}(\|f\|_{L_2(\Omega)} + (a_{min} + a_{max})|g|_{H^1(\Omega)}), \\ &\leq C a_{min}^{-1}(1 + a_{max} + a_{min}), \end{aligned}$$

where the second inequality is by our assumptions (**A2**), (**A3**). Thus

$$|u - \tilde{u}|_{H^1(\Omega)} \leq C a_{min}^{-2}(1 + a_{max} + a_{min})\|\overline{a} - a\|_{L_\infty} \tag{2.3.85}$$

Now for each T, $\boldsymbol{x} \in T$ we have

$$\begin{aligned}
|(\overline{a} - a)(\boldsymbol{x})| &= \left| \frac{1}{\mathcal{A}(T)^{-1} \int_T a^{-1}(\boldsymbol{y})d\boldsymbol{y}} - \frac{1}{\mathcal{A}(T)^{-1} \int_T a^{-1}(\boldsymbol{x})d\boldsymbol{y}} \right| \\
&= \mathcal{A}(T) \left| \frac{\int_T a^{-1}(\boldsymbol{x})d\boldsymbol{y} - \int_T a^{-1}(\boldsymbol{y})d\boldsymbol{y}}{\int_T a^{-1}(\boldsymbol{y})d\boldsymbol{y} \int_T a^{-1}(\boldsymbol{x})d\boldsymbol{y}} \right| \\
&\leq \mathcal{A}(T)^{-1} a_{max}^2 \left| \int_T (a^{-1}(\boldsymbol{x}) - a^{-1}(\boldsymbol{y}))d\boldsymbol{y} \right| \\
&\leq a_{max}^2 \|a^{-1}(\boldsymbol{x}) - a^{-1}(.)\|_{L_\infty(T)} \\
&\leq \left(\frac{a_{max}}{a_{min}} \right)^2 \|a(\boldsymbol{x}) - a(.)\|_{L_\infty(T)} \\
&\leq M \left(\frac{a_{max}}{a_{min}} \right)^2 h,
\end{aligned}$$

by (2.3.80) and the mean value theorem. Hence (2.3.85) implies

$$|u - \tilde{u}|_{H^1(\Omega)} \leq CM \frac{a_{max}^2}{a_{min}^4}(1 + a_{max} + a_{min})h.$$

Thus we also have

$$|\tilde{u} - \Phi|_{H^1(\Omega)} \leq CM \frac{a_{max}^2}{a_{min}^4}(1 + a_{max} + a_{min})h + |u - \Phi|_{H^1(\Omega)},$$

and inserting these facts into (2.3.83) we get (2.3.81).

■

Hence the rate of convergence in the energy norm of the standard finite element method and the harmonic average finite element method is the same.

Note that our proposed method, (2.3.51)–(2.3.53), differs slightly from the one proposed for the model problem (2.3.54) in that we are mass lumping the term r and *not* calculating its approximate average. This greatly simplifies the computation of the right hand side terms in (2.3.52) and (2.3.53).

2.4 Iterative solution techniques

Before introducing a robust solution method for the discretised semiconductor systems, based upon the well–known method of continuation (see, for example, [60]), we will discuss some of the popular existing techniques for the solution of the systems arising from a discretisation of the semiconductor equations. Any of the discretisation methods introduced in Sections 2.2 and 2.3 will result in a system of the form

$$\boldsymbol{F}_1(\boldsymbol{\Psi}, \boldsymbol{V}, \boldsymbol{W}) := \lambda^2(K(0)\boldsymbol{\Psi} + K_D(0)\boldsymbol{\Psi}_D) + \delta \boldsymbol{g}(\Psi, V, W) = \boldsymbol{0}, \tag{2.4.86}$$

$$\boldsymbol{F}_2(\boldsymbol{\Psi}, \boldsymbol{V}, \boldsymbol{W}) := K(\Psi - V)\boldsymbol{V} + K_D(\Psi - V)\boldsymbol{V}_D - \sigma\rho_v \boldsymbol{r}(\Psi, V, W) = \boldsymbol{0}, \tag{2.4.87}$$

$$\boldsymbol{F}_3(\boldsymbol{\Psi}, \boldsymbol{V}, \boldsymbol{W}) := K(W - \Psi)\boldsymbol{W} + K_D(W - \Psi)\boldsymbol{W}_D + \sigma\rho_w \boldsymbol{r}(\Psi, V, W) = \boldsymbol{0}, \tag{2.4.88}$$

which we wish to solve for the vectors $\boldsymbol{\Psi}$, $\boldsymbol{V}$, $\boldsymbol{W}$, and where Ψ, V, W represent the piecewise linear interpolants to $\boldsymbol{\Psi}$, $\boldsymbol{V}$, $\boldsymbol{W}$ respectively. Note that in (2.4.86), $K_D(0)$ represents the interactions between the unknown nodes and the Dirichlet nodes with known values represented by $\boldsymbol{\Psi}_D$. A similar convention has been adopted in (2.4.87), (2.4.88).

Newton's Method

One immediate choice for the solution of

$$\boldsymbol{F} := (\boldsymbol{F}_1^T,\ \boldsymbol{F}_2^T,\ \boldsymbol{F}_3^T)^T = \boldsymbol{0}, \tag{2.4.89}$$

is *Newton's method*, i.e. given an approximate solution $\boldsymbol{X}^k = (\boldsymbol{\Psi}^{k^T},\ \boldsymbol{V}^{k^T},\ \boldsymbol{W}^{k^T})^T$ we calculated the updated solution by

$$\boldsymbol{X}^{k+1} = \boldsymbol{X}^k - J(\boldsymbol{X}^k)^{-1}\boldsymbol{F}(\boldsymbol{X}^k), \tag{2.4.90}$$

where $J(\boldsymbol{X})$ denotes the Jacobian of $\boldsymbol{F}$ evaluated at $\boldsymbol{X}$. We know by the long–established theory of Newton's method (see for instance [60]), that under appropriate assumptions on $\boldsymbol{F}$ this iteration will converge quadratically to the solution of (2.4.89), provided one exists and provided our initial guess, $\boldsymbol{X}^0$, is close

enough to that solution. Unfortunately, in practice, such a starting guess is often extremely difficult to find. Also the calculation of $J(\boldsymbol{X}^k)^{-1}$ at each step proves prohibitively expensive. For these reasons a full Newton iteration on (2.4.89) is seldom implemented, its use being reserved as a final iterative loop to an approximate method which has already supplied a reasonable solution.

Gummel's method

By far the most common algorithm currently in use to solve the semiconductor system (2.4.86)–(2.4.88) is *Gummel's method* [28]. There are many variants of this scheme, but they all basically consist of iterating a map, the fixed points of which provide the solutions of (2.4.89). One such map is:

$$\mathcal{G} \; : \; (\boldsymbol{V}^k, \boldsymbol{W}^k) \longmapsto (\boldsymbol{V}^{k+1}, \boldsymbol{W}^{k+1}), \tag{2.4.91}$$

defined as follows

Step 1 (Fractional step) Find $\boldsymbol{\Psi}^{k+1}$ such that

$$\lambda^2(K(0)\boldsymbol{\Psi}^{k+1} + K_D(0)\boldsymbol{\Psi}_D) + \delta \boldsymbol{g}(\Psi^{k+1}, V^k, W^k) = \mathbf{0}. \tag{2.4.92}$$

Step 2 Find $\boldsymbol{V}^{k+1}$ such that

$$K(\Psi^{k+1} - V^k)\boldsymbol{V}^{k+1} + K_D(\Psi^{k+1} - V^k)\boldsymbol{V}_D - \sigma\rho_v \boldsymbol{r}(\Psi^{k+1}, V^k, W^k) = \mathbf{0}. \tag{2.4.93}$$

Step 3 Find $\boldsymbol{W}^{k+1}$ such that

$$K(W^k - \Psi^{k+1})\boldsymbol{W}^{k+1} + K_D(W^k - \Psi^{k+1})\boldsymbol{W}_D + \sigma\rho_w \boldsymbol{r}(\Psi^{k+1}, V^{k+1}, W^k) = \mathbf{0}. \tag{2.4.94}$$

This is nothing more than a nonlinear block Gauss–Seidel iteration applied to the discretisations of the three coupled PDEs modelling a semiconducting device. Alternatively it can be viewed as a decoupled approximate block Newton method, where only approximations to the diagonal blocks of the Jacobian are inverted at each step. Each iterate of the Gummel algorithm requires first the solution of

the (semilinear) electrostatic potential equation (2.4.92), with the quasi–Fermi potentials frozen. Using this updated electrostatic potential and the existing guesses for the quasi–Fermi potentials, a new electron quasi–Fermi potential is found by solving the linearised continuity equation (2.4.93). A new hole quasi–Fermi potential is found analogously using (2.4.94) and this process is repeated to convergence. Clearly the implementation of this algorithm is far simpler than a full Newton iteration scheme. The convergence properties of such an algorithm will be studied in detail in Chapter 4 for the case of a one–dimensional device, and Chapter 5 for a two–dimensional device.

Block Newton method

The final algorithm that we consider here is something of a half–way house between a full Newton iteration and Gummel's method. This is known as the *block Newton scheme.* This again is an approximate Newton method and involves the following iterative process: Given approximate solutions Ψ^k, V^k and W^k, calculate the updated solutions by

$$\Psi^{k+1} = \Psi^k - J_1(\Psi^k, V^k, W^k)^{-1} F_1(\Psi^k, V^k, W^k), \tag{2.4.95}$$

$$V^{k+1} = V^k - J_2(\Psi^k, V^k, W^k)^{-1} F_2(\Psi^k, V^k, W^k), \tag{2.4.96}$$

$$W^{k+1} = W^k - J_3(\Psi^k, V^k, W^k)^{-1} F_3(\Psi^k, V^k, W^k). \tag{2.4.97}$$

Here $J_1(\Psi^k, V^k, W^k)$ is the derivative of F_1 with respect to Ψ, evaluated at (Ψ^k, V^k, W^k). Similarly J_2 is the derivative of F_2 with respect to V and J_3 is the derivative of F_3 with respect W. Note that J_1, J_2, J_3 are just the blocks on the diagonal of the Jacobian matrix J. Hence this represents an approximate Newton method where we have neglected the off-diagonal blocks of the Jacobian matrix. In some practical applications this has been seen to converge quadratically. Once again, a good initial guess is required for this method, but it has the advantage of having three smaller Jacobian matrices to invert rather than the one large one present in the full Newton method.

This iteration scheme is introduced in [4]. In that paper discretisation, scaling procedures and the efficient solution of the resulting nonlinear equations are discussed. The companion paper [23] addresses the physical aspects of the governing equations and presents numerical results from various actual device simulations.

2.5 A continuation method in one dimension

We conclude this chapter by describing a continuation scheme which provides a robust solution method for the semiconductor equations. We introduce the scheme via the following abstract setting. Suppose we have a system of the form

$$\boldsymbol{F}(\boldsymbol{X}, k) = \boldsymbol{0}, \tag{2.5.98}$$

where $\boldsymbol{X} \in \mathbb{R}^m$, $k \in [0,1]$, $\boldsymbol{F} : \mathbb{R}^{m+1} \longmapsto \mathbb{R}^m$ for some $m \in \mathbb{N}$, and which we wish to solve for $k = 1$. Then our strategy is as follows. Assume we have a solution, $\boldsymbol{X}$, for the system

$$\boldsymbol{F}(\boldsymbol{X}, k_i) = \boldsymbol{0}, \quad \text{where} \quad 0 \leq k_i < 1 \quad \text{is fixed.}$$

Then we wish to solve

$$\boldsymbol{F}(\boldsymbol{X}, k_{i+1}) = \boldsymbol{0} \quad \text{where} \quad k_{i+1} = k_i + \Delta k_i. \tag{2.5.99}$$

We make the assumption that we have a solution when $k_0 = 0$. By differentiating (2.5.98) with respect to k we obtain

$$J(\boldsymbol{X}, k)\frac{d\boldsymbol{X}}{dk} + \boldsymbol{F}_k(\boldsymbol{X}, k) = \boldsymbol{0},$$

where $J(\boldsymbol{X}, k)$ denotes the Jacobian of $\boldsymbol{F}$ with respect to $\boldsymbol{X}$, evaluated at $(\boldsymbol{X}, k)$. Hence

$$\frac{d\boldsymbol{X}}{dk} = -J(\boldsymbol{X}, k)^{-1}\boldsymbol{F}_k(\boldsymbol{X}, k). \tag{2.5.100}$$

If we now approximate (2.5.100) using one step of Euler's method and write $\boldsymbol{X}^i = \boldsymbol{X}(k_i)$ we obtain $\boldsymbol{X}^E$, where

$$\frac{\boldsymbol{X}^E - \boldsymbol{X}^i}{\Delta k_i} = -J(\boldsymbol{X}^i, k_i)^{-1}\boldsymbol{F}_k(\boldsymbol{X}^i, k_i)$$

and hence

$$\boldsymbol{X}^E = \boldsymbol{X}^i - \Delta k_i J(\boldsymbol{X}^i, k_i)^{-1} \boldsymbol{F}_k(\boldsymbol{X}^i, k_i). \tag{2.5.101}$$

We then use $\boldsymbol{X}^E$ as the starting guess for a Newton iteration for solving (2.5.99). If this is found to be diverging after a few iterations, then we reduce step size Δk_i and start again. This type of continuation scheme is often called a "predictor–corrector" method, where we have predicted a starting guess $\boldsymbol{X}^E$ using Euler's method and then corrected it to $\boldsymbol{X}^{i+1}$ by a Newton iteration.

Implementation details

We now apply this idea to the semiconductor equations in one dimension. It should be pointed out that the concept is just as easily applied to the equations modelling a two or three dimensional device, however the coding of the algorithm will be a more demanding exercise. We use the approximate finite element method, introduced in Section 2.2, to discretise the equations. Recall that we require $(\Psi, V, W) \in S_h(\Omega)^3$ satisfying the boundary equations (2.2.4), (2.2.5) and the discrete equations (2.2.23)–(2.2.25) for all $1 \leq p \leq n$. Here we will not consider the harmonic average method, although the continuation method will work equally as well with it. In matrix form the equations may be written: Find $\boldsymbol{\Psi}, \boldsymbol{V}, \boldsymbol{W} \in I\!R^{n+2}$ such that

$$\begin{aligned} \Psi_0 &= \beta_0 + \alpha_0, \quad V_0 = \alpha_0 = W_0 \\ \Psi_{n+1} &= \beta_1 + \alpha_0 + k(\alpha_1 - \alpha_0), \quad V_{n+1} = \alpha_0 + k(\alpha_1 - \alpha_0) = W_{n+1} \end{aligned}$$

and

$$\lambda^2 \tilde{K}(0)\boldsymbol{\Psi} + \delta \boldsymbol{g}(\Psi, V, W) = \boldsymbol{0} \tag{2.5.102}$$

$$\tilde{K}(\Psi - V)\boldsymbol{V} - \sigma\rho_v \boldsymbol{r}(\Psi, V, W) = \boldsymbol{0} \tag{2.5.103}$$

$$\tilde{K}(W - \Psi)\boldsymbol{W} + \sigma\rho_w \boldsymbol{r}(\Psi, V, W) = \boldsymbol{0} \tag{2.5.104}$$

where

$$\tilde{K}(\Delta)_{ij} = \int_0^1 \exp(\Delta)\phi_i'\phi_j', \quad i = 1, \ldots, n, \ j = 0, \ldots, n+1,$$

$$
\begin{aligned}
(\boldsymbol{g}(\Psi, V, W))_i &= \langle \exp(\Psi - V) - \exp(W - \Psi) - d/\delta, \phi_i \rangle, \quad i = 1, \ldots, n, \\
(\boldsymbol{r}(\Psi, V, W))_i &= \langle r(\Psi, V, W), \phi_i \rangle, \quad i = 1, \ldots, n.
\end{aligned}
$$

Here we have incorporated the boundary values into the vectors $\boldsymbol{\Psi}$, $\boldsymbol{V}$ and $\boldsymbol{W}$. We can think of the solution vector, $\boldsymbol{X} = (\boldsymbol{\Psi}^T, \boldsymbol{V}^T, \boldsymbol{W}^T)^T$, of (2.5.102)–(2.5.104) as $\boldsymbol{X} = \boldsymbol{X}(k)$ where we would like to solve the system for $k = 1$.

First note that $\boldsymbol{X}(0)$ is the solution of the zero current problem which collapses to: Find $\boldsymbol{\Psi} \in I\!R^{n+2}$ such that

$$
\Psi_0 = \beta_0 + \alpha_0, \quad \Psi_{n+1} = \beta_1 + \alpha_0,
$$

and

$$
\lambda^2 K(0)\boldsymbol{\Psi} + \delta \boldsymbol{g}(\Psi, 0, 0) = \boldsymbol{0},
$$

which is easily solved for $\boldsymbol{\Psi}$ by using Newton's method or the quasi–Newton method described in Chapter 3.

Secondly notice that, $\boldsymbol{g}$ and $\boldsymbol{r}$ are not explicitly dependent on k, (i.e. $\partial \boldsymbol{g}/\partial k = \partial \boldsymbol{r}/\partial k = 0$. This is very useful when evaluating derivatives with respect to k later. Now (2.5.102)–(2.5.104) can be written in the form (2.5.98) where

$$
\boldsymbol{F}(\boldsymbol{X}, k) = \boldsymbol{F}(\boldsymbol{X}, 0) + \lambda^2 k(\alpha_1 - \alpha_0) K(0)_{n,n+1} \begin{pmatrix} \boldsymbol{e}_n \\ \boldsymbol{0} \\ \boldsymbol{0} \end{pmatrix} +
$$

$$
k(\alpha_1 - \alpha_0) K(\Psi - V)_{n,n+1} \begin{pmatrix} \boldsymbol{0} \\ \boldsymbol{e}_n \\ \boldsymbol{0} \end{pmatrix} + k(\alpha_1 - \alpha_0) K(W - \Psi)_{n,n+1} \begin{pmatrix} \boldsymbol{0} \\ \boldsymbol{0} \\ \boldsymbol{e}_n \end{pmatrix}
$$

and $\boldsymbol{e}_n \in I\!R^n$ has all entries equal to zero apart from the n^{th} which is equal to 1. Then

$$
J(\boldsymbol{X}, k) = J(\boldsymbol{X}, 0) + k(\alpha_1 - \alpha_0) \begin{bmatrix} 0 & 0 & 0 \\ \gamma(\Psi - V) & -\gamma(\Psi - V) & 0 \\ -\gamma(W - \Psi) & 0 & \gamma(W - \Psi) \end{bmatrix}
$$

where $\gamma(\Delta) \in L(\mathbb{R}^n)$, with

$$\gamma_{ij} = \begin{cases} (\exp(\Delta)\phi_j\phi'_{j+1}, \phi'_i) & i = j = n \\ 0 \quad \text{otherwise} \end{cases}$$

and

$$\boldsymbol{F}_k(\boldsymbol{X}, k) = (\alpha_1 - \alpha_0)\{\lambda^2 K(0)_{n,n+1} \begin{pmatrix} \boldsymbol{e}_n \\ \mathbf{0} \\ \mathbf{0} \end{pmatrix} +$$

$$K(\Psi - V)_{n,n+1} \begin{pmatrix} \mathbf{0} \\ \boldsymbol{e}_n \\ \mathbf{0} \end{pmatrix} + K(W - \Psi)_{n,n+1} \begin{pmatrix} \mathbf{0} \\ \mathbf{0} \\ \boldsymbol{e}_n \end{pmatrix}\}$$

From this we see that if we have a code that performs the block Newton iteration (described in Section 2.4) on (2.5.102)–(2.5.104), then we already have the capability of forming $\boldsymbol{F}_k$ and the blocks on the diagonal of J.

This algorithm has been implemented in MATLAB for a simple p–n diode model. We assume this to be equally doped about the point $x = \frac{1}{2}$, which gives rise to a doping profile

$$d = \begin{cases} -1, & x < \frac{1}{2} \\ 0, & x = \frac{1}{2} \\ +1, & x > \frac{1}{2} \end{cases}$$

We have used the statistics proposed in [62], namely

$$\begin{aligned} l &= 10^{-3}, \\ \epsilon &= 1.0359 \times 10^{-12}, \\ U_T &= 0.025852, \\ q &= 1.602189 \times 10^{-19}, \\ \tilde{d} &= 10^{18}, \\ n_i &= 1.22 \times 10^{10}. \end{aligned}$$

This results in the parameter values of

$$\begin{aligned} \lambda^2 &= 1.68 \times 10^{-7}, \\ \delta &= 1.22 \times 10^{-8}, \\ \beta &= 18.22, \\ \sigma &= 3.17 \times 10^{-15}. \end{aligned}$$

We also use the values $\rho_v = 1/1500$, $\rho_w = 1/450$. The equations (2.2.1)–(2.2.3) are discretised with respect to a uniform mesh with an odd number of points interior to the domain. This ensures that the point $x = \frac{1}{2}$ is a mesh point. We assume that the device is in reverse bias; that is $\alpha_0 = 0$ and $\alpha_1 > 0$.

Since we make an approximation in our block Newton approach by neglecting the off–diagonal blocks of the Jacobian matrix we do the same in our Euler prediction step. At present the three decoupled systems in the Euler step are solved using Jacobi iteration. In fact we only perform one step of a Jacobi iteration and regard the first iterate as the desired $\boldsymbol{X}^E$.

Once we have predicted $\boldsymbol{X}^E$ we then perform a block Newton iteration on (2.5.99), regarding the iteration as diverging if the changes in the maximum norms of $\boldsymbol{V}$ and $\boldsymbol{W}$ exceed 100, or too slow if it has not converged after 15 iterations. In either case we reduce the step size and start the predict–correct strategy again. The block Newton iteration is considered converged once the infinity norms of the updates to $\boldsymbol{\Psi}$, $\boldsymbol{V}$ and $\boldsymbol{W}$ have dropped below 10^{-12}. At each new step we attempt to solve for $k_{i+1} = 1$. If this is not successful then we repeatedly half the step size until convergence is achieved. In this respect we are optimistic at each Euler step, hoping that it will provide a sufficiently good starting guess for the block Newton iteration at the desired applied voltage. This, ultimately, will reduce the amount of unnecessary work we do.

Tables 2.1 and 2.2 contain results obtained from this implementation with 19 and 29 interior mesh points respectively. The values 3.87, 19.34, 38.68, 77.36 and 193.41 of α_1 correspond to physical applied voltages of 0.1V, 0.5V, 1.0V, 2.0V

and 5.0V respectively. In both tables we see that since the zero current equation is independent of α_1 then, for fixed h, the number of Newton iterates required for its solution is constant.

α_1	Number of zero current iterations	Number of continuation steps	Values of k	Number of block Newton iterations
3.87	3	1	1.0	9
19.34	3	2	0.25	13
			1.0	7
38.68	3	2	0.125	13
			1.0	11
77.36	3	4	0.0625	13
			0.296875	7
			0.472656	5
			1.0	9
193.41	3	9	0.015625	7
			0.046387	8
			0.165588	4
			0.191664	5
			0.216924	8
			0.229160	6
			0.253249	10
			0.346593	10
			1.0	9

Table 2.1: Results obtained with $h = 1/20$.

Table 2.1 amply demonstrates the robustness of this algorithm. Firstly note that for small enough applied voltages, continuation is not really required. Hence the case $\alpha_1 = 3.87$ is solved at the first attempt. However as the applied voltage

is increased so we require an increased number of continuation steps. Although not apparent from the information in the tables, the block Newton iterations converge quadratically once the approximate solution become close enough to the true solution. Hence the requirement that the block Newton iterations converged within 15 iterates or a new starting guess was obtained, seems a reasonable one. Also note how our strategy for selecting Δk has paid off towards the end of the continuation process. For instance, in the case where $\alpha_1 = 77.36$, we find that attempting to solve for $k = 1$ at the fourth continuation step is sufficient. This has required a much larger value for Δk than in the previous 3 steps and without our optimistic approach we would probably have taken several more continuation steps before solving for the desired boundary condition.

In Table 2.2 we have solved the same set of problems on a finer mesh. Again we see the algorithm to be robust. In fact for small applied voltages we see that the convergence seems to be (almost) unaffected by the decrease in h. Notice how decreasing h has required us to use more continuation steps to solve the problems for $\alpha_1 = 77.36$ and 38.68. However, perhaps surprisingly, we still only require 9 continuation steps to solve the problem for $\alpha_1 = 193.41$. Here again our optimistic approach seems to have paid off, since although the values of k for which a solution was found have changed, we still see a comparatively large value for Δk at the last continuation step.

α_1	Number of zero current iterations	Number of continuation steps	Values of k	Number of block Newton iterations
3.87	4	1	1.0	9
19.34	4	2	0.25	13
			1.0	8
38.68	4	3	0.125	13
			0.5625	9
			1.0	6
77.36	4	8	0.0625	13
			0.296875	9
			0.472656	5
			0.538574	7
			0.596252	8
			0.646721	8
			0.823361	9
			1.0	5
193.41	4	9	0.015625	7
			0.046387	10
			0.165588	4
			0.191664	5
			0.216924	7
			0.241395	10
			0.265102	8
			0.356964	9
			1.0	8

Table 2.2: Results obtained with $h = 1/30$.

Chapter 3

Monotone quasi–Newton iteration schemes

3.1 Introduction

This chapter is concerned with iteration schemes for the solution of nonlinear systems of the form $\boldsymbol{F}(\boldsymbol{x}) = \mathbf{0}$. The work is motivated by the results on quasi-Newton methods in [60]. These schemes inherit the classic quadratic convergence of Newton's method and, in certain cases, the iterates can be shown to converge in a monotonic sequence. Unfortunately, the schemes proposed in [60] rely on convexity of $\boldsymbol{F}$ which we do not have when we consider the function arising from our finite element discretisation of the potential equation in one and two dimensions. Thus we introduce a novel quasi–Newton method with the same desirable properties as those described in [60], but with a slightly less restrictive demands on $\boldsymbol{F}$. Finally we show that this scheme can be successfully applied to the solution of the potential equation.

3.2 Convexity and Newton's method

In this section we summarise some of the results from [60] which provide the foundations for the new results introduced in the sections that follow.

Firstly we will set up some notation. Following [60] we let $L(\mathbb{R}^n, \mathbb{R}^m)$ denote the linear space of linear operators from $\mathbb{R}^n$ to $\mathbb{R}^m$ and, as a short-hand, let $L(\mathbb{R}^n)$ denote the the linear space of linear operators from $\mathbb{R}^n$ to $\mathbb{R}^n$. We let $\|.\|$ denote an arbitrary norm on $\mathbb{R}^n$ or its associated matrix norm depending on context, and let $\|.\|_p$ be the l_p–norm on $\mathbb{R}^n$. Given D, an open subset of $\mathbb{R}^n$, we make the usual definition of *Gateux–differentiable*: we say a mapping $\boldsymbol{F} : D \subset \mathbb{R}^n \longmapsto \mathbb{R}^m$ is Gateux– (or G–) differentiable at a point $\boldsymbol{x}$ of D if there exists a linear operator $J(\boldsymbol{x}) \in L(\mathbb{R}^n, \mathbb{R}^m)$ such that for any $\boldsymbol{h} \in \mathbb{R}^n$

$$\lim_{t \to 0} \frac{1}{t} \|\boldsymbol{F}(\boldsymbol{x} + t\boldsymbol{h}) - \boldsymbol{F}(\boldsymbol{x}) - tJ(\boldsymbol{x})\boldsymbol{h}\| = 0.$$

$J(\boldsymbol{x})$ is called the *Jacobian* of $\boldsymbol{F}$ and can be identified with an $m \times n$ matrix. We write $\boldsymbol{x} \geq \boldsymbol{0}$ to mean $x_p \geq 0$ for all p. If $\boldsymbol{x}$ and $\boldsymbol{y}$ have the same number of components, we write $\boldsymbol{x} \geq \boldsymbol{y}$ if $\boldsymbol{x} - \boldsymbol{y} \geq \boldsymbol{0}$. In this case we denote the set $\{\boldsymbol{\Phi} : \boldsymbol{x} \leq \boldsymbol{\Phi} \leq \boldsymbol{y}\}$ by $[\boldsymbol{x}, \boldsymbol{y}]$. Furthermore, given $A, B \in L(\mathbb{R}^n, \mathbb{R}^m)$ we write $A \leq B$ if and only if $a_{ij} \leq b_{ij},\ i = 1, \ldots, n,\ j = 1, \ldots, m$.

Following [60] we say $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$ are *comparable* if $\boldsymbol{x} \leq \boldsymbol{y}$ or $\boldsymbol{y} \leq \boldsymbol{x}$ and that a mapping $\boldsymbol{F} : D \subset \mathbb{R}^n \longmapsto \mathbb{R}^m$ is *order–convex* on a convex subset $D_0 \subset D$ if

$$\boldsymbol{F}(\lambda \boldsymbol{x} + (1 - \lambda)\boldsymbol{y}) \leq \lambda \boldsymbol{F}(\boldsymbol{x}) + (1 - \lambda)\boldsymbol{F}(\boldsymbol{y}), \tag{3.2.1}$$

where $\boldsymbol{x},\ \boldsymbol{y} \in D_0$ are comparable and $\lambda \in (0, 1)$. If (3.2.1) holds for all $\boldsymbol{x}, \boldsymbol{y} \in D_0, \lambda \in (0, 1)$ then we say that $\boldsymbol{F}$ is *convex* on D_0. Then [60, page 448] provides us with the following theorem.

THEOREM 3.2.1 *Let $\boldsymbol{F} : D \subset \mathbb{R}^n \longmapsto \mathbb{R}^m$ be G–differentiable on the convex set $D_0 \subset D$. Then the following statements are equivalent.*

$$\boldsymbol{F} \text{ is order-convex on } D_0. \tag{3.2.2}$$

$$\boldsymbol{F}(\boldsymbol{y}) - \boldsymbol{F}(\boldsymbol{x}) \geq J(\boldsymbol{x})(\boldsymbol{y} - \boldsymbol{x}) \quad \text{for all comparable } \boldsymbol{x}, \boldsymbol{y} \in D_0. \tag{3.2.3}$$

$$(J(\boldsymbol{y}) - J(\boldsymbol{x}))(\boldsymbol{y} - \boldsymbol{x}) \geq \boldsymbol{0} \quad \text{for all comparable } \boldsymbol{x}, \boldsymbol{y} \in D_0. \tag{3.2.4}$$

Similarly, $\boldsymbol{F}$ is convex on D_0 if and only if the inequalities in (3.2.3) and (3.2.4) hold for all $\boldsymbol{x}, \boldsymbol{y} \in D_0$.

Proof Firstly suppose (3.2.3) holds and for any given comparable $\boldsymbol{x}, \boldsymbol{y} \in D_0$ and $\lambda \in (0,1)$ set $\boldsymbol{z} = \lambda\boldsymbol{x} + (1-\lambda)\boldsymbol{y}$. Then $\boldsymbol{z} \in D_0$ since D_0 is convex and $\boldsymbol{z} = \boldsymbol{y} + \lambda(\boldsymbol{x} - \boldsymbol{y})$, hence $\boldsymbol{z}$ is comparable with $\boldsymbol{x}$ and $\boldsymbol{y}$. It then follows from (3.2.3) that

$$\begin{aligned} \boldsymbol{F}(\boldsymbol{x}) - \boldsymbol{F}(\boldsymbol{z}) &\geq J(\boldsymbol{z})(\boldsymbol{x} - \boldsymbol{z}), \\ \boldsymbol{F}(\boldsymbol{y}) - \boldsymbol{F}(\boldsymbol{z}) &\geq J(\boldsymbol{z})(\boldsymbol{y} - \boldsymbol{z}). \end{aligned}$$

Multiplying the first by λ and the second by $1-\lambda$ and then adding gives

$$\lambda\boldsymbol{F}(\boldsymbol{x}) + (1-\lambda)\boldsymbol{F}(\boldsymbol{y}) - \boldsymbol{F}(\boldsymbol{z}) \geq J(\boldsymbol{z})(\lambda\boldsymbol{x} + (1-\lambda)\boldsymbol{y} - \boldsymbol{z}) = \boldsymbol{0}$$

Hence

$$\lambda\boldsymbol{F}(\boldsymbol{x}) + (1-\lambda)\boldsymbol{F}(\boldsymbol{y}) \geq \boldsymbol{F}(\lambda\boldsymbol{x} + (1-\lambda)\boldsymbol{y})$$

So we have shown that (3.2.3) implies (3.2.2). Conversely now suppose that (3.2.2) holds and $\boldsymbol{x}, \boldsymbol{y} \in D_0$ are comparable. Then for any $t \in (0,1)$,

$$\boldsymbol{F}(t\boldsymbol{y} + (1-t)\boldsymbol{x}) \leq t\boldsymbol{F}(\boldsymbol{y}) + (1-t)\boldsymbol{F}(\boldsymbol{x}).$$

Therefore

$$\boldsymbol{F}(\boldsymbol{y}) - \boldsymbol{F}(\boldsymbol{x}) \geq \frac{1}{t}(\boldsymbol{F}(\boldsymbol{x} + t(\boldsymbol{y} - \boldsymbol{x})) - \boldsymbol{F}(\boldsymbol{x}))$$

and in view of the G-differentiability of $\boldsymbol{F}$, (3.2.3) holds as $t \to 0$.

To show the equivalence of (3.2.3) and (3.2.4), first note that if (3.2.3) holds then adding

$$\boldsymbol{F}(\boldsymbol{y}) - \boldsymbol{F}(\boldsymbol{x}) \geq J(\boldsymbol{x})(\boldsymbol{y} - \boldsymbol{x}) \text{ to } \boldsymbol{F}(\boldsymbol{x}) - \boldsymbol{F}(\boldsymbol{y}) \geq J(\boldsymbol{y})(\boldsymbol{x} - \boldsymbol{y})$$

gives (3.2.4). Conversely if (3.2.4) holds then

$$(\nabla f_i(\boldsymbol{y}) - \nabla f_i(\boldsymbol{x}))^T(\boldsymbol{y} - \boldsymbol{x}) \geq 0, \quad i = 1, \ldots, m \text{ for all comparable } \boldsymbol{x}, \boldsymbol{y} \in D_0,$$

where $f_1, \ldots, f_m$ are the components of $\boldsymbol{F}$. The mean value theorem [60, 3.2.2] gives us $t_i \in (0,1)$ such that

$$f_i(\boldsymbol{y}) - f_i(\boldsymbol{x}) = (\nabla f_i(\boldsymbol{z}^i))^T(\boldsymbol{y} - \boldsymbol{x}) \quad i = 1, \ldots, m$$

where $\boldsymbol{z}^i = \boldsymbol{x} + t_i(\boldsymbol{y} - \boldsymbol{x})$. But each $\boldsymbol{z}^i$ is comparable with $\boldsymbol{x}$ and $\boldsymbol{y}$ so by (3.2.4) again

$$(\nabla f_i(\boldsymbol{z}^i) - \nabla f_i(\boldsymbol{x}))^T(\boldsymbol{y} - \boldsymbol{x}) = \frac{1}{t_i}(\nabla f_i(\boldsymbol{z}^i) - \nabla f_i(\boldsymbol{x}))^T(\boldsymbol{z}^i - \boldsymbol{x}) \geq 0 \quad i = 1, \ldots, m.$$

Hence by the last two equations

$$f_i(\boldsymbol{y}) - f_i(\boldsymbol{x}) = (\nabla f_i(\boldsymbol{z}^i))^T(\boldsymbol{y} - \boldsymbol{x}) \geq (\nabla f_i(\boldsymbol{x}))^T(\boldsymbol{y} - \boldsymbol{x}) \quad i = 1, \ldots, m,$$

and therefore (3.2.3) holds. The proof for convexity is analogous. ■

These equivalence results allow us to show some monotonicity results for Newton's method applied to a function $\boldsymbol{F} : D \subset \mathbb{R}^n \longmapsto \mathbb{R}^n$. We start by listing the assumptions we make on $\boldsymbol{F}$.

- **A1** There exist $\boldsymbol{x}^0, \boldsymbol{y}^0 \in D$ such that

$$\boldsymbol{x}^0 \leq \boldsymbol{y}^0, \quad [\boldsymbol{x}^0, \boldsymbol{y}^0] \subset D, \quad \boldsymbol{F}(\boldsymbol{x}^0) \leq \boldsymbol{0} \leq \boldsymbol{F}(\boldsymbol{y}^0).$$

- **A2** $\boldsymbol{F}$ is continuous on $[\boldsymbol{x}^0, \boldsymbol{y}^0]$.

- **A3** $\boldsymbol{F}$ is G–differentiable on $[\boldsymbol{x}^0, \boldsymbol{y}^0]$.

- **A4** $\boldsymbol{F}$ is order–convex on $[\boldsymbol{x}^0, \boldsymbol{y}^0]$.

- **A5** For each $\boldsymbol{x} \in [\boldsymbol{x}^0, \boldsymbol{y}^0]$, $J(\boldsymbol{x})^{-1}$ exists and is nonnegative.

The relevant results from [60] are then captured in the following theorem.

THEOREM 3.2.2 (Monotone Newton Theorem, see [60, page 451]) *Assume we have a mapping $\boldsymbol{F} : D \subset \mathbb{R}^n \longmapsto \mathbb{R}^n$ such that* **A1–A5** *hold. Then the Newton iterates defined by*

$$\boldsymbol{y}^{k+1} = \boldsymbol{y}^k - J(\boldsymbol{y}^k)^{-1}\boldsymbol{F}(\boldsymbol{y}^k), \quad k = 0, 1, \ldots \tag{3.2.5}$$

satisfy $\boldsymbol{y}^k \searrow \tilde{\boldsymbol{y}} \in [\boldsymbol{x}^0, \boldsymbol{y}^0]$ as $k \to \infty$, and any solution of $\boldsymbol{F}(\boldsymbol{x}) = \boldsymbol{0}$ in $[\boldsymbol{x}^0, \boldsymbol{y}^0]$ is contained in $[\boldsymbol{x}^0, \tilde{\boldsymbol{y}}]$.

Furthermore, if J is continuous at $\tilde{\boldsymbol{y}}$, then $\tilde{\boldsymbol{y}}$ is the unique solution of $\boldsymbol{F}(\boldsymbol{x}) = \boldsymbol{0}$ in $[\boldsymbol{x}^0, \boldsymbol{y}^0]$.

Proof Since $\boldsymbol{F}$ is order–convex and G–differentiable on the convex set $[\boldsymbol{x}^0, \boldsymbol{y}^0]$, then Theorem 3.2.1 with the roles of $\boldsymbol{x}$ and $\boldsymbol{y}$ reversed shows that

$$\boldsymbol{F}(\boldsymbol{y}) - \boldsymbol{F}(\boldsymbol{x}) \leq J(\boldsymbol{y})(\boldsymbol{y} - \boldsymbol{x}), \quad \boldsymbol{x}^0 \leq \boldsymbol{x} \leq \boldsymbol{y} \leq \boldsymbol{y}^0. \tag{3.2.6}$$

We first show by induction that

$$\boldsymbol{y}^0 \geq \boldsymbol{y}^{k-1} \geq \boldsymbol{y}^k \geq \boldsymbol{x}^0, \quad \boldsymbol{F}(\boldsymbol{y}^k) \geq \boldsymbol{0} \quad \text{for all } k. \tag{3.2.7}$$

Let us assume that (3.2.7) holds for some $k \geq 0$, then using (3.2.5) we obtain

$$\boldsymbol{y}^{k+1} = \boldsymbol{y}^k - J(\boldsymbol{y}^k)^{-1}\boldsymbol{F}(\boldsymbol{y}^k) \leq \boldsymbol{y}^k$$

since $J(\boldsymbol{y}^k)^{-1}$ is nonnegative. Also for any $\boldsymbol{x} \in [\boldsymbol{x}^0, \boldsymbol{y}^k]$ we have

$$\begin{aligned} \boldsymbol{x} - J(\boldsymbol{y}^k)^{-1}\boldsymbol{F}(\boldsymbol{x}) &= \boldsymbol{y}^{k+1} - (\boldsymbol{y}^k - \boldsymbol{x}) + J(\boldsymbol{y}^k)^{-1}(\boldsymbol{F}(\boldsymbol{y}^k) - \boldsymbol{F}(\boldsymbol{x})) \\ &\leq \boldsymbol{y}^{k+1} - (\boldsymbol{y}^k - \boldsymbol{x}) + J(\boldsymbol{y}^k)^{-1}J(\boldsymbol{y}^k)(\boldsymbol{y}^k - \boldsymbol{x}) \\ &= \boldsymbol{y}^{k+1} \end{aligned} \tag{3.2.8}$$

where we have used (3.2.6) to obtain the inequality. Hence in particular $\boldsymbol{x}^0 \leq \boldsymbol{x}^0 - J(\boldsymbol{y}^k)^{-1}\boldsymbol{F}(\boldsymbol{x}^0) \leq \boldsymbol{y}^{k+1}$. Also (3.2.6) with $\boldsymbol{y} = \boldsymbol{y}^k$, $\boldsymbol{x} = \boldsymbol{y}^{k+1}$ gives us

$$\begin{aligned} \boldsymbol{F}(\boldsymbol{y}^{k+1}) &\geq \boldsymbol{F}(\boldsymbol{y}^k) + J(\boldsymbol{y}^k)(\boldsymbol{y}^{k+1} - \boldsymbol{y}^k) \\ &= \boldsymbol{F}(\boldsymbol{y}^k) + J(\boldsymbol{y}^k)(-J(\boldsymbol{y}^k)^{-1}\boldsymbol{F}(\boldsymbol{y}^k)) = \boldsymbol{0}. \end{aligned}$$

Thus (3.2.7) holds for $k+1$. Also, since (3.2.7) holds for $k = 0$ it holds for all k by induction.

It follows that $\{\boldsymbol{y}^k\}_{k=0}^{\infty}$, as a bounded, monotonically decreasing sequence, has a limit $\tilde{\boldsymbol{y}} \geq \boldsymbol{x}^0$. Now suppose that $\boldsymbol{z} \in [\boldsymbol{x}^0, \boldsymbol{y}^0]$ is a solution of $\boldsymbol{F}(\boldsymbol{x}) = \boldsymbol{0}$. We know that, analogously to (3.2.8),

$$\boldsymbol{z} = \boldsymbol{z} - J(\boldsymbol{y}^0)^{-1}\boldsymbol{F}(\boldsymbol{z}) \leq \boldsymbol{y}^1$$

and hence, by induction, we see that $\boldsymbol{z} \leq \boldsymbol{y}^k$ for all $k \geq 0$. Therefore $\boldsymbol{z} \leq \tilde{\boldsymbol{y}}$.

Finally if J is continuous at $\tilde{\boldsymbol{y}}$, then there exists a matrix E and an integer k_0 such that $P := J(\tilde{\boldsymbol{y}})^{-1} - E \geq 0$ is nonsingular and $J(\boldsymbol{y}^k)^{-1} \geq P$ for $k \geq k_0$. Hence for $k \geq k_0$

$$\boldsymbol{y}^k - \boldsymbol{y}^{k+1} = J(\boldsymbol{y}^k)^{-1}\boldsymbol{F}(\boldsymbol{y}^k) \geq P\boldsymbol{F}(\boldsymbol{y}^k) \geq 0$$

But $\lim_{k\to\infty}(\boldsymbol{y}^k - \boldsymbol{y}^{k+1}) = 0$ so that $\lim_{k\to\infty}(P\boldsymbol{F}(\boldsymbol{y}^k)) = 0$. The continuity of $\boldsymbol{F}$ at $\tilde{\boldsymbol{y}}$ and the nonsingularity of P then imply that $\boldsymbol{F}(\tilde{\boldsymbol{y}}) = \boldsymbol{0}$. It remains to show the uniqueness of $\tilde{\boldsymbol{y}}$ in $[\boldsymbol{x}^0, \boldsymbol{y}^0]$. Suppose $\boldsymbol{z} \in [\boldsymbol{x}^0, \boldsymbol{y}^0]$ is any other solution of $\boldsymbol{F}(\boldsymbol{x}) = \boldsymbol{0}$. We know that $\boldsymbol{z} \leq \tilde{\boldsymbol{y}}$ and hence by (3.2.3)

$$0 = \boldsymbol{F}(\tilde{\boldsymbol{y}}) - \boldsymbol{F}(\boldsymbol{z}) \geq J(\boldsymbol{z})(\tilde{\boldsymbol{y}} - \boldsymbol{z}).$$

Nonnegativity of J^{-1} then gives $\tilde{\boldsymbol{y}} \leq \boldsymbol{z}$ and hence $\boldsymbol{z} = \tilde{\boldsymbol{y}}$ as required. ■

With a further condition on $\boldsymbol{F}$ we can define a companion Newton iteration that provides a sequence of iterates which are monotonically increasing to the

same limit $\tilde{\boldsymbol{y}}$. To do this we require the following concept. Given a mapping $\boldsymbol{F} : D \subset I\!R^n \longmapsto I\!R^m$, we say that its Jacobian, J, is *isotone* on $D_0 \subset D$ if

$$J(\boldsymbol{x}) \leq J(\boldsymbol{y}) \quad \text{whenever} \quad \boldsymbol{x} \leq \boldsymbol{y}, \quad \boldsymbol{x}, \boldsymbol{y} \in D_0.$$

We now make the assumptions

- **A6** J is isotone on $[\boldsymbol{x}^0, \boldsymbol{y}^0]$.
- **A7** J satisfies the Lipschitz condition

$$\|J(\boldsymbol{x}) - J(\boldsymbol{y})\| \leq \gamma \|\boldsymbol{x} - \boldsymbol{y}\|, \quad \text{for all} \quad \boldsymbol{x}, \boldsymbol{y} \in [\boldsymbol{x}^0, \boldsymbol{y}^0].$$

With this in mind we state the following corollary to Theorem 3.2.2.

COROLLARY 3.2.3 *Suppose we have a mapping $\boldsymbol{F} : D \subset I\!R^n \longmapsto I\!R^n$ such that* **A1–A5** *hold. Moreover, suppose* **A6** *holds then the sequence*

$$\boldsymbol{x}^{k+1} = \boldsymbol{x}^k - J(\boldsymbol{y}^k)^{-1} \boldsymbol{F}(\boldsymbol{x}^k), \quad k = 0, 1, \ldots \tag{3.2.9}$$

satisfies $\boldsymbol{x}^k \nearrow \tilde{\boldsymbol{y}}$ as $k \to \infty$ where the sequence $\{\boldsymbol{y}^k\}_{k=0}^{\infty}$ is generated by (3.2.5).

Also, if in addition J satisfies the Lipschitz condition **A7** *then there is a constant c such that*

$$\|\boldsymbol{y}^{k+1} - \boldsymbol{x}^{k+1}\| \leq c \|\boldsymbol{y}^k - \boldsymbol{x}^k\|^2, \quad k = 0, 1, \ldots \tag{3.2.10}$$

Proof Recall that the proof of Theorem 3.2.1 shows that

$$\boldsymbol{F}(\boldsymbol{y}) - \boldsymbol{F}(\boldsymbol{x}) \leq J(\boldsymbol{y})(\boldsymbol{y} - \boldsymbol{x}), \quad \boldsymbol{x}^0 \leq \boldsymbol{x} \leq \boldsymbol{y} \leq \boldsymbol{y}^0. \tag{3.2.11}$$

We first show by induction that

$$\boldsymbol{x}^0 \leq \boldsymbol{x}^{k-1} \leq \boldsymbol{x}^k \leq \boldsymbol{y}^k, \quad \boldsymbol{F}(\boldsymbol{x}^k) \leq \boldsymbol{0}. \tag{3.2.12}$$

Suppose this holds for some $k \geq 0$, then since $J(\boldsymbol{x})^{-1} > 0$ for all $\boldsymbol{x} \in [\boldsymbol{x}^0, \boldsymbol{y}^0]$

$$\boldsymbol{x}^{k+1} = \boldsymbol{x}^k - J(\boldsymbol{y}^k)^{-1} \boldsymbol{F}(\boldsymbol{x}^k) \geq \boldsymbol{x}^k.$$

Secondly by the properties of J, Theorem 3.2.2, (3.2.9) and (3.2.11) we have

$$\begin{aligned} \boldsymbol{y}^k &\geq \boldsymbol{y}^k - J(\boldsymbol{y}^k)^{-1}\boldsymbol{F}(\boldsymbol{y}^k) = \boldsymbol{x}^{k+1} + (\boldsymbol{y}^k - \boldsymbol{x}^k) + J(\boldsymbol{y}^k)^{-1}(\boldsymbol{F}(\boldsymbol{x}^k) - \boldsymbol{F}(\boldsymbol{y}^k)) \\ &\geq \boldsymbol{x}^{k+1} + (\boldsymbol{y}^k - \boldsymbol{x}^k) - J(\boldsymbol{y}^k)^{-1}(J(\boldsymbol{y}^k)(\boldsymbol{y}^k - \boldsymbol{x}^k)) = \boldsymbol{x}^{k+1}, \end{aligned}$$

and hence by (3.2.11) with $\boldsymbol{y} = \boldsymbol{x}^{k+1}$, $\boldsymbol{x} = \boldsymbol{x}^k$

$$\begin{aligned} \boldsymbol{F}(\boldsymbol{x}^{k+1}) &\leq \boldsymbol{F}(\boldsymbol{x}^k) + J(\boldsymbol{x}^{k+1})(\boldsymbol{x}^{k+1} - \boldsymbol{x}^k) \\ &= \boldsymbol{F}(\boldsymbol{x}^k) + J(\boldsymbol{x}^{k+1})(-J(\boldsymbol{y}^k)^{-1}\boldsymbol{F}(\boldsymbol{x}^k)) \\ &\leq \boldsymbol{F}(\boldsymbol{x}^k) + J(\boldsymbol{y}^k)(-J(\boldsymbol{y}^k)^{-1}\boldsymbol{F}(\boldsymbol{x}^k)) = \boldsymbol{0}, \end{aligned}$$

where we have used the fact that J is isotone to obtain the last inequality. Hence

$$\begin{aligned} \boldsymbol{x}^{k+1} &\leq \boldsymbol{x}^{k+1} - J(\boldsymbol{y}^k)^{-1}\boldsymbol{F}(\boldsymbol{x}^{k+1}) \\ &= \boldsymbol{y}^{k+1} - (\boldsymbol{y}^k - \boldsymbol{x}^{k+1}) + J(\boldsymbol{y}^k)^{-1}(\boldsymbol{F}(\boldsymbol{y}^k) - \boldsymbol{F}(\boldsymbol{x}^{k+1})) \\ &\leq \boldsymbol{y}^{k+1} - (\boldsymbol{y}^k - \boldsymbol{x}^{k+1}) + J(\boldsymbol{y}^k)^{-1}J(\boldsymbol{y}^k)(\boldsymbol{y}^k - \boldsymbol{x}^{k+1}) = \boldsymbol{y}^{k+1}. \end{aligned}$$

Hence we have proved (3.2.12) for $k+1$ and since it holds for $k = 0$, (3.2.12) holds for all k by induction.

Now $\{\boldsymbol{x}^k\}_{k=0}^{\infty}$, as a bounded, monotonically increasing sequence, has a limit $\tilde{\boldsymbol{x}} \leq \tilde{\boldsymbol{y}}$. For any $\boldsymbol{x} \in [\boldsymbol{x}^k, \boldsymbol{y}^0]$ we have

$$\begin{aligned} \boldsymbol{x} - J(\boldsymbol{y}^k)^{-1}\boldsymbol{F}(\boldsymbol{x}) &= \boldsymbol{x}^{k+1} - (\boldsymbol{x}^k - \boldsymbol{x}) + J(\boldsymbol{y}^k)^{-1}(\boldsymbol{F}(\boldsymbol{x}^k) - \boldsymbol{F}(\boldsymbol{x})) \\ &\geq \boldsymbol{x}^{k+1} - (\boldsymbol{x}^k - \boldsymbol{x}) + J(\boldsymbol{y}^k)^{-1}J(\boldsymbol{x})(\boldsymbol{x}^k - \boldsymbol{x}) \\ &\geq \boldsymbol{x}^{k+1} - (\boldsymbol{x}^k - \boldsymbol{x}) + J(\boldsymbol{y}^k)^{-1}J(\boldsymbol{x}^k)(\boldsymbol{x}^k - \boldsymbol{x}) \\ &\geq \boldsymbol{x}^{k+1} - (\boldsymbol{x}^k - \boldsymbol{x}) + J(\boldsymbol{x}^k)^{-1}J(\boldsymbol{x}^k)(\boldsymbol{x}^k - \boldsymbol{x}) = \boldsymbol{x}^{k+1} \end{aligned}$$

where we have made extensive use of the fact that J is isotone with a nonnegative inverse. In particular, if $\boldsymbol{z} \in [\boldsymbol{x}^0, \boldsymbol{y}^0]$ is any solution of $\boldsymbol{F}(\boldsymbol{x}) = \boldsymbol{0}$ then

$$\boldsymbol{z} = \boldsymbol{z} - J(\boldsymbol{y}^0)^{-1}\boldsymbol{F}(\boldsymbol{z}) \geq \boldsymbol{x}^1$$

and hence by induction $\boldsymbol{z} \geq \boldsymbol{x}^k$ for all k. Therefore $\tilde{\boldsymbol{x}} \leq \boldsymbol{z} \leq \tilde{\boldsymbol{y}}$. Since J is isotone on $[\boldsymbol{x}^0, \boldsymbol{y}^0]$ then $J(\boldsymbol{y}^k) \leq J(\boldsymbol{y}^0)$ and hence by the nonnegativity of the inverse we have

$$0 \leq P := J(\boldsymbol{y}^0)^{-1} \leq J(\boldsymbol{y}^k)^{-1}.$$

and hence

$$\begin{aligned} \boldsymbol{x}^{k+1} - \boldsymbol{x}^k &= -J(\boldsymbol{y}^k)^{-1}\boldsymbol{F}(\boldsymbol{x}^k) \\ &\geq -P\boldsymbol{F}(\boldsymbol{x}^k) \geq 0, \end{aligned}$$

and

$$\begin{aligned} \boldsymbol{y}^k - \boldsymbol{y}^{k+1} &= J(\boldsymbol{y}^k)^{-1}\boldsymbol{F}(\boldsymbol{y}^k) \\ &\geq P\boldsymbol{F}(\boldsymbol{y}^k) \geq 0. \end{aligned}$$

Nonsingularity of P and continuity of $\boldsymbol{F}$ then give $\boldsymbol{F}(\tilde{\boldsymbol{x}}) = \boldsymbol{F}(\tilde{\boldsymbol{y}}) = \boldsymbol{0}$. So if $\boldsymbol{z} \in [\boldsymbol{x}^0, \boldsymbol{y}^0]$ is any solution of $\boldsymbol{F}(\boldsymbol{x}) = \boldsymbol{0}$ then by our argument above $\tilde{\boldsymbol{x}} \leq \boldsymbol{z} \leq \tilde{\boldsymbol{y}}$ and

$$0 = \boldsymbol{F}(\tilde{\boldsymbol{y}}) - \boldsymbol{F}(\boldsymbol{z}) \geq J(\boldsymbol{z})(\tilde{\boldsymbol{y}} - \boldsymbol{z}) \text{ and } 0 = \boldsymbol{F}(\boldsymbol{z}) - \boldsymbol{F}(\tilde{\boldsymbol{x}}) \geq J(\tilde{\boldsymbol{x}})(\boldsymbol{z} - \tilde{\boldsymbol{x}}).$$

Whence $\boldsymbol{z} \geq \tilde{\boldsymbol{y}}$ and $\boldsymbol{z} \leq \tilde{\boldsymbol{x}}$. and therefore $\tilde{\boldsymbol{x}} = \boldsymbol{z} = \tilde{\boldsymbol{y}}$ as required. Finally, to prove (3.2.10), suppose that **A7** holds. Then J is continuous on $[\boldsymbol{x}^0, \boldsymbol{y}^0]$ and, since $J(\boldsymbol{x})$ is nonsingular, there exists a β such that $\|J(\boldsymbol{x})^{-1}\| \leq \beta$, $\boldsymbol{x} \in [\boldsymbol{x}^0, \boldsymbol{y}^0]$. It then follows, by the generalised mean value theorem [60, 3.2.12], that

$$\begin{aligned} \|\boldsymbol{y}^{k+1} - \boldsymbol{x}^{k+1}\| &= \|\boldsymbol{y}^k - \boldsymbol{x}^k - J(\boldsymbol{y}^k)^{-1}(\boldsymbol{F}(\boldsymbol{y}^k) - \boldsymbol{F}(\boldsymbol{x}^k))\| \\ &\leq \beta\|J(\boldsymbol{y}^k)(\boldsymbol{y}^k - \boldsymbol{x}^k) - (\boldsymbol{F}(\boldsymbol{y}^k) - \boldsymbol{F}(\boldsymbol{x}^k))\| \leq \beta\gamma\|\boldsymbol{y}^k - \boldsymbol{x}^k\|^2 \end{aligned}$$

and we have shown the quadratic convergence property. ■

3.3 A new quasi–Newton scheme

In this section we discuss a novel quasi–Newton scheme. This was brought about by the need to relax the order–convexity constraint (**A4**) which played such an important role in the results obtained in Section 3.2. This criterion is not met by the nonlinear equation resulting from a finite element discretisation of the potential equation and so a new scheme is proposed which requires only criteria that hold in the case of the potential equation. We introduce the scheme and its properties in abstract form in this section and then in the sections that follow we verify that any assumptions we have made hold for the potential equation discretised by the finite element method in both one and two dimensions.

We begin by stating our new set of assumptions for $\boldsymbol{F} : D \subset I\!R^n \longmapsto I\!R^n$.

- **B1** There exist $\boldsymbol{x}^0, \boldsymbol{y}^0 \in D$ such that

$$\boldsymbol{x}^0 \leq \boldsymbol{y}^0, \quad [\boldsymbol{x}^0, \boldsymbol{y}^0] \subset D, \quad \boldsymbol{F}(\boldsymbol{x}^0) \leq \boldsymbol{0} \leq \boldsymbol{F}(\boldsymbol{y}^0).$$

- **B2** $\boldsymbol{F}$ is continuous on $[\boldsymbol{x}^0, \boldsymbol{y}^0]$.

- **B3** $\boldsymbol{F}$ is G–differentiable on $[\boldsymbol{x}^0, \boldsymbol{y}^0]$.

- **B4** For any comparable $\boldsymbol{x}, \boldsymbol{y} \in [\boldsymbol{x}^0, \boldsymbol{y}^0]$ there exists a mapping $A(\boldsymbol{x}, \boldsymbol{y}) \in L(I\!R^n, I\!R^n)$ such that, if $\boldsymbol{x} \leq \boldsymbol{y}$ then,

$$\boldsymbol{F}(\boldsymbol{y}) - \boldsymbol{F}(\boldsymbol{z}) - A(\boldsymbol{x}, \boldsymbol{y})(\boldsymbol{y} - \boldsymbol{z}) \leq \boldsymbol{0} \leq \boldsymbol{F}(\boldsymbol{x}) - \boldsymbol{F}(\boldsymbol{z}) - A(\boldsymbol{x}, \boldsymbol{y})(\boldsymbol{x} - \boldsymbol{z}),$$

  for all $\boldsymbol{z} \in [\boldsymbol{x}, \boldsymbol{y}]$.

- **B5** For any comparable $\boldsymbol{x}, \boldsymbol{y} \in [\boldsymbol{x}^0, \boldsymbol{y}^0]$, $(A(\boldsymbol{x}, \boldsymbol{y}))^{-1}$ exists and is nonnegative.

Hence the order–convexity assumption **A4** has been replaced by the weaker assumption **B4**.

Then for $k = 0, 1, \ldots$ we define the quasi–Newton iterates as follows

$$\boldsymbol{x}^{k+1} = \boldsymbol{x}^k - (A(\boldsymbol{x}^k, \boldsymbol{y}^k))^{-1}\boldsymbol{F}(\boldsymbol{x}^k), \tag{3.3.13}$$

$$\boldsymbol{y}^{k+1} = \boldsymbol{y}^k - (A(\boldsymbol{x}^k, \boldsymbol{y}^k))^{-1}\boldsymbol{F}(\boldsymbol{y}^k). \tag{3.3.14}$$

Firstly it should be noted that for each k, (3.3.13) and (3.3.14) can be performed in parallel. Secondly it transpires that we can show the scheme (3.3.13), (3.3.14) converges monotonically to some $\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{y}} \in [\boldsymbol{x}^0, \boldsymbol{y}^0]$ with $\tilde{\boldsymbol{x}} \leq \tilde{\boldsymbol{y}}$. Moreover, under a further assumption on $\boldsymbol{F}$ we can show that $\tilde{\boldsymbol{x}} = \tilde{\boldsymbol{y}}$. These results are given in full in Theorem 3.3.3 the proof of which is expedited by the following two lemmas.

LEMMA 3.3.1 *Assume we have a mapping $\boldsymbol{F} : D \subset \mathbb{R}^n \longmapsto \mathbb{R}^n$ such that* **B1–B5** *hold. Let $\{\boldsymbol{x}^k\}_{k=0}^{\infty}, \{\boldsymbol{y}^k\}_{k=0}^{\infty}$ be defined by (3.3.13), (3.3.14). If $\boldsymbol{x}^0 \leq \boldsymbol{x}^k \leq \boldsymbol{y}^k \leq \boldsymbol{y}^0$ for some k, then for any $\boldsymbol{z} \in [\boldsymbol{x}^k, \boldsymbol{y}^k]$*

$$\boldsymbol{x}^{k+1} \leq \boldsymbol{z} - (A(\boldsymbol{x}^k, \boldsymbol{y}^k))^{-1}\boldsymbol{F}(\boldsymbol{z}) \leq \boldsymbol{y}^{k+1}$$

Proof First note, by (3.3.14)

$$\begin{aligned} \boldsymbol{z} - (A(\boldsymbol{x}^k, \boldsymbol{y}^k))^{-1}\boldsymbol{F}(\boldsymbol{z}) &= \boldsymbol{y}^{k+1} - (\boldsymbol{y}^k - \boldsymbol{z}) + (A(\boldsymbol{x}^k, \boldsymbol{y}^k))^{-1}(\boldsymbol{F}(\boldsymbol{y}^k) - \boldsymbol{F}(\boldsymbol{z})) \\ &\leq \boldsymbol{y}^{k+1} - (\boldsymbol{y}^k - \boldsymbol{z}) + (A(\boldsymbol{x}^k, \boldsymbol{y}^k))^{-1}A(\boldsymbol{x}^k, \boldsymbol{y}^k)(\boldsymbol{y}^k - \boldsymbol{z}) \\ &= \boldsymbol{y}^{k+1} \end{aligned}$$

where the inequality follows from **B4**. This proves the right hand inequality. The left hand inequality is obtained analogously. ■

LEMMA 3.3.2 *Assume we have a mapping $\boldsymbol{F} : D \subset \mathbb{R}^n \longmapsto \mathbb{R}^n$ such that* **B1–B5** *hold. Let $\{\boldsymbol{x}^k\}_{k=1}^{\infty}, \{\boldsymbol{y}^k\}_{k=1}^{\infty}$ be defined by (3.3.13), (3.3.14). Then for all $k \geq 0$*

$$\boldsymbol{x}^0 \leq \boldsymbol{x}^k \leq \boldsymbol{y}^k \leq \boldsymbol{y}^0 \tag{3.3.15}$$

$$\boldsymbol{F}(\boldsymbol{x}^k) \leq \boldsymbol{0} \leq \boldsymbol{F}(\boldsymbol{y}^k) \tag{3.3.16}$$

$$\boldsymbol{y}^{k+1} \leq \boldsymbol{y}^k \tag{3.3.17}$$

$$\boldsymbol{x}^k \leq \boldsymbol{x}^{k+1} \tag{3.3.18}$$

Proof First note that if (3.3.15) and (3.3.16) hold, then (by **B5**, (3.3.13) and (3.3.14)), so do (3.3.17) and (3.3.18). Also note that $\boldsymbol{x}^0$ and $\boldsymbol{y}^0$ satisfy (3.3.15) and (3.3.16). Hence the result is true for $k = 0$. Now suppose that the result holds for some $k \geq 0$. We need to show that (3.3.15) and (3.3.16) hold for $k+1$. Putting $\boldsymbol{z} = \boldsymbol{x}^k$ in Lemma 3.3.1 yields

$$\boldsymbol{x}^{k+1} = \boldsymbol{x}^k - (A(\boldsymbol{x}^k, \boldsymbol{y}^k))^{-1}\boldsymbol{F}(\boldsymbol{x}^k) \leq \boldsymbol{y}^{k+1}.$$

Hence we have $\boldsymbol{x}^0 \leq \boldsymbol{x}^k \leq \boldsymbol{x}^{k+1} \leq \boldsymbol{y}^{k+1} \leq \boldsymbol{y}^k \leq \boldsymbol{y}^0$. On the other hand, putting $\boldsymbol{x} = \boldsymbol{x}^k$, $\boldsymbol{y} = \boldsymbol{y}^k$, $\boldsymbol{z} = \boldsymbol{x}^{k+1}$ in **B4** we obtain

$$\begin{aligned} \mathbf{0} &\leq \boldsymbol{F}(\boldsymbol{x}^k) - \boldsymbol{F}(\boldsymbol{x}^{k+1}) - A(\boldsymbol{x}^k, \boldsymbol{y}^k)(\boldsymbol{x}^k - \boldsymbol{x}^{k+1}) \\ &= -\boldsymbol{F}(\boldsymbol{x}^{k+1}) + \boldsymbol{F}(\boldsymbol{x}^k) - A(\boldsymbol{x}^k, \boldsymbol{y}^k)(A(\boldsymbol{x}^k, \boldsymbol{y}^k))^{-1}\boldsymbol{F}(\boldsymbol{x}^k) \\ &= -\boldsymbol{F}(\boldsymbol{x}^{k+1}) \end{aligned}$$

Thus $\boldsymbol{F}(\boldsymbol{x}^{k+1}) \leq \mathbf{0}$. Similarly we can prove that $\boldsymbol{F}(\boldsymbol{y}^{k+1}) \geq \mathbf{0}$, and the result follows by induction. ■

We now assume further that

- **B6** There exists a constant M such that, for all $\boldsymbol{x}, \boldsymbol{y} \in [\boldsymbol{x}^0, \boldsymbol{y}^0]$,

$$\|A(\boldsymbol{x}, \boldsymbol{y})\|_\infty \leq M.$$

- **B7** For all $\boldsymbol{x}, \boldsymbol{y} \in [\boldsymbol{x}^0, \boldsymbol{y}^0]$, $\boldsymbol{x} < \boldsymbol{y}$, there exists a $\boldsymbol{z} \in [\boldsymbol{x}, \boldsymbol{y}]$ such that

$$\boldsymbol{F}(\boldsymbol{y}) - \boldsymbol{F}(\boldsymbol{x}) = J(\boldsymbol{z})(\boldsymbol{y} - \boldsymbol{x}) \quad \text{and} \quad J(\boldsymbol{z}) \text{ is nonsingular.}$$

Note that **B7** is a mean value property for $\boldsymbol{F}$. This does not imply order–convexity of $\boldsymbol{F}$ as the examples in Sections 3.4 and 3.5 will show. We can now prove the convergence of the quasi-Newton method.

THEOREM 3.3.3 *Assume we have a mapping* $\boldsymbol{F} : D \subset \mathbb{R}^n \longmapsto \mathbb{R}^n$ *such that* **B1–B5** *hold. The sequences* $\{\boldsymbol{x}^k\}_{k=1}^{\infty}, \{\boldsymbol{y}^k\}_{k=1}^{\infty}$ *defined by (3.3.13), (3.3.14) converge to limits* $\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{y}}$ *respectively with* $\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{y}} \in [\boldsymbol{x}^0, \boldsymbol{y}^0]$ *and* $\tilde{\boldsymbol{x}} \leq \tilde{\boldsymbol{y}}$. *Also if* $\tilde{\boldsymbol{z}}$ *is any solution of* $\boldsymbol{F}(\boldsymbol{x}) = \boldsymbol{0}$ *in* $[\boldsymbol{x}^0, \boldsymbol{y}^0]$ *then* $\tilde{\boldsymbol{x}} \leq \tilde{\boldsymbol{z}} \leq \tilde{\boldsymbol{y}}$. *Now, if* **B6** *holds, then* $\boldsymbol{F}(\tilde{\boldsymbol{x}}) = \boldsymbol{0} = \boldsymbol{F}(\tilde{\boldsymbol{y}})$ *and furthermore, if* **B7** *holds, then* $\tilde{\boldsymbol{x}} = \tilde{\boldsymbol{y}}$ *which is the unique solution of* $\boldsymbol{F}(\boldsymbol{x}) = \boldsymbol{0}$ *in* $[\boldsymbol{x}^0, \boldsymbol{y}^0]$.

Proof By Lemma 3.3.2, $\{\boldsymbol{y}^k\}_{k=1}^{\infty} \subset [\boldsymbol{x}^0, \boldsymbol{y}^0]$ is a bounded monotonically decreasing sequence. Hence

$$\boldsymbol{y}^k \searrow \tilde{\boldsymbol{y}} \in [\boldsymbol{x}^0, \boldsymbol{y}^0].$$

Similarly it follows that

$$\boldsymbol{x}^k \nearrow \tilde{\boldsymbol{x}} \in [\boldsymbol{x}^0, \boldsymbol{y}^0].$$

Lemma 3.3.2 also implies that $\tilde{\boldsymbol{x}} \leq \tilde{\boldsymbol{y}}$. Now observe that if $\tilde{\boldsymbol{z}}$ is any solution of $\boldsymbol{F}(\boldsymbol{z}) = \boldsymbol{0}$ then, Lemma 3.3.1 with $k = 0$ gives

$$\boldsymbol{x}^1 \leq \tilde{\boldsymbol{z}} \leq \boldsymbol{y}^1.$$

Continuing by induction shows that $\boldsymbol{x}^k \leq \tilde{\boldsymbol{z}} \leq \boldsymbol{y}^k$ for all k and hence $\tilde{\boldsymbol{x}} \leq \tilde{\boldsymbol{z}} \leq \tilde{\boldsymbol{y}}$.

We now show that $\boldsymbol{F}(\tilde{\boldsymbol{x}}) = \boldsymbol{0} = \boldsymbol{F}(\tilde{\boldsymbol{y}})$. To do this use **B6** and the definition of $\boldsymbol{y}^{k+1}$, to obtain

$$\|\boldsymbol{F}(\boldsymbol{y}^k)\|_\infty = \|A(\boldsymbol{x}^k, \boldsymbol{y}^k)(\boldsymbol{y}^k - \boldsymbol{y}^{k+1})\|_\infty \leq M\|\boldsymbol{y}^k - \boldsymbol{y}^{k+1}\|_\infty \ \to 0, \text{ as } k \to \infty.$$

Hence, by continuity, $\boldsymbol{F}(\tilde{\boldsymbol{y}}) = \boldsymbol{0}$. Similarly $\boldsymbol{F}(\tilde{\boldsymbol{x}}) = \boldsymbol{0}$. If we now assume that **B7** holds then we have

$$\boldsymbol{0} = \boldsymbol{F}(\tilde{\boldsymbol{y}}) - \boldsymbol{F}(\tilde{\boldsymbol{x}}) = J(\boldsymbol{z})(\tilde{\boldsymbol{y}} - \tilde{\boldsymbol{x}})$$

where $\boldsymbol{z} \in [\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{y}}] \subset [\boldsymbol{x}^0, \boldsymbol{y}^0]$. Hence by our assumption $\tilde{\boldsymbol{x}} = \tilde{\boldsymbol{y}}$ as required. ■

With a further assumption on the matrix $A(\boldsymbol{x}, \boldsymbol{y})$ we can show that the new scheme also converges quadratically. This is done with the following corollary.

COROLLARY 3.3.4 *If, for any* $\boldsymbol{x}, \boldsymbol{y} \in [\boldsymbol{x}^0, \boldsymbol{y}^0]$ *with* $\boldsymbol{x} \leq \boldsymbol{y}$, *there exists* $\gamma, \alpha \in I\!R$ *such that*

$$\begin{aligned} \|A(\boldsymbol{x},\boldsymbol{y}) - J(\boldsymbol{z})\| &\leq \gamma\|\boldsymbol{x}-\boldsymbol{y}\| \quad \textit{for all} \quad \boldsymbol{z} \in [\boldsymbol{x},\boldsymbol{y}], \\ \|A(\boldsymbol{x},\boldsymbol{y})^{-1}\| &\leq \alpha, \end{aligned}$$

then there is a constant c *such that*

$$\|\boldsymbol{y}^{k+1} - \boldsymbol{x}^{k+1}\| \leq c\|\boldsymbol{y}^k - \boldsymbol{x}^k\|^2.$$

Proof By (3.3.13), (3.3.14) we have

$$\begin{aligned} \|\boldsymbol{y}^{k+1} - \boldsymbol{x}^{k+1}\| &= \|\boldsymbol{y}^k - \boldsymbol{x}^k - A(\boldsymbol{x}^k,\boldsymbol{y}^k)^{-1}(\boldsymbol{F}(\boldsymbol{y}^k) - \boldsymbol{F}(\boldsymbol{x}^k))\| \\ &\leq \|A(\boldsymbol{x}^k,\boldsymbol{y}^k)^{-1}\|\,\|A(\boldsymbol{x}^k,\boldsymbol{y}^k)(\boldsymbol{y}^k - \boldsymbol{x}^k) - (\boldsymbol{F}(\boldsymbol{y}^k) - \boldsymbol{F}(\boldsymbol{x}^k))\| \\ &\leq \alpha\|(A(\boldsymbol{x}^k,\boldsymbol{y}^k) - J(\boldsymbol{z}))(\boldsymbol{y}^k - \boldsymbol{x}^k)\|, \end{aligned}$$

for some $\boldsymbol{z} \in [\boldsymbol{x}^k, \boldsymbol{y}^k]$ by **B7**, and hence

$$\|\boldsymbol{y}^{k+1} - \boldsymbol{x}^{k+1}\| \leq \alpha\gamma\|\boldsymbol{y}^k - \boldsymbol{x}^k\|^2$$

as required. ■

3.4 The potential equation in one dimension

We now show that the potential equation in one dimension, discretised by the finite element method described in Chapter 2, does not satisfy an order–convexity property **A4** on any suitable set of vectors. Hence the results in Section 3.2 cannot be applied. However, we will then construct a mapping $A(\boldsymbol{x},\boldsymbol{y}) : I\!R^n \times I\!R^n \longmapsto L(I\!R^n)$ which satisfies the criteria **B1–B7** laid out in Section 3.3. Hence, using this mapping in the quasi–Newton scheme (3.3.13), (3.3.14) the convergence results in Section 3.3 will hold.

Firstly we define the set

$$B(\Lambda) = \{(V, W) \in S_h(\Lambda) \times S_h(\Lambda) : \underline{\alpha} \leq V, W \leq \overline{\alpha}\}. \tag{3.4.19}$$

where

$$\overline{\alpha} = \max_{i=0,1} \alpha_i, \quad \underline{\alpha} = \min_{i=0,1} \alpha_i \tag{3.4.20}$$

Then recall from Chapter 2, given any $(V, W) \in B(\Lambda)$, our finite element discretisation of the potential equation in one dimension is to find $\Psi \in S_h(\Lambda)$ satisfying

$$\Psi(0) = \beta_0 + \alpha_0, \quad \Psi(1) = \beta_1 + \alpha_1$$

and

$$(\lambda^2 \Psi', \phi_p') + \langle \delta\{\exp(\Psi - V) - \exp(W - \Psi)\} - d, \phi_p \rangle = 0 \tag{3.4.21}$$

for all $1 \leq p \leq n$. Now write $\boldsymbol{\Psi}, \boldsymbol{V}, \boldsymbol{W}$ for the vectors of values of Ψ, V, W at the interior nodes of our mesh (2.2.7) and let $\boldsymbol{\Psi}_D$ be the vector $(\beta_0 + \alpha_0, \beta_1 + \alpha_1)^T$ representing the Dirichlet data. Let $\boldsymbol{d}$ be the vector with $d_p = d(x_p)$, $1 \leq p \leq n$ and, considering $\boldsymbol{V}$ and $\boldsymbol{W}$ to be fixed, define

$$\boldsymbol{g}(\boldsymbol{\Psi}) = \operatorname{diag}\left\{\overline{\boldsymbol{h}}\right\} (\delta\{\exp(\boldsymbol{\Psi} - \boldsymbol{V}) - \exp(\boldsymbol{W} - \boldsymbol{\Psi})\} - \boldsymbol{d})$$

where $\overline{h_i}$ is given by (2.2.13).

Then (3.4.21) may be written as the problem of finding the solution $\tilde{\boldsymbol{\Psi}}$ to $\boldsymbol{F}(\boldsymbol{\Psi}) = \boldsymbol{0}$ where

$$\boldsymbol{F}(\boldsymbol{\Psi}) = \lambda^2(K\boldsymbol{\Psi} + K_D\boldsymbol{\Psi}_D) + \boldsymbol{g}(\boldsymbol{\Psi}) \tag{3.4.22}$$

Here K is the stiffness matrix representing coupling between nodes interior to Λ whereas K_D represents the coupling between each of the interior nodes and the two Dirichlet nodes, $x_0 = 0$, $x_{n+1} = 1$. Obviously K is just the stiffness matrix arising from the finite element approximation to the Laplacian with a Dirichlet condition at each end. Before proceeding, let us consider this problem

with Dirichlet data taken to be unity. Store this data in the vector $\mathbf{1}_D$. The exact solution is the vector $\mathbf{1}$ which is unity at each of the interior nodes. Hence we have the relation

$$K\mathbf{1} + K_D\mathbf{1}_D = \mathbf{0}. \tag{3.4.23}$$

The system (3.4.22) has Jacobian

$$\begin{aligned} J(\boldsymbol{\Psi}) &= \lambda^2 K + \delta \mathrm{diag}\{\overline{\boldsymbol{h}}\}\mathrm{diag}\{\exp(\boldsymbol{\Psi} - \boldsymbol{V}) + \exp(\boldsymbol{W} - \boldsymbol{\Psi})\} \\ &=: \lambda^2 K + \boldsymbol{g}_{\Psi}(\boldsymbol{\Psi}), \end{aligned} \tag{3.4.24}$$

where $\boldsymbol{g}_{\Psi}$ is the (diagonal) Jacobian matrix of $\boldsymbol{g}$. Firstly set

$$x^0 = \underline{\alpha} + \min_{x \in \overline{\Lambda}} \sinh^{-1}(d(x)/2\delta), \tag{3.4.25}$$

$$y^0 = \overline{\alpha} + \max_{x \in \overline{\Lambda}} \sinh^{-1}(d(x)/2\delta), \tag{3.4.26}$$

then define

$$\begin{aligned} \boldsymbol{x}^0 = x^0\mathbf{1}, \quad \boldsymbol{x}^0_D = x^0\mathbf{1}_D, \\ \boldsymbol{y}^0 = y^0\mathbf{1}, \quad \boldsymbol{y}^0_D = y^0\mathbf{1}_D. \end{aligned}$$

Clearly $\boldsymbol{x}^0 \leq \boldsymbol{y}^0$ and using (3.4.22), (3.4.23)

$$\begin{aligned} \boldsymbol{F}(\boldsymbol{x}^0) &= \lambda^2(K\boldsymbol{x}^0 + K_D\boldsymbol{\Psi}_D) + \boldsymbol{g}(\boldsymbol{x}^0) \\ &= \lambda^2 K_D(\boldsymbol{\Psi}_D - \boldsymbol{x}^0_D) + \boldsymbol{g}(\boldsymbol{x}^0). \end{aligned}$$

Since $\boldsymbol{\Psi}_D \geq \boldsymbol{x}^0_D$ and K_D contains only non–positive entries, the first term is non–positive. Also since $(V, W) \in B(\Lambda)$, we have

$$\begin{aligned} \boldsymbol{g}(\boldsymbol{x}^0) &= \mathrm{diag}\{\overline{\boldsymbol{h}}\}(\delta(\exp(\boldsymbol{x}^0 - \boldsymbol{V}) - \exp(\boldsymbol{W} - \boldsymbol{x}^0)) - \boldsymbol{d}) \\ &\leq \mathrm{diag}\{\overline{\boldsymbol{h}}\}(2\delta\sinh((x^0 - \underline{\alpha})\mathbf{1}) - \boldsymbol{d}) \leq \mathbf{0}. \end{aligned}$$

Hence $\boldsymbol{F}(\boldsymbol{x}^0) \leq \mathbf{0}$, $\boldsymbol{F}(\boldsymbol{y}^0) \geq \mathbf{0}$ is proved similarly. Thus we have a convex set $D_0 := [\boldsymbol{x}^0, \boldsymbol{y}^0]$ with

$$\boldsymbol{x}^0 \leq \boldsymbol{y}^0 \quad \text{and} \quad \boldsymbol{F}(\boldsymbol{x}^0) \leq \mathbf{0} \leq \boldsymbol{F}(\boldsymbol{y}^0)$$

Hence we have verified that **B1** holds for $\boldsymbol{F}$ defined by (3.4.22). Recall that, by the remarks made in Chapter 1, in most practical device simulations $x^0 < 0$. If this is not the case then in fact $\boldsymbol{F}$ *is* order–convex on $[\boldsymbol{x}^0, \boldsymbol{y}^0]$ and the monotone method of Theorem 3.2.2 can be applied to the potential equation. However when $x^0 < 0$ consider $\boldsymbol{F}$ given by (3.4.22) and let $V = W$. Then for any $\boldsymbol{x}, \boldsymbol{y} \in [\boldsymbol{x}^0, \boldsymbol{y}^0]$ such that

$$\boldsymbol{x} < \boldsymbol{y} \leq \mathbf{0}$$

we have

$$\begin{aligned}(J(\boldsymbol{x}) - J(\boldsymbol{y}))(\boldsymbol{x} - \boldsymbol{y}) &= (\boldsymbol{g}_\Psi(\boldsymbol{x}) - \boldsymbol{g}_\Psi(\boldsymbol{y}))(\boldsymbol{x} - \boldsymbol{y}) \\ &= \delta \text{diag}\left\{\overline{\boldsymbol{h}}\right\} \text{diag}\left\{\cosh(\boldsymbol{x} - \boldsymbol{V}) - \cosh(\boldsymbol{y} - \boldsymbol{V})\right\}(\boldsymbol{x} - \boldsymbol{y})\end{aligned}$$

But $\boldsymbol{x} - \boldsymbol{V} < \boldsymbol{y} - \boldsymbol{V} \leq \mathbf{0}$ and hence $\cosh(\boldsymbol{x} - \boldsymbol{V}) > \cosh(\boldsymbol{y} - \boldsymbol{V}) \geq \mathbf{0}$. Therefore

$$(J(\boldsymbol{x}) - J(\boldsymbol{y}))(\boldsymbol{x} - \boldsymbol{y}) < \mathbf{0}.$$

Hence, by Theorem 3.2.1, when $x^0 < 0$, $\boldsymbol{F}$ is not order–convex on $D_0 = [\boldsymbol{x}^0, \boldsymbol{y}^0]$ and unfortunately the monotone Newton theory of Section 3.2 does not follow. The assumption that $x^0 < 0$ is in no way restrictive, since for all realistic device models d will be negative in some part of the domain. However, now consider the following quasi–Newton method. With our start iterates $\boldsymbol{x}^0, \boldsymbol{y}^0$, for $k \geq 0$ set

$$\boldsymbol{x}^{k+1} = \boldsymbol{x}^k - (A(\boldsymbol{x}^k, \boldsymbol{y}^k))^{-1}\boldsymbol{F}(\boldsymbol{x}^k) \tag{3.4.27}$$

$$\boldsymbol{y}^{k+1} = \boldsymbol{y}^k - (A(\boldsymbol{x}^k, \boldsymbol{y}^k))^{-1}\boldsymbol{F}(\boldsymbol{y}^k) \tag{3.4.28}$$

where

$$A(\boldsymbol{x}^k, \boldsymbol{y}^k) := \max\{J(\boldsymbol{x}^k), J(\boldsymbol{y}^k)\}. \tag{3.4.29}$$

(The maximum in (3.4.29) is taken *elementwise.*) As we shall now see, this satisfies the conditions **B2–B7** of Section 3.3 and hence converges.

THEOREM 3.4.1 *The sequences* $\{\boldsymbol{x}^k\}_{k=1}^{\infty}$ *and* $\{\boldsymbol{y}^k\}_{k=1}^{\infty}$ *defined by (3.4.27) and (3.4.28) converge to the same limit* $\tilde{\boldsymbol{z}}$ *which is the unique solution of* $\boldsymbol{F}(\boldsymbol{x}) = \boldsymbol{0}$ *in* $[\boldsymbol{x}^0, \boldsymbol{y}^0]$. *Furthermore there exists a constant* C, *independent of* h *and* k *such that*

$$\|\boldsymbol{y}^{k+1} - \boldsymbol{x}^{k+1}\|_2 \leq C\|\boldsymbol{y}^k - \boldsymbol{x}^k\|_2^2 \tag{3.4.30}$$

Proof The proof that the scheme (3.4.21), (3.4.22) converges to a unique limit merely involves verifying the assumptions **B2–B7** of Section 3.3 for this particular quasi–Newton method. Firstly note that it is trivial to show that **B2, B3** hold for this particular $\boldsymbol{F}$. Secondly the matrix K is positive definite and symmetric. In addition it has non–positive off–diagonal elements and is irreducible. For any vector $\boldsymbol{\Psi}$, $J(\boldsymbol{\Psi})$ also has this property. Hence by [71, page 85, Corollary 3] both K and $J(\boldsymbol{\Psi})$ are nonsingular and

$$K^{-1} > 0, \quad J(\boldsymbol{\Psi})^{-1} > 0. \tag{3.4.31}$$

Hence by (3.4.29), for any $\boldsymbol{x}, \boldsymbol{y} \in [\boldsymbol{x}^0, \boldsymbol{y}^0]$, and therefore for any comparable $\boldsymbol{x}, \boldsymbol{y} \in [\boldsymbol{x}^0, \boldsymbol{y}^0]$, $A(\boldsymbol{x}, \boldsymbol{y})$ is nonsingular and $(A(\boldsymbol{x}, \boldsymbol{y}))^{-1} > 0$, and we have shown **B5** holds.

Now note, by (3.4.24) we can write

$$A(\boldsymbol{x}, \boldsymbol{y}) = \lambda^2 K + \boldsymbol{g}_{\Psi}^{(xy)}, \quad \text{where } \boldsymbol{g}_{\Psi}^{(xy)} = \max\{\boldsymbol{g}_{\Psi}(\boldsymbol{x}), \boldsymbol{g}_{\Psi}(\boldsymbol{y})\}. \tag{3.4.32}$$

Now, for any $\boldsymbol{x}, \boldsymbol{y} \in [\boldsymbol{x}^0, \boldsymbol{y}^0]$ with $\boldsymbol{x} < \boldsymbol{y}$ let $\boldsymbol{z} \in [\boldsymbol{x}, \boldsymbol{y}]$, then using the mean value theorem and the fact that $\boldsymbol{g}_{\Psi}$ is diagonal,

$$\begin{aligned} \boldsymbol{F}(\boldsymbol{y}) - \boldsymbol{F}(\boldsymbol{z}) - A(\boldsymbol{x}, \boldsymbol{y})(\boldsymbol{y} - \boldsymbol{z}) &= \boldsymbol{g}(\boldsymbol{y}) - \boldsymbol{g}(\boldsymbol{z}) - \boldsymbol{g}_{\Psi}^{(xy)}(\boldsymbol{y} - \boldsymbol{z}) \\ &= (\boldsymbol{g}_{\Psi}(\boldsymbol{\eta}) - \boldsymbol{g}_{\Psi}^{(xy)})(\boldsymbol{y} - \boldsymbol{z}) \end{aligned}$$

for some $\boldsymbol{\eta} \in [\boldsymbol{x}, \boldsymbol{y}]$. Note that $\boldsymbol{g}_\Psi(\boldsymbol{\eta})$ is diagonal and $\boldsymbol{g}_\Psi(\boldsymbol{\eta})_{ii}$ is a convex function of η_i, hence

$$\boldsymbol{g}_\Psi(\boldsymbol{\eta})_{ii} \leq \max\{\boldsymbol{g}_\Psi(\boldsymbol{x})_{ii}, \boldsymbol{g}_\Psi(\boldsymbol{y})_{ii}\} \equiv (\boldsymbol{g}_\Psi^{(xy)})_{ii}.$$

Therefore the left–hand inequality of **B4** follows. The right–hand inequality is proved analogously. Hence **B1–B5** hold and the monotonicity results of Lemma 3.3.2 follow. Also for any $\boldsymbol{x}, \boldsymbol{y} \in [\boldsymbol{x}^0, \boldsymbol{y}^0]$, using (3.4.29) we obtain the trivial bound

$$\begin{aligned} \|A(\boldsymbol{x}, \boldsymbol{y})\|_\infty &\leq \lambda^2 \|K\|_\infty + \|\boldsymbol{g}_\Psi^{(xy)}\|_\infty \\ &\leq \lambda^2 \|K\|_\infty + \|\boldsymbol{g}_\Psi^{(x^0 y^0)}\|_\infty =: M \end{aligned}$$

where it should be noted that M is independent of k. Hence we have shown **B6**. Now note that for any $\boldsymbol{x}, \boldsymbol{y} \in [\boldsymbol{x}^0, \boldsymbol{y}^0]$ with $\boldsymbol{x} < \boldsymbol{y}$ we have

$$\begin{aligned} \boldsymbol{F}(\boldsymbol{y}) - \boldsymbol{F}(\boldsymbol{x}) &= \lambda^2 K(\boldsymbol{y} - \boldsymbol{x}) + \boldsymbol{g}(\boldsymbol{y}) - \boldsymbol{g}(\boldsymbol{x}) \\ &= \lambda^2 K(\boldsymbol{y} - \boldsymbol{x}) + \boldsymbol{g}_\Psi(\boldsymbol{z})(\boldsymbol{y} - \boldsymbol{x}) = J(\boldsymbol{z})(\boldsymbol{y} - \boldsymbol{x}) \end{aligned}$$

for some $\boldsymbol{z} \in [\boldsymbol{x}, \boldsymbol{y}]$, and therefore **B7** holds. Hence it follows that the convergence properties of Theorem 3.3.3 hold for the quasi–Newton scheme (3.4.27), (3.4.28).

Now to derive the convergence estimate (3.4.30), use (3.4.31), (3.4.32) and the fact that $\boldsymbol{g}_\Psi$ is a nonnegative matrix, to obtain

$$(\lambda^2 K)^{-1} A(\boldsymbol{x}^k, \boldsymbol{y}^k) = I + (\lambda^2 K)^{-1} \boldsymbol{g}_\Psi^{(x^k y^k)} \geq I.$$

Then using the fact that **B5** holds, it follows that

$$(\lambda^2 K)^{-1} \geq (A(\boldsymbol{x}^k, \boldsymbol{y}^k))^{-1} > 0. \tag{3.4.33}$$

Now note that K and A are symmetric matrices and hence so are there inverses. Furthermore all the eigenvalues of K^{-1} and A^{-1} will be real. Now, for any symmetric matrix P, let $\mu_{max}(P)$ denote its largest eigenvalue. Then since

$A(\boldsymbol{x}^k, \boldsymbol{y}^k)^{-1}$ is positive definite and symmetric we have, by the Rayleigh Quotient Theorem

$$\mu_{max}(A(\boldsymbol{x}^k, \boldsymbol{y}^k)^{-1}) = \max_{\boldsymbol{x} \neq \boldsymbol{0}} \left\{ \frac{\boldsymbol{x}^T (A(\boldsymbol{x}^k, \boldsymbol{y}^k)^{-1}) \boldsymbol{x}}{\boldsymbol{x}^T \boldsymbol{x}} \right\} = \frac{\tilde{\boldsymbol{x}}^T (A(\boldsymbol{x}^k, \boldsymbol{y}^k)^{-1}) \tilde{\boldsymbol{x}}}{\tilde{\boldsymbol{x}}^T \tilde{\boldsymbol{x}}},$$

for some $\tilde{\boldsymbol{x}}$. Since $A(\boldsymbol{x}^k, \boldsymbol{y}^k)^{-1}$ is also positive, the Perron–Frobenius Theorem [26, page 118] implies that we can choose $\tilde{\boldsymbol{x}}$ to be nonnegative. Hence, using (3.4.33) and the Rayleigh Quotient Theorem applied to $(\lambda^2 K)^{-1}$ it follows that

$$\mu_{max}(A(\boldsymbol{x}^k, \boldsymbol{y}^k)^{-1}) = \frac{\boldsymbol{x}^T (A(\boldsymbol{x}^k, \boldsymbol{y}^k)^{-1}) \boldsymbol{x}}{\boldsymbol{x}^T \boldsymbol{x}} \leq \frac{\boldsymbol{x}^T (\lambda^2 K)^{-1} \boldsymbol{x}}{\boldsymbol{x}^T \boldsymbol{x}} \leq \mu_{max}((\lambda^2 K)^{-1}).$$

Hence

$$\|A(\boldsymbol{x}^k, \boldsymbol{y}^k)^{-1}\|_2 \leq \|(\lambda^2 K)^{-1}\|_2 = \lambda^{-2} \|K^{-1}\|_2 \leq C \lambda^{-2} h^{-1},$$

with C independent of h, where the last inequality has come from our assumption (2.2.8) and the result given in the Appendix A.1. Therefore

$$\begin{aligned}
\|\boldsymbol{y}^{k+1} - \boldsymbol{x}^{k+1}\|_2 &= \|\boldsymbol{y}^k - \boldsymbol{x}^k - (A(\boldsymbol{x}^k, \boldsymbol{y}^k))^{-1} (\boldsymbol{F}(\boldsymbol{y}^k) - \boldsymbol{F}(\boldsymbol{x}^k))\|_2 \\
&= \|(A(\boldsymbol{x}^k, \boldsymbol{y}^k))^{-1} [A(\boldsymbol{x}^k, \boldsymbol{y}^k)(\boldsymbol{y}^k - \boldsymbol{x}^k) - (\boldsymbol{F}(\boldsymbol{y}^k) - \boldsymbol{F}(\boldsymbol{x}^k))]\|_2 \\
&\leq \|(A(\boldsymbol{x}^k, \boldsymbol{y}^k))^{-1}\|_2 \|A(\boldsymbol{x}^k, \boldsymbol{y}^k)(\boldsymbol{y}^k - \boldsymbol{x}^k) - (\boldsymbol{F}(\boldsymbol{y}^k) - \boldsymbol{F}(\boldsymbol{x}^k))\|_2 \\
&\leq \|(A(\boldsymbol{x}^k, \boldsymbol{y}^k))^{-1}\|_2 \|(A(\boldsymbol{x}^k, \boldsymbol{y}^k) - J(\boldsymbol{\eta}))(\boldsymbol{y}^k - \boldsymbol{x}^k)\|_2 \\
&\leq \lambda^{-2} h^{-1} \|A(\boldsymbol{x}^k, \boldsymbol{y}^k) - J(\boldsymbol{\eta})\|_2 \|\boldsymbol{y}^k - \boldsymbol{x}^k\|_2
\end{aligned}$$

for some $\boldsymbol{\eta} \in [\boldsymbol{x}^k, \boldsymbol{y}^k]$. Now $A(\boldsymbol{x}^k, \boldsymbol{y}^k) - J(\boldsymbol{\eta})$ is a diagonal matrix and an easy application of the mean value theorem yields

$$|(A(\boldsymbol{x}^k, \boldsymbol{y}^k) - J(\boldsymbol{\eta}))_{ii}| \leq \delta |\overline{h_i}| \; |(\exp(\gamma_i + V_i) + \exp(W_i - \gamma_i))(y_i^k - x_i^k)| \quad (3.4.34)$$

where $\boldsymbol{\gamma} \in [\boldsymbol{x}^k, \boldsymbol{y}^k] \subset [\boldsymbol{x}^0, \boldsymbol{y}^0]$. Recall that each $\overline{h_i}$ is simply half the sum of the lengths of the intervals which meet at node i and hence $\overline{h_i} \leq Ch$ for each i. Also since $(V, W) \in B(\Lambda)$ and since $\boldsymbol{x}^0$ and $\boldsymbol{y}^0$ are independent of h, we have by (3.4.34)

$$\|A(\boldsymbol{x}^k, \boldsymbol{y}^k) - J(\boldsymbol{\eta})\|_2 \leq C \delta h \|\boldsymbol{y}^k - \boldsymbol{x}^k\|_2$$

and so

$$\|\boldsymbol{y}^{k+1} - \boldsymbol{x}^{k+1}\|_2 \leq C\|\boldsymbol{y}^k - \boldsymbol{x}^k\|_2^2,$$

with C independent of h and k, as required. ■

Hence, by careful refinement of the procedure used in the proof of Corollary 3.3.4, we have shown quadratic convergence which is also independent of the mesh size h. *Mesh independence* results for classical Newton's method for general nonlinear systems have appeared in recent literature, see for example [17]. However, as is usual for Newton's method, these results depend on good estimates for the inverse of the Jacobian near the root. Such estimates are not needed for the analysis of Gummel's method which we give here. Indeed it is not clear whether such estimates can be obtained for our complicated PDE problem containing several small parameters.

3.5 The potential equation in two dimensions

In a similar manner to the previous section, we now show that the potential equation in two dimensions, discretised by the finite element method described in Chapter 2, does not satisfy an order–convexity property. We then go on to define a quasi–Newton scheme that satisfies the assumptions **B1–B7** and hence exhibits the convergence properties of Section 3.3.

We define the appropriate set in which (V, W) lie as follows.

$$B(\Omega) = \{(V, W) \in S_h(\Omega) \times S_h(\Omega) : \underline{\alpha} \leq V, W \leq \overline{\alpha}\}. \tag{3.5.35}$$

where

$$\overline{\alpha} = \max_i \alpha_i, \quad \underline{\alpha} = \min_i \alpha_i \tag{3.5.36}$$

and α_i are the applied voltages appearing in (2.3.37). Then as in Chapter 2, given any $(V, W) \in B(\Omega)$, we seek $\Psi \in S_h(\Omega)$ satisfying (2.3.39) and such that

$$\lambda^2(\nabla\Psi, \nabla\phi_p) + \langle\delta\{\exp(\Psi - V) - \exp(W - \Psi)\} - d, \phi_p\rangle = 0 \tag{3.5.37}$$

at all nodes $p \notin \partial\Omega_D$. Again write $\boldsymbol{\Psi}, \boldsymbol{V}, \boldsymbol{W}$ for the vectors of values of Ψ, V, W at the nodes on $\Omega\backslash\partial\Omega_D$ and write $\boldsymbol{\Psi}_D$ for the vector of values of Ψ at the nodes on $\partial\Omega_D$. Let $\boldsymbol{d}$ be the vector with $d_p = d(p)$ for each node p, and, (considering $\boldsymbol{V}$ and $\boldsymbol{W}$ to be fixed,) define

$$\boldsymbol{g}(\boldsymbol{\Psi}) = \operatorname{diag}\{\boldsymbol{w}\}\,(\delta\{\exp(\boldsymbol{\Psi} - \boldsymbol{V}) - \exp(\boldsymbol{W} - \boldsymbol{\Psi})\} - \boldsymbol{d})$$

where $\boldsymbol{w}$ is the vector of weights from the quadrature rule (2.3.45). Then (3.5.37) may be written as the problem of finding the solution $\tilde{\boldsymbol{\Psi}}$ to $\boldsymbol{F}(\boldsymbol{\Psi}) = \boldsymbol{0}$ where

$$\boldsymbol{F}(\boldsymbol{\Psi}) = \lambda^2(K\boldsymbol{\Psi} + K_D\boldsymbol{\Psi}_D) + \boldsymbol{g}(\boldsymbol{\Psi}). \tag{3.5.38}$$

Here K is now the stiffness matrix representing coupling between nodes of $\Omega\backslash\partial\Omega_D$ and K_D represents the coupling between nodes on $\partial\Omega_D$ and $\Omega\backslash\partial\Omega_D$ as before. Arguing as in Section 3.4 we have the relation

$$K\mathbf{1} + K_D\mathbf{1}_D = \mathbf{0}. \tag{3.5.39}$$

where $\mathbf{1}_D$ is the vector with value unity at each node of $\partial\Omega_D$ and $\mathbf{1}$ is the vector with value unity at each node in $\Omega\backslash\partial\Omega_D$. Similarly we can write the Jacobian of (3.5.38) as

$$\begin{aligned} J(\boldsymbol{\Psi}) &= \lambda^2 K + \delta\operatorname{diag}\{\boldsymbol{w}\}\operatorname{diag}\{\exp(\boldsymbol{\Psi} - \boldsymbol{V}) + \exp(\boldsymbol{W} - \boldsymbol{\Psi})\} \\ &=: \lambda^2 K + \boldsymbol{g}_{\Psi}(\boldsymbol{\Psi}), \end{aligned} \tag{3.5.40}$$

where again $\boldsymbol{g}_{\Psi}$ is the (diagonal) Jacobian matrix of $\boldsymbol{g}$. Setting

$$x^0 = \underline{\alpha} + \min_{\boldsymbol{x}\in\overline{\Omega}} \sinh^{-1}(d(\boldsymbol{x})/2\delta), \tag{3.5.41}$$

$$y^0 = \overline{\alpha} + \max_{\boldsymbol{x}\in\overline{\Omega}} \sinh^{-1}(d(\boldsymbol{x})/2\delta), \tag{3.5.42}$$

and defining

$$\boldsymbol{x}^0 = x^0\mathbf{1}, \quad \boldsymbol{x}^0_D = x^0\mathbf{1}_D,$$

$$\boldsymbol{y}^0 = y^0\mathbf{1}, \quad \boldsymbol{y}^0_D = y^0\mathbf{1}_D.$$

then arguing as in Section 3.4 it is easily shown that we have a convex set $D_0 := [\boldsymbol{x}^0, \boldsymbol{y}^0]$ with

$$\boldsymbol{x}^0 \leq \boldsymbol{y}^0 \quad \text{and} \quad \boldsymbol{F}(\boldsymbol{x}^0) \leq \boldsymbol{0} \leq \boldsymbol{F}(\boldsymbol{y}^0)$$

Hence we have shown **B1** also holds for (3.5.38). $\boldsymbol{F}$ is only order–convex if $x^0 \geq 0$. Since $x^0 < 0$ occurs often in practice it is unrealistic to assume $\boldsymbol{F}$ is order–convex.

However, now consider the following quasi–Newton method. With our start iterates $\boldsymbol{x}^0, \boldsymbol{y}^0$, for $k \geq 0$ set

$$\boldsymbol{x}^{k+1} = \boldsymbol{x}^k - (A(\boldsymbol{x}^k, \boldsymbol{y}^k))^{-1}\boldsymbol{F}(\boldsymbol{x}^k) \tag{3.5.43}$$

$$\boldsymbol{y}^{k+1} = \boldsymbol{y}^k - (A(\boldsymbol{x}^k, \boldsymbol{y}^k))^{-1}\boldsymbol{F}(\boldsymbol{y}^k) \tag{3.5.44}$$

where

$$A(\boldsymbol{x}^k, \boldsymbol{y}^k) := \max\{J(\boldsymbol{x}^k), J(\boldsymbol{y}^k)\}. \tag{3.5.45}$$

(The maximum in (3.5.45) is again taken *elementwise.*) Before proving any convergence results about the scheme (3.5.43), (3.5.44) we state and prove the following lemma which will help us derive the properties of the stiffness matrix K.

LEMMA 3.5.1 *Consider any triangle T of our fine mesh, described in Chapter 2, with nodes N_1, N_2, N_3. Let ϕ_i, $i = 1, 2, 3$ be the usual hat basis functions based on N_1, N_2, N_3. Consider*

$$(\nabla\phi_i, \nabla\phi_j)|_T = \int_T \nabla\phi_i . \nabla\phi_j .$$

Let E_i be the edge opposite N_i, h_i be the perpendicular distance of N_i from E_i and let γ_{ij} be the angle between the normals to E_i and E_j acting in to T. Then

$$(\nabla\phi_i, \nabla\phi_j)|_T = cos(\gamma_{ij})\frac{1}{h_i h_j}\mathcal{A}(T)$$

Proof Since $\phi_i(N_j) = \delta_{ij}$ we have

$$\nabla\phi_i \perp E_i \text{ for all } i. \tag{3.5.46}$$

Now since the distance of N_i from E_i is defined to be h_i and since $\phi_i = 1$ at N_i, 0 on E_i, we have by (3.5.46)

$$\nabla\phi_i = \frac{1}{h_i}\hat{\boldsymbol{n}}_i,$$

where $\hat{\boldsymbol{n}}_i$ is the unit normal to E_i. Then

$$\begin{aligned}(\nabla\phi_i, \nabla\phi_j)|_T &= \int_T \frac{1}{h_i h_j}(\hat{\boldsymbol{n}}_i.\hat{\boldsymbol{n}}_j) \\ &= \cos(\gamma_{ij})\frac{1}{h_i h_j}\mathcal{A}(T),\end{aligned}$$

as required. ■

We now give the theorem which provides the convergence results for our quasi–Newton scheme applied to the discrete two–dimensional potential equation.

THEOREM 3.5.2 *The sequences* $\{\boldsymbol{x}^k\}_{k=1}^{\infty}$ *and* $\{\boldsymbol{y}^k\}_{k=1}^{\infty}$, *defined by (3.5.43) and (3.5.44), converge to the same limit* $\tilde{\boldsymbol{z}}$ *which is the unique solution of* $\boldsymbol{F}(\boldsymbol{x}) = \boldsymbol{0}$ *in* $[\boldsymbol{x}^0, \boldsymbol{y}^0]$. *Furthermore there exists a constant* C, *independent of* h *and* k *such that*

$$\|\boldsymbol{y}^{k+1} - \boldsymbol{x}^{k+1}\|_2 \leq C\|\boldsymbol{y}^k - \boldsymbol{x}^k\|_2^2$$

Proof Again the proof consists of a verification of the assumptions (**B2–B7**) made in Section 3.3. We begin by noting that **B2, B3** are trivial to obtain. Now consider the stiffness matrix K arising from a finite element discretisation of the Laplacian with zero Dirichlet condition on $\partial\Omega_D$. For this we have

$$K_{ij} = \int_\Omega \nabla\phi_i.\nabla\phi_j = \sum_T \int_T \nabla\phi_i.\nabla\phi_j$$

where the sum is over all triangles containing both N_i and N_j. When $i \neq j$ we either have N_i, N_j are two vertices of the same triangle, connected by an edge, or N_i, N_j are not common to any triangle. Clearly in the second case

$$\int_T \nabla\phi_i.\nabla\phi_j = 0 \quad \text{for all } T$$

and hence $K_{ij} = 0$. In the first case N_i, N_j may be vertices of one or two triangles. In either case, providing the triangulation is weakly acute, Lemma 3.5.1 tells us that

$$\int_T \nabla\phi_i . \nabla\phi_j \leq 0$$

and hence $K_{ij} \leq 0$. Also K is clearly symmetric and positive definite, since for any vector $\boldsymbol{x}$

$$\begin{aligned} \boldsymbol{x}^T K \boldsymbol{x} &= \sum_{j,i \notin \partial\Omega_D} (\nabla x_j \phi_j, \nabla x_i \phi_i) \\ &= (\nabla X, \nabla X) \text{ where } X = \sum_j x_j \phi_j \end{aligned}$$

and hence $\boldsymbol{x}^T K \boldsymbol{x} \geq 0$ with equality if and only if $\nabla X = 0$ which occurs if and only if $X =$ constant. Then using the zero Dirichlet condition on $\partial\Omega_D$, this occurs if and only if $X = 0$, i.e. if and only if $\boldsymbol{x} = \mathbf{0}$.

So K is positive definite, symmetric and $K_{ij} \leq 0$ when $i \neq j$. Hence by [71, page 85], K is a Stieltjes matrix which is also irreducible, hence K is an M–matrix. Since, for any vector $\boldsymbol{\Psi}$, $J(\boldsymbol{\Psi})$ is just K plus some additional nonnegative terms on the diagonal, we have that $J(\boldsymbol{\Psi})$ is an M–matrix as well. Therefore by [71, page 85] again we know that K and $J(\boldsymbol{\Psi})$ are nonsingular with

$$K^{-1} > 0, \quad J(\boldsymbol{\Psi})^{-1} > 0.$$

Hence by (3.5.45), for any $\boldsymbol{x}, \boldsymbol{y} \in D_0$, $A(\boldsymbol{x}, \boldsymbol{y})$ is an irreducible M–matrix which is therefore nonsingular with $A(\boldsymbol{x}, \boldsymbol{y})^{-1} > 0$, and we have shown that **B5** holds. The verification of **B4** is analogous to that in the proof of Theorem 3.4.1, as is the trivial bound in **B6**. Hence the scheme (3.5.43), (3.5.44) converges monotonically to the unique solution $\tilde{\boldsymbol{z}}$. It remains to show the quadratic convergence. To do this we use analogous arguments to those in the proof of Theorem 3.4.1 along with the fact that

$$\|K^{-1}\|_2 \leq Ch^{-2}$$

which is obtained, for example, from [39, Section 7.7] and that the weights w_p from the quadrature rule (2.3.45) can be bounded as $w_p \leq Ch^2$ for each node p.

■

Remark 3.5.1 It has recently been pointed out to us (F. A. Potra, private communication) that the basic convergence properties of $\boldsymbol{x}^k$, $\boldsymbol{y}^k$ proved above are a consequence of the much more general theory in [17]. However neither the *a priori* bounds derived in Sections 3.4 and 3.5, nor the mesh–independence of the convergence proved in Theorems 3.4.1 and 3.5.2, are in [17] and so we have taken the opportunity here to give a full elementary proof of the convergence as well.

Chapter 4

Gummel's map in one dimension

4.1 Introduction

In this chapter we shall discuss the convergence of the discrete versions of Gummel's decoupling algorithm for the systems modelling a device in one dimension. In Chapter 3 we have introduced a one–dimensional scheme which allows us to solve the discretised potential equation. Our method of studying the coupled systems will be adaptions of the results found in [37], [40]. We will set the Gummel iteration up as a map on a certain set and then use the contraction mapping theorem to show that the scheme converges to a unique fixed point in that set. This shows essentially linear convergence for small enough applied voltages.

However we will then restrict our attention to a p–n diode in reverse bias. We have found in practice that a slight variant of Gummel's iteration for this problem converges even for large applied voltages. In an attempt to understand this behaviour we will present some new results which, without being completely rigorous, shed some light on what is happening within this iteration.

Finally in this chapter we present some further results for the reverse bias diode which show that the computed potential has a sharp interior layer which mirrors that known to exist in the solution to the undiscretised system (see, for

example [9], [61], [47]).

4.2 General one–dimensional device

Firstly we shall set the recombination rate to zero throughout the device. Recall from Chapter 2, our approximate solution to (2.2.1)–(2.2.3) is defined to be $(\Psi, V, W) \in S_h(\Lambda)^3$ satisfying

$$V(0) = \alpha_0 = W(0), \quad V(1) = \alpha_1 = W(1) \tag{4.2.1}$$

$$\Psi(0) = \alpha_0 + \beta_0, \quad \Psi(1) = \alpha_1 + \beta_1 \tag{4.2.2}$$

and such that

$$\lambda^2(\Psi', \phi_p') + \langle \delta\{\exp(\Psi - V) - \exp(W - \Psi)\} - d, \phi_p \rangle = 0,$$

$$(\exp(\overline{\Psi - V})V', \phi_p') = 0,$$

$$(\exp(\overline{W - \Psi})W', \phi_p') = 0,$$

for $1 \leq p \leq n$. Notice that here we are using the harmonic average approximation for the coefficients in the continuity equations. The results given in this section also hold for the standard finite element method, the proofs requiring only minor modifications. We shall return to the standard finite element scheme when analysing the case of large applied voltages in Section 4.3.

Following our definition of Gummel's map in Chapter 2 we shall iterate the map $(\Psi, V, W) \longmapsto (\tilde{\Psi}, \tilde{V}, \tilde{W})$ defined as follows

$$\lambda^2(\tilde{\Psi}', \phi_p') + \langle \delta\{\exp(\tilde{\Psi} - V) - \exp(W - \tilde{\Psi})\} - d, \phi_p \rangle = 0, \tag{4.2.3}$$

$$(\exp(\overline{\tilde{\Psi} - V})\tilde{V}', \phi_p') = 0, \tag{4.2.4}$$

$$(\exp(\overline{W - \tilde{\Psi}})\tilde{W}', \phi_p') = 0, \tag{4.2.5}$$

for $1 \leq p \leq n$. We will demonstrate the convergence of the map $\mathcal{G} : (V, W) \longmapsto (\tilde{V}, \tilde{W})$ defined by (4.2.3)–(4.2.5), with the generation of $\tilde{\Psi}$ considered as a fractional step. This is done via the contraction mapping theorem in the set $B(\Lambda)$ defined by (3.4.19) in Chapter 3. We equip $B(\Lambda)$ with the norm

$$\|(V, W)\|_{B(\Lambda)} = \|V\|_{H^1(\Lambda)} + \|W\|_{H^1(\Lambda)}.$$

Firstly recall from Chapter 3, given any $(V, W) \in B(\Lambda)$ we can use (3.4.27), (3.4.28) to solve the discretised potential equation defined by (4.2.3) to obtain $\tilde{\mathbf{\Psi}} \in I\!R^n$ such that the piecewise linear function $\tilde{\Psi}$ associated with $\tilde{\mathbf{\Psi}}$ satisfies $\tilde{\Psi} \in E(\Lambda)$ where

$$E(\Lambda) := \left\{ \Psi \in S_h(\Lambda) : x^0 \leq \Psi \leq y^0 \right\}$$

and x^0, y^0 are defined by (3.4.25), (3.4.26) in Section 3.4. We now embark on constructing the appropriate contraction constant for the mapping $\mathcal{G}$. In the next two results, (V^i, W^i), $i = 1, 2$ will denote two arbitrary elements of $B(\Lambda)$. For each i, $\tilde{\Psi}^i$ will be the corresponding solutions of (4.2.3), and $(\tilde{V}^i, \tilde{W}^i)$ will be the corresponding solutions of (4.2.4), (4.2.5), with all solutions satisfying the appropriate Dirichlet boundary conditions.

LEMMA 4.2.1 *For each $M > 0$, there exists a constant C independent of h such that, for all (V^1, W^1), $(V^2, W^2) \in B(\Lambda)$,*

$$|\tilde{\Psi}^1 - \tilde{\Psi}^2|_{H^1(\Lambda)} \leq C\|(V^1, W^1) - (V^2, W^2)\|_{B(\Lambda)}$$

provided $\max\{|\alpha_0|, |\alpha_1|\} \leq M$.

Proof Given $(V^i, W^i) \in B(\Lambda)$, $i = 1, 2$, $\tilde{\Psi}^i$ satisfies (4.2.2) and

$$\lambda^2((\tilde{\Psi}^i)', \phi') + \langle \delta\{\exp(\tilde{\Psi}^i - V^i) - \exp(W^i - \tilde{\Psi}^i)\} - d, \phi \rangle = 0,$$

for all $\phi \in H_0^1(\Lambda)$. Subtracting the case $i = 1$ from $i = 2$ and putting $\phi = \tilde{\Psi}^2 - \tilde{\Psi}^1$ yields

$$\begin{aligned} \lambda^2((\tilde{\Psi}^2 - \tilde{\Psi}^1)', (\tilde{\Psi}^2 - \tilde{\Psi}^1)') + \delta\langle \exp(\tilde{\Psi}^2 - V^2) - \exp(W^2 - \tilde{\Psi}^2) \\ -\exp(\tilde{\Psi}^1 - V^1) + \exp(W^1 - \tilde{\Psi}^1), \tilde{\Psi}^2 - \tilde{\Psi}^1 \rangle \;=\; 0. \end{aligned}$$

We now write this as

$$\lambda^2((\tilde{\Psi}^2-\tilde{\Psi}^1)',(\tilde{\Psi}^2-\tilde{\Psi}^1)')+\delta\langle t_1,\tilde{\Psi}^2-\tilde{\Psi}^1\rangle \;=\; -\delta\langle t_2,\tilde{\Psi}^2-\tilde{\Psi}^1\rangle$$

where

$$t_1=(\exp(\tilde{\Psi}^2)-\exp(\tilde{\Psi}^1))\exp(-V^2)+(\exp(-\tilde{\Psi}^1)-\exp(-\tilde{\Psi}^2))\exp(W^2),$$

and

$$t_2=(\exp(-V^2)-\exp(-V^1))\exp(\tilde{\Psi}^1)+(\exp(W^1)-\exp(W^2))\exp(-\tilde{\Psi}^1).$$

Next note that, by the mean value theorem,

$$\langle t_1,\tilde{\Psi}^2-\tilde{\Psi}^1\rangle \;=\; \langle\exp(\eta-V^2)(\tilde{\Psi}^2-\tilde{\Psi}^1)+\exp(W^2-\mu)(\tilde{\Psi}^2-\tilde{\Psi}^1),\tilde{\Psi}^2-\tilde{\Psi}^1\rangle \;\geq 0$$

where η,μ are functions which lie between $\tilde{\Psi}^1$ and $\tilde{\Psi}^2$. Hence

$$\lambda^2((\tilde{\Psi}^2-\tilde{\Psi}^1)',(\tilde{\Psi}^2-\tilde{\Psi}^1)')\leq -\delta\langle t_2,\tilde{\Psi}^2-\tilde{\Psi}^1\rangle. \tag{4.2.6}$$

Also, by the definition of the quadrature rule (2.2.22),

$$\begin{aligned}|\langle t_2,\tilde{\Psi}^2-\tilde{\Psi}^1\rangle| \;&=\; \left|\sum_{i=1}^{n+1}\frac{h_i}{2}\left(t_2(x_i)(\tilde{\Psi}^1-\tilde{\Psi}^2)(x_i)+t_2(x_{i-1})(\tilde{\Psi}^1-\tilde{\Psi}^2)(x_{i-1})\right)\right|\\ &\leq\; \sum_{i=1}^{n+1}h_i\|t_2\|_{L_\infty(\Lambda)}\|\tilde{\Psi}^2-\tilde{\Psi}^1\|_{L_\infty(\Lambda)}\\ &\leq\; \|t_2\|_{H^1(\Lambda)}\|\tilde{\Psi}^2-\tilde{\Psi}^1\|_{H^1(\Lambda)}. \end{aligned}\tag{4.2.7}$$

Now since $(V^i,W^i)\in B(\Lambda)$, $\tilde{\Psi}^i\in E(\Lambda)$ and by the mean value theorem we have

$$\begin{aligned}&\|t_2\|_{H^1(\Lambda)}=\\ &\|(\exp(-V^2)-\exp(-V^1))\exp(\tilde{\Psi}^1)+(\exp(W^1)-\exp(W^2))\exp(-\tilde{\Psi}^1)\|_{H^1(\Lambda)}\\ &=\|\exp(\tilde{\Psi}^1-\eta)(V^1-V^2)+\exp(\mu-\tilde{\Psi}^1)(W^1-W^2)\|_{H^1(\Lambda)}\\ &\leq C(\|V^1-V^2\|_{H^1(\Lambda)}+\|W^1-W^2\|_{H^1(\Lambda)}),\end{aligned}\tag{4.2.8}$$

where η is a function lying between V^1 and V^2, and μ is a function lying between W^1 and W^2. Therefore by (4.2.7), (4.2.8) and Poincaré's inequality we have

$$|\langle t_2,\tilde{\Psi}^2-\tilde{\Psi}^1\rangle|\leq C\|(V^1,W^1)-(V^2,W^2)\|_{B(\Lambda)}|\tilde{\Psi}^2-\tilde{\Psi}^1|_{H^1(\Lambda)}.$$

So finally by (4.2.6),

$$|\tilde{\Psi}^2 - \tilde{\Psi}^1|^2_{H^1(\Lambda)} \leq C\|(V^1, W^1) - (V^2, W^2)\|_{B(\Lambda)}|\tilde{\Psi}^2 - \tilde{\Psi}^1|_{H^1(\Lambda)},$$

which gives the required result. ∎

We now use the result of Lemma 4.2.1 in the following theorem to obtain our contraction constant.

THEOREM 4.2.2 *The solutions* $\tilde{V}^i$, $\tilde{W}^i$, $i = 1, 2$ *satisfy*

$$(\tilde{V}^i, \tilde{W}^i) \in B(\Lambda).$$

Also, for each $M > 0$, *there exists a constant* C *independent of* h *such that*

$$\|(\tilde{V}^1, \tilde{W}^1) - (\tilde{V}^2, \tilde{W}^2)\|_{B(\Lambda)} \leq C \max\{|\alpha_1|, |\alpha_0|\}\|(V^1, W^1) - (V^2, W^2)\|_{B(\Lambda)},$$

for all $(V^1, W^1), (V^2, W^2) \in B(\Lambda)$, *provided* $\max\{|\alpha_0|, |\alpha_1|\} \leq M$.

Remark 4.2.1 Note that the contraction constant, $C \max\{|\alpha_1|, |\alpha_0|\}$, is independent of the mesh size, h. Hence if the mapping is a contraction for a particular mesh and boundary conditions then we are guaranteed that it will remain a contraction for all subsequent refinements of that mesh provided the boundary conditions are not altered. As we shall see in Chapter 5, this is in contrast to what we can prove in the two–dimensional case where the bound on the contraction constant is (weakly) mesh dependent.

Proof We first show that $\underline{\alpha} \leq \tilde{V}^i \leq \overline{\alpha}$, (the results for $\tilde{W}^i$ is analogous), so that $(\tilde{V}^i, \tilde{W}^i) \in B(\Lambda)$. We let $\overline{\alpha}$ denote the element of $S_h(\Lambda)$ which takes the value $\overline{\alpha}$ at every node of the mesh (2.2.7). Clearly $\overline{\alpha}' = 0$ so by (4.2.4) we have

$$(\exp(\overline{\tilde{\Psi}^i - V^i})(\tilde{V}^i - \overline{\alpha})', \phi_p') = 0 \tag{4.2.9}$$

for all interior nodes p. Now let $\boldsymbol{x} \in I\!R^n$ denote the vector of values of $\tilde{V}^i - \overline{\alpha}$ at interior nodes of our mesh and $\boldsymbol{x}_D$ denote the ordered pair $(\underline{\alpha} - \overline{\alpha}, 0)$ representing

the values of $\tilde{V}^i - \overline{\alpha}$ at the Dirichlet end–points. Then we see that (4.2.9) is in the form

$$[K \; K_D] \begin{pmatrix} \boldsymbol{x} \\ \boldsymbol{x}_D \end{pmatrix} = 0$$

where K is the stiffness matrix obtained from a piecewise linear finite element discretisation of (4.2.9) and K_D represents the coupling between the Dirichlet nodes and the interior nodes in our problem. It is easily verified that $K^{-1} > 0$ and K_D contains only non–positive entries, see for instance [71, page 85, Corollary 3]. Hence, since $\boldsymbol{x}_D \leq \mathbf{0}$, we have

$$\boldsymbol{x} = -K^{-1} K_D \boldsymbol{x}_D \leq \mathbf{0}.$$

Hence $\tilde{V}^i \leq \overline{\alpha}$. The proof that $\tilde{V}^i \geq \underline{\alpha}$ is analogous.

We now prove the bound stated in the theorem. Consider

$$\begin{aligned}
&(\exp(\overline{\tilde{\Psi}^1 - V^1})(\tilde{V}^2 - \tilde{V}^1)', (\tilde{V}^2 - \tilde{V}^1)') \\
&\quad = ((\exp(\overline{\tilde{\Psi}^1 - V^1}) - \exp(\overline{\tilde{\Psi}^2 - V^2}))(\tilde{V}^2)', (\tilde{V}^2 - \tilde{V}^1)') \\
&\qquad + (\exp(\overline{\tilde{\Psi}^2 - V^2})(\tilde{V}^2)', (\tilde{V}^2 - \tilde{V}^1)') - (\exp(\overline{\tilde{\Psi}^1 - V^1})(\tilde{V}^1)', (\tilde{V}^2 - \tilde{V}^1)').
\end{aligned}$$

By (4.2.4) the last two terms vanish since $\tilde{V}^2 - \tilde{V}^1 \in S_h^0(\Lambda)$. Hence, using the fact that $(V^1, W^1), (V^2, W^2) \in B(\Lambda)$ and $\tilde{\Psi}^1, \tilde{\Psi}^2 \in E(\Lambda)$, the Cauchy–Schwarz inequality gives us

$$C|\tilde{V}^2 - \tilde{V}^1|^2_{H^1(\Lambda)} \leq \|\exp(\overline{\tilde{\Psi}^1 - V^1}) - \exp(\overline{\tilde{\Psi}^2 - V^2})\|_{L_\infty(\Lambda)} |\tilde{V}^2|_{H^1(\Lambda)} |\tilde{V}^2 - \tilde{V}^1|_{H^1(\Lambda)},$$

and hence

$$|\tilde{V}^2 - \tilde{V}^1|_{H^1(\Lambda)} \leq C\|\exp(\overline{\tilde{\Psi}^1 - V^1}) - \exp(\overline{\tilde{\Psi}^2 - V^2})\|_{L_\infty(\Lambda)} |\tilde{V}^2|_{H^1(\Lambda)}. \quad (4.2.10)$$

We now bound each of the terms of the right hand side of (4.2.10) in turn. Firstly note that on each interval $I_i = [x_{i-1}, x_i]$, $i = 1, \ldots, n+1$

$$|\exp(\overline{\tilde{\Psi}^1 - V^1}) - \exp(\overline{\tilde{\Psi}^2 - V^2})|$$

$$
\begin{aligned}
&= h_i \left| \left\{ \int_{I_i} \exp(V^1 - \tilde{\Psi}^1) \right\}^{-1} - \left\{ \int_{I_i} \exp(V^2 - \tilde{\Psi}^2) \right\}^{-1} \right| \\
&= h_i \frac{|\int_{I_i} \{\exp(V^2 - \tilde{\Psi}^2) - \exp(V^1 - \tilde{\Psi}^1)\}|}{\{\int_{I_i} \exp(V^1 - \tilde{\Psi}^1)\}\{\int_{I_i} \exp(V^2 - \tilde{\Psi}^2)\}} \\
&\leq C h_i^{-1} \int_{I_i} |\exp(V^2 - \tilde{\Psi}^2) - \exp(V^1 - \tilde{\Psi}^1)| \\
&\leq C \| \exp(V^2 - \tilde{\Psi}^2) - \exp(V^1 - \tilde{\Psi}^1) \|_{L_\infty(I_i)}.
\end{aligned}
$$

Hence by the mean value theorem we have

$$
\| \exp(\overline{\tilde{\Psi}^1 - V^1}) - \exp(\overline{\tilde{\Psi}^2 - V^2}) \|_{L_\infty(\Lambda)} \leq C \{ \| \tilde{\Psi}^1 - \tilde{\Psi}^2 \|_{L_\infty(\Lambda)} + \| V^1 - V^2 \|_{L_\infty(\Lambda)} \}.
$$

Thus, using the fundamental theorem of calculus and Lemma 4.2.1, we obtain the inequality

$$
\| \exp(\overline{\tilde{\Psi}^1 - V^1}) - \exp(\overline{\tilde{\Psi}^2 - V^2}) \|_{L_\infty(\Lambda)} \leq C \| (V^1, W^1) - (V^2, W^2) \|_{B(\Lambda)}. \quad (4.2.11)
$$

This bounds the first term on the right hand side of (4.2.10). Considering the second term, recall that $\tilde{V}^2$ is defined by (4.2.4) with $V = V^2$ and $\tilde{\Psi} = \tilde{\Psi}^2$. For $i = 0, 1$, define u_i to be the solution of the weak problem: Find $u_i \in H^1(\Lambda)$ with $u_i(j) = \delta_{ij}$ for $j = 0, 1$ and such that

$$
(\exp(\overline{\tilde{\Psi}^2 - V^2}) u_i', \phi') = 0, \quad \text{for all } \phi \in H^1(\Lambda), \quad \phi = 0 \ \text{ at } \ x = 0, 1.
$$

Let $U_i \in S_h(\Lambda)$ be the usual finite element approximation of u_i. Then by standard theory, these finite element problems are well-posed and the solution is stable in the energy norm, i.e. $|U_i|_{H^1(\Lambda)} \leq C$, with C independent of h. Then by uniqueness and linearity, $\tilde{V}^2 = \sum_{i=0}^{1} \alpha_i U_i$, and hence we obtain the bound for the second term of (4.2.10):

$$
|\tilde{V}^2|_{H^1(\Lambda)} \leq \sum_i |\alpha_i| |U_i|_{H^1(\Lambda)} \leq C \max\{|\alpha_0|, |\alpha_1|\}, \quad (4.2.12)
$$

with C independent of h and α_i for each i . Combining (4.2.11) and (4.2.12) in (4.2.10) gives the required result. ■

Hence we have shown, using a conservative scheme, that Gummel's method for a one–dimensional device converges provided the applied voltage is sufficiently small. However, in practice, we have found decoupling strategies that are observed to converge even for large applied voltages. In an attempt to understand this behaviour we now present some arguments for a model diode problem. The discussion is not entirely rigorous but will hopefully give the reader a feel for why some of these schemes behave so well in apparently extreme conditions. For convenience we will consider only the standard finite element discretisation (2.2.19), (2.2.20), (2.2.21).

4.3 p–n diode with large applied voltage

We now consider our one–dimensional model for a simple p–n diode under reverse bias conditions. We have found in practice that a certain variation of Gummel's method (to be defined below) converges for this configuration even for large applied voltages. To explain this convergence, we first need to introduce the concept of *machine precision.* This is defined to be the smallest positive number, ϵ_{mp} with the property that the logical expression,

$$1.0 < 1.0 + \epsilon_{mp},$$

is evaluated as *true* on the machine in question. We shall regard single machine precision to be

$$\epsilon_{mp} = 1.49 \times 10^{-8},$$

which is the square root of the double precision available in MATLAB on a Sun4.

We discretise the the equations on a uniform mesh containing n points interior to Λ and hence with interval length $h = 1/(n+1)$. A simple one–dimensional p–n diode consists of a negatively doped interval coupled to a positively doped interval. These are the n– and the p–regions respectively. There is an extremely narrow interface between the two regions which we approximate by a point. We

shall label this point ν. We assume that the point ν occurs at the mesh point x_N, where $2 \leq N \leq n-1$. Again we assume that there is zero recombination. Discretisation by the standard finite element method leads to the problem: Seek $(\Psi, V, W) \in S_h(\Lambda)^3$ satisfying

$$\lambda^2(\Psi', \phi_p') + \langle \delta\{\exp(\Psi - V) - \exp(W - \Psi)\} - d, \phi_p \rangle = 0, \tag{4.3.13}$$

$$(\exp(\Psi - V)V', \phi_p') = 0, \tag{4.3.14}$$

$$(\exp(W - \Psi)W', \phi_p') = 0, \tag{4.3.15}$$

for $1 \leq p \leq n$.

For simplicity, in this section we assume that the diode has equal doping of electrons and holes either side of an interface which occurs at $x = \nu$. It follows that the doping profile, $d = -1$ for $x \in [0, \nu)$, $d = +1$ for $x \in (\nu, 1]$ and $d(\nu) = 0$. Furthermore we assume zero applied voltage at the left hand end of the device. This leads to the boundary conditions

$$V(0) = 0 = W(0), \quad V(1) = \alpha = W(1), \tag{4.3.16}$$

$$\Psi(0) = -\beta, \quad \Psi(1) = \beta + \alpha, \tag{4.3.17}$$

with $\alpha > 0$ and where

$$\beta = \sinh^{-1}(1/2\delta). \tag{4.3.18}$$

By "large applied voltage", we mean values of α such that

$$2\beta < \alpha < \exp(2\beta) \times (\epsilon_{mp})/2 \tag{4.3.19}$$

If we take the parameters given in [62] which are for a silicon device of length 10^{-3}cm at room temperature then $\lambda^2 = 1.68 \times 10^{-7}$, $\delta = 1.22 \times 10^{-8}$ and $\beta = 18.22$. With this value of β and ϵ_{mp} as given above, we see that (4.3.19) is roughly the range,

$$37 < \alpha < 5 \times 10^7.$$

This corresponds to an applied voltage between 1V and 1.3×10^6V. The upper bound is therefore not restrictive as the device would melt long before the applied voltage reached 10^6V.

Now we write $\Psi = \sum_i \Psi_i \phi_i$, $V = \sum_i V_i \phi_i$ and $W = \sum_i W_i \phi_i$ and recall the MATLAB style notation introduced in Chapter 2.

If Θ is any piecewise linear function with respect to our uniform grid, define

$$k_i(\Theta) = \frac{1}{h^2} \int_{x_{i-1}}^{x_i} \exp(\Theta), \quad i = 1, \ldots, n+1. \tag{4.3.20}$$

Then, let $\tilde{K}(\Theta)$ denote the $n \times (n+2)$ matrix given for $i = 1, \ldots, n$ by:

$$\left.\begin{aligned} \tilde{K}(\Theta)_{i,i} &= k_i(\Theta) + k_{i+1}(\Theta), \\ \tilde{K}(\Theta)_{i,i-1} &= -k_i(\Theta), \\ \tilde{K}(\Theta)_{i,i+1} &= -k_{i+1}(\Theta), \\ \tilde{K}(\Theta)_{ij} &= 0, \text{ for all other } i, j. \end{aligned}\right\} \tag{4.3.21}$$

Let $K(\Theta)$ be the $n \times n$ symmetric tridiagonal matrix with elements

$$K(\Theta)_{ij} = \tilde{K}(\Theta)_{ij}, \quad i, j = 1, \ldots, n.$$

Clearly $\tilde{K}(\Theta)$ is the matrix introduced in Section 2.2, but here we have a fixed mesh size h. Using this notation (4.3.16)–(4.3.22) can be written as: Find $\boldsymbol{\Psi}, \boldsymbol{V}, \boldsymbol{W} \in I\!R^n$ such that

$$\lambda^2 \tilde{K}(0)[-\beta; \boldsymbol{\Psi}; \beta + \alpha] + h[\delta(\exp(\boldsymbol{\Psi} - \boldsymbol{V}) - \exp(\boldsymbol{W} - \boldsymbol{\Psi})) - \boldsymbol{d}] = \boldsymbol{0}, \tag{4.3.22}$$

$$\tilde{K}(\Psi - V)[0; \boldsymbol{V}; \alpha] = \boldsymbol{0} \tag{4.3.23}$$

$$\tilde{K}(W - \Psi)[0; \boldsymbol{W}; \alpha] = \boldsymbol{0} \tag{4.3.24}$$

where

$$d_i = d(x_i) = \begin{cases} -1 & i = 1, \ldots, N-1 \\ 0 & i = N \\ 1 & i = N+1, \ldots, n \end{cases} \tag{4.3.25}$$

Then (4.3.22)-(4.3.24) form a coupled nonlinear system in $(\mathbb{R}^n)^3$ which we rewrite collectively as

$$\boldsymbol{F}(\boldsymbol{\Psi}, \boldsymbol{V}, \boldsymbol{W}) = \mathbf{0}, \tag{4.3.26}$$

where

$$\boldsymbol{F} = (\boldsymbol{F}_1, \boldsymbol{F}_2, \boldsymbol{F}_3)^T,$$

and $\boldsymbol{F}_1$, $\boldsymbol{F}_2$, and $\boldsymbol{F}_3$ are the left hand sides of (4.3.22) (4.3.23), and (4.3.24) respectively.

We now consider the following decoupling strategy for (4.3.22)–(4.3.24) which is a variation on Gummel's method. In practice, this strategy is observed to converge even for large applied voltages. We shall explain this convergence below.

- *Step 1* Define the starting vectors $\boldsymbol{V}^0, \boldsymbol{W}^0 \in \mathbb{R}^n$, by

$$V_i^0 = \begin{cases} 0, & i = 1, \ldots, N-1 \\ \alpha, & i = N, \ldots, n \end{cases} \tag{4.3.27}$$

$$W_i^0 = \begin{cases} 0, & i = 1, \ldots, N \\ \alpha, & i = N+1, \ldots, n \end{cases} \tag{4.3.28}$$

- *Step 2* With $\boldsymbol{V}^0$, $\boldsymbol{W}^0$ fixed, (4.3.22) is a nonlinear system

$$\boldsymbol{F}_1(\boldsymbol{\Psi}, \boldsymbol{V}^0, \boldsymbol{W}^0) = \mathbf{0} \tag{4.3.29}$$

  for unknown $\boldsymbol{\Psi} \in \mathbb{R}^n$. Construct lower and upper solutions, $\boldsymbol{X}^0, \boldsymbol{Y}^0$, and then use the quasi–Newton method introduced in Chapter 3 to iterate to the unique solution, $\boldsymbol{\Psi}^*$, of (4.3.29).

- *Step 3* With $\boldsymbol{\Psi} = \boldsymbol{\Psi}^*$ and using starting values $\boldsymbol{V} = \boldsymbol{V}^0$, $\boldsymbol{W} = \boldsymbol{W}^0$, perform *Gummel's iteration* on (4.3.23), (4.3.24), i.e., for $k \geq 0$, iterate:

$$\dot{\tilde{K}}(\Psi^* - V^k)[0; \boldsymbol{V}^{k+1}; \alpha] = \mathbf{0}, \tag{4.3.30}$$

$$\tilde{K}(W^k - \Psi^*)[0; \boldsymbol{W}^{k+1}; \alpha] = \mathbf{0}, \tag{4.3.31}$$

This iteration (which can be performed as two parallel steps) preserves certain classes of functions for all $\alpha > 0$. (Lemmas 4.3.3 and 4.3.4) Using empirical arguments we show that it is convergent for large α. Denote the limit functions by $(\boldsymbol{V}^*, \boldsymbol{W}^*)$

- *Step 4* $\boldsymbol{\Psi}^*, \boldsymbol{V}^*, \boldsymbol{W}^*$ are now close to the true solutions in the sense that the residual $\boldsymbol{F}(\boldsymbol{\Psi}^*, \boldsymbol{V}^*, \boldsymbol{W}^*)$ is zero to within (single) machine precision.

We now proceed to give arguments (some rather heuristic in their nature) which explain why this scheme is successful. We start by producing tight bounds on the solution, $\boldsymbol{\Psi}^*$, of (4.3.29). This is done by constructing lower and upper solutions, $\boldsymbol{X}^0, \boldsymbol{Y}^0$, to (4.3.29). The results in Chapter 3 then tell us that by using the quasi–Newton method described therein we will obtain a solution $\boldsymbol{\Psi}^*$ which lies between $\boldsymbol{X}^0$ and $\boldsymbol{Y}^0$.

In order to produce good bounds on $\boldsymbol{\Psi}^*$, let us introduce the "characteristic equation"

$$\frac{\lambda^2}{2h}(\beta - \epsilon) + h(2\delta\sinh(\beta - \epsilon) - 1) + \frac{\lambda^2}{2h}(\alpha + \beta) = 0, \tag{4.3.32}$$

which is to be solved for ϵ. It is useful to first consider some properties of the solution ϵ. We do this in the following lemma

LEMMA 4.3.1 *The solution ϵ of (4.3.32) satisfies*

$$0 < \epsilon < 2\beta + \alpha$$

Proof Clearly $0 < \epsilon < 2\beta + \alpha$ if and only if

$$\beta > \beta - \epsilon > -\beta - \alpha. \tag{4.3.33}$$

We first prove the left hand inequality in (4.3.33). Assume for a contradiction that $\beta \leq \beta - \epsilon$, then by (4.3.32) and the definition of β, (4.3.18),

$$\begin{aligned} 0 &\geq \frac{\lambda^2}{2h}\beta + h(2\delta\sinh(\beta) - 1) + \frac{\lambda^2}{2h}(\alpha + \beta) \\ &= \frac{\lambda^2}{2h}(\alpha + 2\beta) > 0, \end{aligned}$$

which is a contradiction. Hence $\beta > \beta - \epsilon$.

Now for the right hand inequality in (4.3.33). Suppose that $\beta - \epsilon \leq -\beta - \alpha$. Then (4.3.32) again gives

$$\begin{aligned} 0 &\leq -\frac{\lambda^2}{2h}(\beta+\alpha) + h(2\delta\sinh(-\beta-\alpha)-1) + \frac{\lambda^2}{2h}(\alpha+\beta) \\ &= -h(2\delta\sinh(\alpha+\beta)+1) < 0, \end{aligned}$$

and since this is once again a contradiction we have $\beta-\epsilon > -\beta-\alpha$. ■

Considering our model problem with statistics given by [62], we see that ϵ is in fact very small. In Table 4.1 we give some computed values of ϵ for suitable large values of α. These results were obtained using a MATLAB code which performs Newton's method on (4.3.32). The values of α correspond to physical applied voltages of 5V and 20V respectively.

	ϵ	
h	$\alpha = 193.4$	$\alpha = 773.6$
0.050	7.71×10^{-3}	2.75×10^{-2}
0.033	1.74×10^{-2}	6.29×10^{-2}
0.025	3.12×10^{-2}	1.12×10^{-1}
0.020	4.92×10^{-2}	1.85×10^{-1}

Table 4.1: Typical values of ϵ

With ϵ as in Lemma 4.3.1, consider the following vectors which we shall show to be lower and upper solutions for (4.3.29). Let

$$X_i^0 = \begin{cases} -\beta & i = 1,\ldots,N-1, \\ (\alpha-\epsilon)/2 & i = N, \\ \beta+\alpha-\epsilon & i = N+1,\ldots,n. \end{cases} \tag{4.3.34}$$

$$Y_i^0 = \begin{cases} -\beta+\epsilon & i = 1,\ldots,N-1, \\ (\alpha+\epsilon)/2 & i = N, \\ \beta+\alpha, & i = N+1,\ldots,n. \end{cases} \tag{4.3.35}$$

Note that these are much more sophisticated upper and lower solutions than those proposed in Chapter 3, and demonstrate the sudden jump in Ψ^* in the vicinity of x_N.

LEMMA 4.3.2 *The vectors $\boldsymbol{X}^0$ and $\boldsymbol{Y}^0$ defined by (4.3.34) and (4.3.35) are lower and upper solutions for (4.3.29) with $\boldsymbol{V}^0$ and $\boldsymbol{W}^0$ defined by (4.3.27) and (4.3.28) respectively.*

Proof In this proof we shall freely use the inequality proved in Lemma 4.3.1. We first show that $\boldsymbol{X}^0$ is a lower solution. For $i = 1, \ldots, N-2$,

$$(\boldsymbol{F}_1(\boldsymbol{X}^0, \boldsymbol{V}^0, \boldsymbol{W}^0))_i = h[2\delta\sinh(-\beta) + 1] = 0.$$

For $i = N+2, \ldots, n$,

$$(\boldsymbol{F}_1(\boldsymbol{X}^0, \boldsymbol{V}^0, \boldsymbol{W}^0))_i = h[2\delta\sinh(\beta - \epsilon) - 1] \leq h[2\delta\sinh(\beta) - 1] = 0.$$

Also

$$\begin{aligned}(\boldsymbol{F}_1(\boldsymbol{X}^0, \boldsymbol{V}^0, \boldsymbol{W}^0))_{N-1} &= \frac{\lambda^2}{h}\left(-\beta - \frac{\alpha - \epsilon}{2}\right) + h[\delta(\exp(-\beta) - \exp(\beta)) + 1] \\ &= \frac{\lambda^2}{2h}(\epsilon - (2\beta + \alpha)) < 0.\end{aligned}$$

Moreover

$$\begin{aligned}(\boldsymbol{F}_1(\boldsymbol{X}^0, \boldsymbol{V}^0, \boldsymbol{W}^0))_N &= h[\delta(\exp((\alpha - \epsilon)/2 - \alpha) - \exp(-(\alpha - \epsilon)/2)] \\ &= h[\delta(\exp((-\alpha - \epsilon)/2) - \exp((-\alpha + \epsilon)/2)] < 0.\end{aligned}$$

Finally

$$\begin{aligned}&(\boldsymbol{F}_1(\boldsymbol{X}^0, \boldsymbol{V}^0, \boldsymbol{W}^0))_{N+1} \\ = \ &\frac{\lambda^2}{h}\left(\beta + \alpha - \epsilon - \left(\frac{\alpha - \epsilon}{2}\right)\right) + h[\delta(\exp(\beta - \epsilon) - \exp(-(\beta - \epsilon)) - 1] \\ = \ &\frac{\lambda^2}{2h}(\beta - \epsilon) + h[2\delta\sinh(\beta - \epsilon) - 1] + \frac{\lambda^2}{2h}(\beta + \alpha) = 0,\end{aligned}$$

by (4.3.32). Hence $\boldsymbol{X}^0$ is a lower solution. The proof that $\boldsymbol{Y}^0$ is an upper solution is very similar. ■

Hence by Theorem 3.4.1, the resulting $\boldsymbol{\Psi}^*$ obtained from Step 2 of our iteration satisfies

$$\boldsymbol{X}^0 \leq \boldsymbol{\Psi}^* \leq \boldsymbol{Y}^0. \tag{4.3.36}$$

We now consider Step 3 of our iteration. By our earlier comments we know that ϵ is small and so $\boldsymbol{X}^0$ and $\boldsymbol{Y}^0$ are almost equal. Hence to simplify the analysis of Step 3 (and Step 4) we shall replace the true solution, $\boldsymbol{\Psi}^*$, by the approximation $\boldsymbol{\Psi}$ defined by

$$\Psi_i = \begin{cases} -\beta, & i = 1, \ldots, N-1 \\ \alpha/2, & i = N, \\ \beta + \alpha, & i = N+1, \ldots, n. \end{cases} \tag{4.3.37}$$

Note that by (4.3.34), (4.3.35), $\boldsymbol{X}^0$, $\boldsymbol{Y}^0$ almost coincide with $\boldsymbol{\Psi}^*$ when ϵ is small. To analyse the iteration (4.3.30), (4.3.31) let us introduce the following sets:

$$K = \left\{ \boldsymbol{V} \in I\!R^{n+2} : 0 = V_0 \leq V_1 \leq \ldots \leq V_n \leq V_{n+1} = \alpha \right\}. \tag{4.3.38}$$

$$K_\alpha^V = \{ \boldsymbol{V} \in K : V_i = \alpha \text{ to machine precision, } i = N, \ldots, n \}. \tag{4.3.39}$$

$$K_\alpha^W = \{ \boldsymbol{W} \in K : W_i = 0 \text{ to machine precision, } i = 1, \ldots, N \}. \tag{4.3.40}$$

Also, define

$$\partial V_i = V_i - V_{i-1}. \tag{4.3.41}$$

We now consider solving (4.3.30) and (4.3.31) with $\boldsymbol{\Psi}^*$ replaced by $\boldsymbol{\Psi}$ given in (4.3.37). Using the notation defined by (4.3.20), we may write (4.3.30) as

$$k_i(\Psi - V^k)\partial V_i^{k+1} - k_{i+1}(\Psi - V^k)\partial V_{i+1}^{k+1} = 0, \quad i = 1, \ldots, n,$$

where Ψ is the piecewise linear interpolant of $\boldsymbol{\Psi}$. Alternatively we can write

$$\frac{\partial V_{i+1}^{k+1}}{\partial V_i^{k+1}} = \frac{k_i(\Psi - V^k)}{k_{i+1}(\Psi - V^k)} := q_i^V, \quad i = 1, \ldots, n. \tag{4.3.42}$$

Similarly (4.3.31) can be rewritten as

$$\frac{\partial W_{i+1}^{k+1}}{\partial W_i^{k+1}} = \frac{k_i(W^k - \Psi)}{k_{i+1}(W^k - \Psi)} := q_i^W, \quad i = 1, \ldots, n. \tag{4.3.43}$$

These reformulations of (4.3.30) and (4.3.31) help in the following lemma.

LEMMA 4.3.3 *The mappings $\boldsymbol{V}^k \to \boldsymbol{V}^{k+1}$ and $\boldsymbol{W}^k \to \boldsymbol{W}^{k+1}$ are invariant on K.*

Proof We give the proof for $\boldsymbol{V}^k \to \boldsymbol{V}^{k+1}$. The other part is similar. Clearly by (4.3.20), $k_i > 0$ for all i. Hence $q_i^V > 0$ for each i. Therefore (4.3.42) implies that $\left\{\partial V_i^{k+1},\ i = 1, \ldots, n+1\right\}$ are all of the same sign. Since $V_0^{k+1} = 0$ and $V_{n+1}^{k+1} = \alpha > 0$ we conclude that $\partial V_i^{k+1} > 0$ for all $i = 1, \ldots, n+1$. Hence $\boldsymbol{V}^{k+1} \in K$. ■

By taking account of the effect of machine precision we can say more:

LEMMA 4.3.4 *The computer implementation of the mapping $\boldsymbol{V}^k \to \boldsymbol{V}^{k+1}$ leaves K_α^V invariant. Similarly the computer implementation of the mapping $\boldsymbol{W}^k \to \boldsymbol{W}^{k+1}$ leaves K_α^W invariant.*

Proof Let $\boldsymbol{V}^k \in K_\alpha^V$. We will show that $\boldsymbol{V}^{k+1} \in K_\alpha^V$. The other part is analogous. Recall that (4.3.30) can be written in the form (4.3.42) and for any piecewise linear function Θ consider

$$\begin{aligned} k_i(\Theta) &= \frac{1}{h^2}\int_{x_{i-1}}^{x_i} \exp(\Theta)dx = \frac{1}{h}\left[\frac{\exp(\Theta_i) - \exp(\Theta_{i-1})}{\Theta_i - \Theta_{i-1}}\right] \\ &= \frac{\exp(\Theta_i)}{h}\left[\frac{1 - \exp(\Theta_{i-1} - \Theta_i)}{\Theta_i - \Theta_{i-1}}\right] = \frac{\exp(\Theta_i)}{h}\varphi(-\partial\Theta_i), \end{aligned}$$

where

$$\varphi(x) := \frac{\exp(x) - 1}{x}. \tag{4.3.44}$$

The properties of φ are discussed in the Appendix A1. Similarly

$$\begin{aligned} k_{i+1}(\Theta) &= \frac{1}{h^2}\int_{x_i}^{x_{i+1}} \exp(\Theta)dx = \frac{1}{h}\left[\frac{\exp(\Theta_{i+1})-\exp(\Theta_i)}{\Theta_{i+1}-\Theta_i}\right] \\ &= \frac{\exp(\Theta_i)}{h}\left[\frac{\exp(\Theta_{i+1}-\Theta_i)-1}{\Theta_{i+1}-\Theta_i}\right] = \frac{\exp(\Theta_i)}{h}\varphi(\partial\Theta_{i+1}). \end{aligned}$$

Hence, for $i = 1, \ldots, n$, we have from (4.3.42)

$$q_i^V = \frac{\varphi(-\partial\Theta_i)}{\varphi(\partial\Theta_{i+1})}. \tag{4.3.45}$$

Set $\Theta = \Psi - V^k$. Then using the fact that $\boldsymbol{V}^k \in K_\alpha^V$ and the definition (4.3.37) of $\boldsymbol{\Psi}$, we have

$$\begin{aligned} \partial\Theta_i &= -\partial V_i^k, \quad i = 1, \ldots, N-1, \\ \partial\Theta_N &= \beta + V_{N-1}^k - \alpha/2, \\ \partial\Theta_{N+1} &= \beta + \alpha/2, \\ \partial\Theta_i &= 0, \qquad i = N+2, \ldots, n+1. \end{aligned}$$

Hence substituting in (4.3.45) we have

$$\begin{gathered} q_i^V = \varphi(\partial V_i^k)/\varphi(-\partial V_{i+1}^k), \quad i = 1, \ldots, N-2, \\ q_{N-1}^V = \varphi(\partial V_{N-1}^k)/\varphi(\beta + V_{N-1}^k - \alpha/2), \\ q_N^V = \varphi(-\beta - V_{N-1}^k + \alpha/2)/\varphi(\beta + \alpha/2), \\ q_{N+1}^V = \varphi(-\beta - \alpha/2), \\ q_i^V = 1, \quad i = N+2, \ldots, n. \end{gathered}$$

From this we know that

$$\partial V_i^{k+1} = \partial V_{N+2}^{k+1}, \quad \text{for } i = N+3, \ldots, n+1. \tag{4.3.46}$$

Also, since $\varphi(x) < \exp(x)$ for any x and by the monotonicity of φ proved in Lemma A.2.1, we have

$$q_N^V \le \frac{\varphi(-\beta+\alpha/2)}{\varphi(\beta+\alpha/2)} < \exp(-\beta+\alpha/2)\left(\frac{\beta+\alpha/2}{\exp(\beta+\alpha/2)-1}\right)$$

$$
\begin{aligned}
&= \exp(-\beta+\alpha/2)\exp(-\beta-\alpha/2)\left(\frac{\beta+\alpha/2}{1-\exp(-\beta-\alpha/2)}\right) \\
&= (\beta+\alpha/2)\exp(-2\beta)(1-\exp(-2\beta))^{-1} \\
&< 2\exp(-2\beta)(\beta+\alpha/2),
\end{aligned}
$$

since $\alpha > 2\beta$ and $\beta = 18.22$. Also

$$
q^V_{N+1} = \varphi(-\beta-\alpha/2) = \frac{1-\exp(\beta+\alpha/2)}{\beta+\alpha/2} < \frac{1}{\beta+\alpha/2}. \tag{4.3.47}
$$

Hence for $i = N+3,\ldots,n+1$, we have by (4.3.42), (4.3.46), (4.3.47) and Lemma 4.3.3,

$$
\begin{aligned}
\partial V_i^{k+1} = \partial V_{N+2}^{k+1} &< \frac{1}{\beta+\alpha/2}\partial V_{N+1}^{k+1} \\
&< \frac{1}{\beta+\alpha/2}2\exp(-2\beta)(\beta+\alpha/2)\partial V_N^{k+1} \\
&< 2\alpha\exp(-2\beta) = 0 \ \text{ to machine precision.}
\end{aligned}
$$

It follows that $V_i^{k+1} = \alpha$ to machine precision for $i = N,\ldots,n$ and hence $\boldsymbol{V}^{k+1} \in K_\alpha^V$ as required. ■

Clearly $\boldsymbol{V}^0, \boldsymbol{W}^0$ defined by (4.3.27), (4.3.28) satisfy $\boldsymbol{V}^0 \in K_\alpha^V$, $\boldsymbol{W}^0 \in K_\alpha^W$. Therefore Lemma 4.3.4 and mathematical induction show that the sequence generated by (4.3.30), starting with $\boldsymbol{V}^0$ defined by (4.3.27), satisfies $\left\{\boldsymbol{V}^k\right\}_{k=0}^\infty \subset K_\alpha^V$. Similarly $\left\{\boldsymbol{W}^k\right\}_{k=0}^\infty \subset K_\alpha^W$. We now make the *assumption* that the sequences $\{\boldsymbol{V}^k\}_{k=0}^\infty$, $\{\boldsymbol{W}^k\}_{k=0}^\infty$ so produced converge to some limits which we denote $\boldsymbol{V}^* \in K_\alpha^V$, $\boldsymbol{W}^* \in K_\alpha^W$.

Finally we consider the residual produced when we evaluate $\boldsymbol{F}_1(\boldsymbol{\Psi}, \boldsymbol{V}^*, \boldsymbol{W}^*)$. In Theorem 4.3.5 we will see that $\boldsymbol{F}_1$ evaluated with our approximate solution $\boldsymbol{\Psi}$ and the actual solutions $\boldsymbol{V}^*$, $\boldsymbol{W}^*$ is effectively the same as $\boldsymbol{F}_1(\boldsymbol{\Psi}, \boldsymbol{V}^0, \boldsymbol{W}^0)$.

THEOREM 4.3.5 $\boldsymbol{F}_1(\boldsymbol{\Psi}, \boldsymbol{V}^*, \boldsymbol{W}^*) = \boldsymbol{F}_1(\boldsymbol{\Psi}, \boldsymbol{V}^0, \boldsymbol{W}^0)$ *to (single) machine precision.*

Proof For ease of notation we will write $\simeq$ to mean equality to machine precision. Firstly note that

$$\boldsymbol{F}_1(\boldsymbol{\Psi}, \boldsymbol{V}^*, \boldsymbol{W}^*) - \boldsymbol{F}_1(\boldsymbol{\Psi}, \boldsymbol{V}^0, \boldsymbol{W}^0) =$$
$$h[\delta(\exp(\boldsymbol{\Psi} - \boldsymbol{V}^*) - \exp(\boldsymbol{W}^* - \boldsymbol{\Psi})] - h[\delta(\exp(\boldsymbol{\Psi} - \boldsymbol{V}^0) - \exp(\boldsymbol{W}^0 - \boldsymbol{\Psi})].$$

Now observe that for $1 \leq i \leq N-1$,

$$\exp(\boldsymbol{\Psi} - \boldsymbol{V}^*) \leq \exp(-\beta) = 1.22 \times 10^{-8} < \epsilon_{mp},$$

$$\exp(\boldsymbol{\Psi} - \boldsymbol{V}^0) = \exp(-\beta) = 1.22 \times 10^{-8} < \epsilon_{mp}.$$

Hence

$$\begin{aligned}(\exp(\boldsymbol{\Psi} - \boldsymbol{V}^*) - \exp(\boldsymbol{W}^* - \boldsymbol{\Psi}))_i &\simeq -(\exp(\boldsymbol{W}^* - \boldsymbol{\Psi}))_i \\ &= -(\exp(\boldsymbol{W}^0 - \boldsymbol{\Psi}))_i \\ &\simeq (\exp(\boldsymbol{\Psi} - \boldsymbol{V}^0) - \exp(\boldsymbol{W}^0 - \boldsymbol{\Psi}^*))_i,\end{aligned}$$

where the equality in the second line comes from Lemma 4.3.4. Also for $N+1 \leq i \leq n$ we have

$$\exp(\boldsymbol{W}^* - \boldsymbol{\Psi}) \leq \exp(-\beta) = 1.22 \times 10^{-8} < \epsilon_{mp},$$

$$\exp(\boldsymbol{W}^0 - \boldsymbol{\Psi}) = \exp(-\beta) = 1.22 \times 10^{-8} < \epsilon_{mp}.$$

Therefore

$$\begin{aligned}(\exp(\boldsymbol{\Psi} - \boldsymbol{V}^*) - \exp(\boldsymbol{W}^* - \boldsymbol{\Psi}))_i &\simeq (\exp(\boldsymbol{\Psi} - \boldsymbol{V}^*))_i \\ &= (\exp(\boldsymbol{\Psi} - \boldsymbol{V}^0))_i \\ &\simeq (\exp(\boldsymbol{\Psi} - \boldsymbol{V}^0) - \exp(\boldsymbol{W}^0 - \boldsymbol{\Psi}))_i.\end{aligned}$$

Finally

$$\begin{aligned}(\exp(\boldsymbol{\Psi} - \boldsymbol{V}^*) - \exp(\boldsymbol{W}^* - \boldsymbol{\Psi})_N &= (\exp(\boldsymbol{\Psi} - \alpha) - \exp(-\boldsymbol{\Psi}))_N \\ &= (\exp(\boldsymbol{\Psi} - \boldsymbol{V}^0) - \exp(\boldsymbol{W}^0 - \boldsymbol{\Psi})_N.\end{aligned}$$

Hence

$$\boldsymbol{F}_1(\boldsymbol{\Psi}, \boldsymbol{V}^*, \boldsymbol{W}^*) \simeq \boldsymbol{F}_1(\boldsymbol{\Psi}, \boldsymbol{V}^0, \boldsymbol{W}^0),$$

as required. ■

Remark 4.3.1 It remains to point out that in the majority of cases, $\boldsymbol{\Psi}^*$ is identical to $\boldsymbol{\Psi}$ at most of the nodes of the grid and only differs slightly at the rest. Hence, in view of the fact that $\boldsymbol{F}_1(\boldsymbol{\Psi}^*, \boldsymbol{V}^0, \boldsymbol{W}^0) = \mathbf{0}$ and by the result of the previous lemma, the residual $\boldsymbol{F}_1(\boldsymbol{\Psi}^*, \boldsymbol{V}^*, \boldsymbol{W}^*)$ is zero to machine precision for the types of mesh and applied voltage that we have discussed in this section. Hence, as is often the case in practice, if a computer were to use this residual as a stopping criterion then it would accept $\boldsymbol{\Psi}^*$, $\boldsymbol{V}^*$, $\boldsymbol{W}^*$ as the true solutions.

As we have stressed early, this is by no means a rigorous argument. A complete analysis of this algorithm would , we feel, be a much harder undertaking. However the above results give the reader a feel for why such a process works so well in practice. We now continue to present some more detailed results on the shape and structure of the solution $\boldsymbol{\Psi}^*$ to (4.3.29).

4.4 Further shape results for the potential

We now give qualitative and quantitative results for the solution, $\boldsymbol{\Psi}^*$, of the discretised potential equation with $\boldsymbol{V}, \boldsymbol{W}$ set to our initial guesses $\boldsymbol{V}^0, \boldsymbol{W}^0$ defined by (4.3.27), (4.3.28). We provide a qualitative description of the shape of $\boldsymbol{\Psi}^*$ and a quantitative estimate of the width of the internal layer in $\boldsymbol{\Psi}^*$ at $x = x_N = \mu$. These are refinements of the estimates established in Lemma 4.3.2 and provide numerical versions of the singular perturbation results in, for example, [9], [61], describing the behaviour of ψ in the undiscretised system (4.2.1)–(4.2.3). The results of this section will also further justify the approximation (4.3.37), made in Section 4.3.

We no longer restrict ourselves to a uniform grid, returning instead to the

system discretised by the finite element method with respect to the mesh (2.2.7). Again mass lumping is employed in handling the nonlinear term. Hence we wish to find $\boldsymbol{\Psi} \in I\!R^n$ such that

$$\lambda^2 \tilde{K}(0)[-\beta; \boldsymbol{\Psi}; \beta + \alpha] + \mathrm{diag}\{\overline{\boldsymbol{h}}\}[\delta(\exp(\boldsymbol{\Psi} - \boldsymbol{V}^0) - \exp(\boldsymbol{W}^0 - \boldsymbol{\Psi})) - \boldsymbol{d}] = \boldsymbol{0}, \tag{4.4.48}$$

where $\tilde{K}(.), \boldsymbol{d}$ are defined by (4.3.21), (4.3.25), but now

$$k_i(0) = \frac{1}{h_i}, \quad i = 1, \ldots, n+1,$$

and $\overline{\boldsymbol{h}} \in I\!R^n$ is the vector with components defined by

$$\overline{h}_i = (h_i + h_{i+1})/2, \quad i = 1, \ldots, n.$$

Our first result describes some qualitative properties of the solution $\boldsymbol{\Psi}^*$ of (4.4.48). Let Ψ^* denote the piecewise linear interpolant to $[-\beta; \boldsymbol{\Psi}^*; \beta + \alpha]$, then we have the following result.

THEOREM 4.4.1 *Ψ^* is strictly monotone increasing on $[0,1]$, convex on $[0, x_N]$ and concave on $[x_N, 1]$.*

Remark 4.4.1 $x_N = \nu$ is the breakpoint of d as defined in Section 4.3.

Proof As a notational convenience, write

$$J^*_{i-\frac{1}{2}} = \lambda^2(\Psi^*_i - \Psi^*_{i-1})/h_i, \tag{4.4.49}$$

where $\Psi^*_0 := -\beta$, $\Psi^*_{n+1} := \beta + \alpha$. Then, recalling the definitions of $\boldsymbol{V}^0, \boldsymbol{W}^0$, (4.4.48) may be rewritten as

$$J^*_{i+\frac{1}{2}} - J^*_{i-\frac{1}{2}} = \begin{cases} \overline{h}_i[2\delta \sinh(\Psi^*_i) + 1], & i = 1, \ldots, N-1, \\ \overline{h}_N \delta[\exp(\Psi^*_N - \alpha) - \exp(-\Psi^*_N)], & i = N, \\ \overline{h}_i[2\delta \sinh(\Psi^*_i - \alpha) - 1], & i = N+1, \ldots, n. \end{cases} \tag{4.4.50}$$

We shall first establish that

$$J^*_{\frac{1}{2}} > 0, \tag{4.4.51}$$

$$J^*_{n+\frac{1}{2}} > 0. \tag{4.4.52}$$

To obtain (4.4.51), suppose $J^*_{\frac{1}{2}} \leq 0$. Then $\Psi^*_1 \leq \Psi^*_0 = -\beta$. Also if $N \geq 2$, (4.4.50) with $i = 1$ implies

$$J^*_{\frac{3}{2}} - J^*_{\frac{1}{2}} \leq \overline{h}_1[2\delta \sinh(-\beta) + 1] = 0.$$

Hence $J^*_{\frac{3}{2}} - J^*_{\frac{1}{2}} \leq 0$, and so $\Psi^*_2 \leq \Psi^*_1 \leq \Psi^*_0 = -\beta$. Then continuing this argument inductively shows that $J_{N-\frac{1}{2}} \leq 0$ and

$$\Psi^*_N \leq \Psi^*_{N-1} \leq \ldots \leq \Psi^*_1 \leq \Psi^*_0 = -\beta.$$

Now (4.4.50) with $i = N$ gives

$$J^*_{N+\frac{1}{2}} - J^*_{N-\frac{1}{2}} \leq \overline{h}_N \delta[\exp(-\beta - \alpha) - \exp(\beta)] < 0,$$

and hence $J^*_{N+\frac{1}{2}} < J^*_{N-\frac{1}{2}} \leq 0$, and so

$$\Psi^*_{N+1} < \Psi^*_N \leq \ldots \leq \Psi^*_0 = -\beta,$$

which, recalling (4.3.34), contradicts (4.3.36). So (4.4.51) follows. A similar argument by contradiction establishes (4.4.52).

Now we use (4.4.51), (4.4.52) to prove the result. Observe first that (4.4.51) implies $\Psi^*_1 > -\beta$, and , by (4.4.50), we have when $N \geq 2$,

$$J^*_{\frac{3}{2}} - J^*_{\frac{1}{2}} > \overline{h}_1[2\delta \sinh(-\beta) + 1] = 0.$$

So $J^*_{\frac{3}{2}} > J^*_{\frac{1}{2}} > 0$, and $\Psi^*_2 > \Psi^*_1 > -\beta$. Continuing inductively shows

$$J^*_{i+\frac{1}{2}} > J^*_{i-\frac{1}{2}} > 0, \quad i = 1, \ldots, N-1. \tag{4.4.53}$$

Similarly, starting from (4.4.52), and using (4.4.50) for $i = n, \ldots, N+1$ yields

$$J^*_{i-\frac{1}{2}} > J^*_{i+\frac{1}{2}} > 0, \quad i = n, \ldots, N+1. \tag{4.4.54}$$

It follows clearly from (4.4.49), (4.4.53) and (4.4.54) that Ψ^* is strictly monotone increasing. To see that Ψ^* is convex on $[0, x_N]$, observe that for any i such that $1 \leq i \leq N-1$, we have

$$0 < \frac{\lambda^2}{h_i}(\Psi_i^* - \Psi_{i-1}^*) < \frac{\lambda^2}{h_{i+1}}(\Psi_{i+1}^* - \Psi_i^*), \tag{4.4.55}$$

by (4.4.53). Rearranging (4.4.55) gives

$$\Psi_i^* < \Psi_{i-1}^* + \frac{h_i}{h_i + h_{i+1}}(\Psi_{i+1}^* - \Psi_{i-1}^*), \tag{4.4.56}$$

where the right hand expression is the straight line joining Ψ_{i-1}^* to Ψ_{i+1}^* evaluated at the point x_i. Then, since Ψ^* is piecewise linear, (4.4.56) shows that Ψ^* is convex on $[x_{i-1}, x_{i+1}]$. As (4.4.56) holds for all $i = 1, \ldots, N-1$ we conclude that Ψ^* is convex on $[0, x_N]$. The concavity of Ψ^* on $[x_N, 1]$ follows similarly from (4.4.54). ■

The next result gives a quantitative estimate of the width of the interior layer at $x = x_N = \nu$. First we introduce the constant

$$K_1 = 2\delta \cosh \beta \simeq 1 + (3 \times 10^{-16}). \tag{4.4.57}$$

Then define

$$\varepsilon_i = \begin{cases} \lambda^2/(h_{i+1}\overline{h}_i), & i = 1, \ldots, N, \\ \lambda^2/(h_i\overline{h}_i), & i = N+1, \ldots, n. \end{cases}$$

$$\sigma_i = (1+\beta)\varepsilon_i, \qquad i = 1, \ldots, n.$$

THEOREM 4.4.2 *Suppose the mesh (2.2.7) satisfies*

$$\max\{\varepsilon_{N-1}, \varepsilon_{N+1}\} < 1/(2\beta + \alpha). \tag{4.4.58}$$

Then the solution Ψ^* of (4.4.48) satisfies

$$\Psi_{N-1}^* < \sinh^{-1}((\varepsilon_{N-1}(2\beta+\alpha) - 1)/2\delta), \tag{4.4.59}$$

$$0 < \beta + \Psi_{N-i}^* < \beta \prod_{j=2}^{i} \sigma_{N-j} \qquad i = 2, \ldots, N-1, \tag{4.4.60}$$

$$\Psi^*_{N+1} > \alpha + \sinh^{-1}((1 - \varepsilon_{N+1}(2\beta + \alpha))/2\delta), \tag{4.4.61}$$

$$0 < \beta + \alpha - \Psi^*_{N+i} < \beta \prod_{j=2}^{i} \sigma_{N+j} \qquad i = 2, \ldots, n - N. \tag{4.4.62}$$

Remark 4.4.2 Theorem 4.4.2 demonstrates the severe interior layer which can arise in Ψ^* near the point $x_N = \nu$. If, for example, we have a uniform grid with 100 subintervals, then $h = 0.01$, $\varepsilon_i = 1.68 \times 10^{-3}$, and (4.4.58) is satisfied provided $\alpha \in [0, 558]$ This equates to a maximum physical voltage of 14.4V. Then (4.4.59) shows $\Psi^*_{N-1} < -16.98$ and (4.4.60) shows that Ψ^*_{N-i}, $i = 2, \ldots, N-1$ is greater than, but very close to, $-\beta$. In fact, for $i = 4, \ldots, N-1$, Ψ^*_{N-i} is equal to $-\beta$ at least up to the fifth decimal place. A similar argument using (4.4.61), (4.4.62) shows Ψ^*_{N+i} is less than, but very close to $\beta + \alpha$, $i = 1, \ldots, n - N$.

Of course this layer might be resolved by steeply grading the mesh (2.2.7) near ν. However our goal here is to examine the basic theory of convergence of iterative schemes, and in the first instance we need to consider meshes which do not necessarily resolve the layer completely, since this is more likely to be the case when real two–dimensional problems are solved in practice.

Proof We shall prove (4.4.59), (4.4.60) only. Similar arguments prove (4.4.61), (4.4.62). First by (4.4.50) with $i = N - 1$ we have

$$\lambda^2 \left(\frac{\Psi^*_N - \Psi^*_{N-1}}{h_N} - \frac{\Psi^*_{N-1} - \Psi^*_{N-2}}{h_{N-1}} \right) = \overline{h}_{N-1}(2\delta \sinh(\Psi^*_{N-1}) + 1),$$

and since $\Psi^*_{N-1} > \Psi^*_{N-2}$ by Theorem 4.4.1, we have

$$\lambda^2 \left(\frac{\Psi^*_N - \Psi^*_{N-1}}{h_N} \right) > \overline{h}_{N-1}(2\delta \sinh(\Psi^*_{N-1}) + 1).$$

Hence since $\Psi^*_N - \Psi^*_{N-1} < \Psi^*_{n+1} - \Psi^*_0 = 2\beta + \alpha$, we have

$$\frac{\lambda^2}{h_N \overline{h}_{N-1}}(2\beta + \alpha) > 2\delta \sinh(\Psi^*_{N-1}) + 1,$$

from which it follows that

$$\Psi^*_{N-1} < \sinh^{-1}((\varepsilon_{N-1}(2\beta + \alpha) - 1)/2\delta),$$

and we have (4.4.59). Now it follows immediately that as long as (4.4.58) holds then $\Psi^*_{N-1} < 0$. Hence $-\beta < \Psi^*_{N-1} < 0$ and therefore, by Theorem 4.4.1,

$$0 < \beta + \Psi^*_{N-i} < \beta + \Psi^*_{N-1} < \beta, \quad i = 2, \ldots, N-1.$$

So, to prove (4.4.60), we need only show that

$$\beta + \Psi^*_{N-i} \leq \sigma_{N-i}(\beta + \Psi^*_{N-i+1}), \quad i = 2, \ldots, N-1, \tag{4.4.63}$$

and then use a simple induction argument. To obtain (4.4.63), consider the "truncated system" defined by

$$\hat{\mathcal{F}}(\Psi)_j = \lambda^2(\hat{K}(0)[-\beta, \Psi, \Psi^*_{N-i+1}])_j + \overline{h}_j(2\delta \sinh(\Psi_j) + 1), \quad j = 1, \ldots, N-i, \tag{4.4.64}$$

where $\hat{K}(0)$ denotes the first $N-i$ rows and $N-i+2$ columns of $\tilde{K}(0)$. This system, which is to be solved for the unknown $\Psi \in \mathbb{R}^{N-i}$ is, by (4.4.50), just the first $N-i$ equations in (4.4.48), with Ψ^*_{N-i+1} taken as the boundary value on the $(N-i)$th equation. The same argument as that used in Theorem 3.4.1 shows (4.4.64) has a unique solution, which must be $(\Psi^*_1, \ldots, \Psi^*_{N-i})^T$. Now let ρ be the solution of "the characteristic equation"

$$-\frac{\lambda^2}{h_{N-i+1}}\rho + \overline{h}_{N-i}(2\delta \sinh(-\rho) + 1) - \frac{\lambda^2}{h_{N-i+1}}\Psi^*_{N-i+1} = 0.$$

Then since $-\beta < \Psi^*_{N-i+1}$, it follows easily that

$$-\beta < -\rho \leq \Psi^*_{N-i+1}. \tag{4.4.65}$$

Now consider the vectors $\boldsymbol{X}^0, \boldsymbol{Y}^0 \in \mathbb{R}^{N-i}$ given by

$$\boldsymbol{X}^0_j = -\beta, \quad \boldsymbol{Y}^0_j = -\rho \quad j = 1, \ldots, N-i.$$

then following the procedure in Lemma 4.3.2, it is easily shown that $\hat{\mathcal{F}}(\boldsymbol{X}^0) \leq \mathbf{0} \leq \hat{\mathcal{F}}(\boldsymbol{Y}^0)$, and arguing as in Theorem 3.4.1, $\boldsymbol{X}^0$ and $\boldsymbol{Y}^0$ are lower and upper solutions for (4.4.64) and so

$$\Psi^*_{N-i} \leq \boldsymbol{Y}^0_{N-i} = -\rho. \tag{4.4.66}$$

But now we can use Lemma 4.4.3 below, with $\varepsilon = \lambda^2/(h_{N-i+1}\overline{h}_{N-i}) = \varepsilon_{N-i}$, $\gamma = -\Psi^*_{N-i+1}$ to show

$$\rho > \beta - \chi - \chi^2/\varepsilon_{N-i}, \tag{4.4.67}$$

with

$$\begin{aligned} \chi &= (\varepsilon_{N-i}/K_1)(1+\varepsilon_{N-i}/K_1)^{-1}(\beta + \Psi^*_{N-i+1}) \\ &< (\varepsilon_{N-i}/K_1)(\beta + \Psi^*_{N-i+1}) \\ &< (\varepsilon_{N-i}/K_1)\beta < \varepsilon_{N-i}\beta. \end{aligned} \tag{4.4.68}$$

So combining (4.4.66)-(4.4.68),

$$\begin{aligned} \beta + \Psi^*_{N-i} &\leq \beta - \rho \\ &< \chi + \chi^2/\varepsilon_{N-i} \\ &= \chi(1+\chi/\varepsilon_{N-i}) \\ &< (\varepsilon_{N-i}/K_1)(\beta + \Psi^*_{N-i+1})(1+\beta) \\ &< \sigma_{N-i}(\beta + \Psi^*_{N-i+1}) \end{aligned}$$

Hence (4.4.63) is proved and the result follows. ■

The following technical lemma was used in Theorem 4.4.2

LEMMA 4.4.3 *Suppose* $0 < \gamma < \beta$, $\varepsilon > 0$, *and let* ρ *be the solution of the equation*

$$\varepsilon\rho + (2\delta \sinh\rho - 1) = \varepsilon\gamma. \tag{4.4.69}$$

Then

$$\beta - \chi - \chi^2/\varepsilon < \rho < \beta - \chi, \tag{4.4.70}$$

where $\chi = (\varepsilon/K_1)(1+\varepsilon/K_1)^{-1}(\beta - \gamma)$, *and* K_1 *is defined in (4.4.57).*

Remark 4.4.3 If ε is small compared to $\beta - \gamma$, then ρ is close to β.

Proof Observe first that a simple argument by contradiction shows that $0 < \rho < \beta$. Hence by (4.4.69), the mean value theorem and the monotonicity of cosh on $\mathbb{R}^+$,

$$\begin{aligned} \varepsilon\rho &= 2\delta(\sinh\beta - \sinh\rho) + \varepsilon\gamma \\ &< K_1(\beta - \rho) + \varepsilon\gamma. \end{aligned}$$

Hence,

$$\begin{aligned} (1 + \varepsilon/K_1)\rho &< \beta + (\varepsilon/K_1)\gamma, \\ &= (1 + \varepsilon/K_1)\beta - (\varepsilon/K_1)(\beta - \gamma), \end{aligned}$$

and so

$$\rho < \beta - \chi.$$

That is we have proved the right hand inequality in (4.4.70). To obtain the left hand inequality in (4.4.70), use the monotonicity of sinh and the mean value theorem to obtain

$$2\delta\sinh\rho < 2\delta\sinh(\beta - \chi) = 2\delta(\sinh\beta - \chi\cosh\eta) = 1 - 2\delta\chi\cosh\eta,$$

with $\beta - \chi < \eta < \beta$. Hence $(1 - 2\delta\sinh\rho) > 2\delta\chi\cosh\eta$, and by (4.4.69), we have

$$\varepsilon(\rho - \gamma) > 2\delta\chi\cosh\eta. \tag{4.4.71}$$

Now since

$$\eta > \beta - \chi > \rho > 0,$$

we can use (4.4.71), the monotonicity of cosh on $\mathbb{R}^+$ and the mean value theorem again to obtain

$$\varepsilon(\rho - \gamma) > 2\delta\chi\cosh(\beta - \chi) = 2\delta\chi(\cosh\beta - \chi\sinh\zeta),$$

with $\beta - \chi < \zeta < \beta$. Hence,

$$\varepsilon\rho > \varepsilon\gamma + \chi K_1 - \chi^2$$

$$
\begin{aligned}
&= \varepsilon\gamma + \varepsilon(1+\varepsilon/K_1)^{-1}(\beta-\gamma) - \chi^2 \\
&= \varepsilon\beta - \varepsilon(\varepsilon/K_1)(1+\varepsilon/K_1)^{-1}(\beta-\gamma) - \chi^2 \\
&= \varepsilon\beta - \varepsilon\chi - \chi^2,
\end{aligned}
$$

which yields the left hand inequality of (4.4.70). ■

4.5 Numerical experiments

We conclude this chapter with a discussion of some numerical experiments which vindicate the results obtained in the Sections 4.2, 4.3 and 4.4. All of these experiments are conducted on a model for a p–n diode. We choose this to have an equal doping of holes and electrons about the centre point $x = \frac{1}{2}$. That is, the doping profile, d, takes the value -1 on $[0, \frac{1}{2})$, +1 on $(\frac{1}{2}, 1]$ and $d(\frac{1}{2}) = 0$. Note that, in terms of the nomenclature in Section 4.3 this corresponds to $\nu = \frac{1}{2}$. We use the statistics proposed in [62]. This results in the parameter values of

$$
\begin{aligned}
\lambda^2 &= 1.68 \times 10^{-7}, \\
\delta &= 1.22 \times 10^{-8}, \\
\beta &= 18.22.
\end{aligned}
$$

In all cases, the mesh we use may be refined from a uniform mesh to provide resolution about the doping interface. The mesh is initially defined by an odd number, n, of interior nodes. This fixes the uniform mesh with step length $h = 1/(n+1)$, and ensures that $x = \frac{1}{2}$ is a node of that mesh. We then have the option to successively refine this mesh. If we so wish, we are then asked for the number of intervals away from the centre point that we would like to be bisected with a new mesh point. We may repeat this process up to ten times. Hence if a mesh was defined with $n = 9$ and having successive refinement distances of 3 then 2, we would generate a mesh with $h_{max} = \frac{1}{10}$, $h_{min} = \frac{1}{40}$ and with nodal

values

$$\left\{0, \frac{1}{10}, \frac{2}{10}, \frac{5}{20}, \frac{3}{10}, \frac{7}{20}, \frac{4}{10}, \frac{17}{40}, \frac{9}{20}, \frac{19}{40}, \frac{5}{10}, \frac{21}{40}, \frac{11}{20}, \frac{23}{40}, \frac{6}{10}, \frac{13}{20}, \frac{7}{10}, \frac{15}{20}, \frac{8}{10}, \frac{9}{10}, 1\right\}.$$

All computations are performed in MATLAB.

Example 4.5.1 In this example we iterate the map defined by (4.2.3)–(4.2.5) for the p–n diode described above.

α_0	α_1	n	Refinement distances	h_{min}	h_{max}	Its to convergence
0.39	0.00	9	none	1/10	1/10	7
0.39	0.00	19	none	1/20	1/20	7
0.39	0.00	29	none	1/30	1/30	7
0.39	0.00	39	none	1/40	1/40	7
0.39	0.00	49	none	1/50	1/50	7
0.39	0.00	7	2,3,4,2	1/10	1/160	7
0.00	0.39	9	none	1/10	1/10	7
0.00	0.39	19	none	1/20	1/20	7
0.00	0.39	29	none	1/30	1/30	7
0.00	0.39	39	none	1/40	1/40	7
0.00	0.39	49	none	1/50	1/50	7
0.00	0.39	7	2,3,4,2	1/10	1/160	7
0.00	3.89	19	none	1/20	1/20	17
0.00	3.89	29	none	1/30	1/30	32
0.00	3.89	19	5	1/20	1/40	34
0.00	3.89	9	5,5	1/10	1/40	34
0.00	3.89	7	2,3,4,2	1/10	1/160	39
0.00	386.82	19	none	1/20	1/20	39
0.00	386.82	9	5,3	1/10	1/40	60

Table 4.2: Results for Gummel's map.

Table 4.2 details the number of Gummel iterates required for convergence.

This is done for several values of α_0 and α_1 and for various shapes of mesh. Convergence was assumed when the infinity norm of the updates to both $\boldsymbol{V}$ and $\boldsymbol{W}$ was less than 10^{-8}. The boundary conditions correspond to physical voltages of 0.01V, 0.1V and 10V respectively.

As shown by Theorem 4.2.2, we see mesh independent convergence when $\max\{|\alpha_0|, |\alpha_1|\}$ is sufficiently small. Observe that this is the case for both reverse bias ($\alpha_0 < \alpha_1$) and forward bias ($\alpha_0 > \alpha_1$) configurations of the diode. Notice also that as the reverse bias voltage is increased, convergence is maintained although now it appears to be somewhat mesh dependent. This phenomenon could not be predicted from the results of Section 4.2.

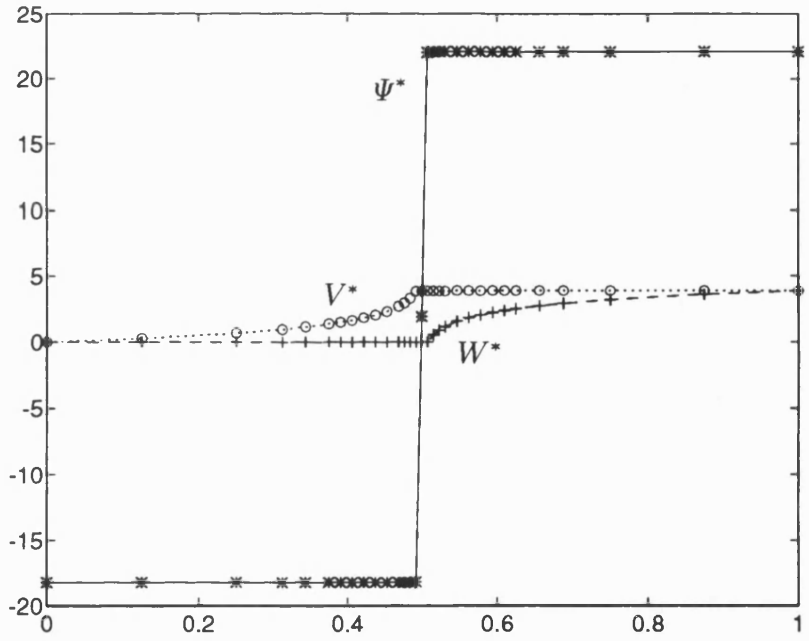


Figure 4.1: Converged solutions, $n = 7$, refinement 2,3,4,2.

In Figure 4.1 we include a plot of the converged solutions for the case $\alpha_0 = 0$, $\alpha_1 = 3.89$, $n = 7$ and refinement distances of 2,3,4 and 2. This illustrates nicely the grading we have achieved in the mesh. Moreover, although this is not quite

the scenario in Section 4.4, we see the distinctive shape properties of Ψ^* that were discussed therein. In particular, we see that the potential is monotone increasing, convex on $[0, \frac{1}{2}]$ and concave on $[\frac{1}{2}, 1]$ as predicted by Theorem 4.4.1. Also observe the extremely sharp layer in the potential at the doping interface. Furthermore, the potential attains its boundary values throughout most of the domain, which we would expect from Theorem 4.4.2.

Example 4.5.2 We now use the algorithm (4.3.27)- (4.3.31) outlined in Section 4.3 to solve our model diode problem with large reverse bias voltage. Here we restrict ourselves to a uniform (unrefined) mesh as we have done in Section 4.3. Table 4.3 shows us that the algorithm converges even when the applied voltage is very large. The convergence criterion for the potential equation was that the infinity norm of the quasi–Newton correction to Ψ was less than 10^{-11}. The continuity equations were considered solved when the infinity norm of the updates of each iterate was less than 10^{-5}.

		Iterations to convergence	
Reverse bias voltage	n	Potential equation	Continuity equations
100V	9	4	34
100V	19	5	39
100V	25	5	41
250V	9	4	36
250V	19	5	47
250V	25	6	66

Table 4.3: Results for algorithm in Section 4.3.

We also note that the convergence of the quasi–Newton iteration for the potential equation is mesh independent as demonstrated in Chapter 3. Again the convergence of the linear continuity problems is slightly mesh dependent. Figures 4.2, 4.3 and 4.4 are plots of the converged solutions Ψ^*, V^* and W^* for an applied

voltage of 100V and mesh with $n = 19$.

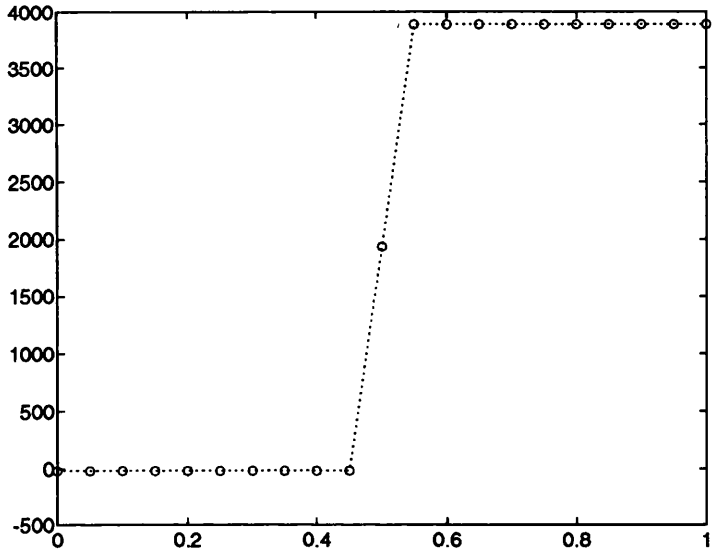


Figure 4.2: Converged Ψ^*, $n = 19$, 100V applied voltage.

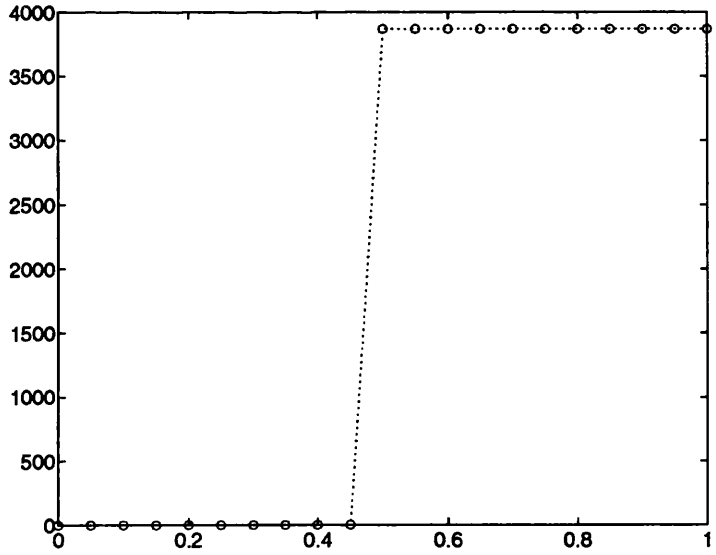


Figure 4.3: Converged V^*, $n = 19$, 100V applied voltage.

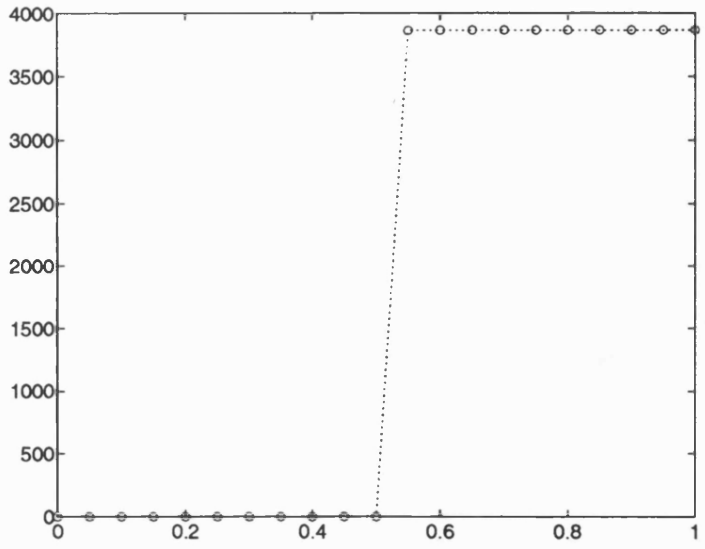


Figure 4.4: Converged W^*, $n = 19$, 100V applied voltage.

The shape results proved in Theorems 4.4.1, 4.4.2 and Lemmas 4.3.3, 4.3.4 are reflected in these plots. Figure 4.2 shows that Ψ^* is monotone increasing, convex on $[0, \frac{1}{2}]$ and concave on $[\frac{1}{2}, 1]$ as predicted by Theorem 4.4.1. It also shows an extremely sharp layer at the doping interface ($x = \frac{1}{2}$). Furthermore, Ψ^* appears to (almost) attain its contact values in either interval which is in accordance with Theorem 4.4.2. We also see from Figure 4.2 that our approximation given by (4.3.37) seems reasonable. Figure 4.3 shows that V^* is monotone increasing and attains its right hand boundary value for $i = N, \ldots, n-1$. Hence $V^* \in K_\alpha^V$ as shown in Lemma 4.3.4. Similarly, Figure 4.4 shows that $W^* \in K_\alpha^W$.

In practice we have found this a very useful algorithm for constructing starting guesses to a decoupled Newton iteration of the full system with recombination included. These starting guesses are very good in the sense that, in all the cases we have tried, they are close enough to the full solution to allow the Newton iterates to converge in their customary quadratic fashion.

Chapter 5

Gummel's map in two dimensions

5.1 Introduction

In this chapter we discuss the convergence of the two–dimensional analogue of one of the schemes studied in Chapter 4. That is, we consider Gummel's decoupling algorithm applied to the system modelling a two–dimensional semiconducting device. Here again, our method of studying the coupled system is an adaption of the results found in [37], [40]. We will set the Gummel iteration up as a map on the appropriately defined set and again use the contraction mapping theorem to show that this scheme converges to a unique fixed point in that set, provided the applied bias across the device is sufficiently small. However, unlike Chapter 4 where we showed that the Lipschitz constant was independent of h, here we will show that it may grow logarithmically with h (as the mesh diameter $h \to 0$). Again we shall require that the mesh is refined in a regular manner.

The resulting linear systems are now far from trivial to solve, and the efficient implementation of a solution technique is considered in the ensuing 3 chapters. Hence all numerical results are reserved until Chapter 8.

5.2 Convergence of Gummel's iteration

Recall from Chapter 2, our approximate solution to (2.2.1)–(2.2.3) is defined to be $(\Psi, V, W) \in S_h(\Omega)^3$ satisfying

$$V|_{\partial\Omega_{D_i}} = W|_{\partial\Omega_{D_i}} = \alpha_i = \text{ constant}, \quad \text{for each } i, \tag{5.2.1}$$

$$\psi|_{\partial\Omega_{D_i}} = \alpha_i + \sinh^{-1}\left(\frac{d|_{\partial\Omega_{D_i}}}{2\delta}\right) =: \alpha_i + \beta_i, \quad \text{for each } i, \tag{5.2.2}$$

and such that

$$\lambda^2(\nabla\Psi,\ \nabla\phi_p) + \langle\delta\{\exp(\Psi - V) - \exp(W - \Psi)\} - d,\ \phi_p\rangle = 0, \tag{5.2.3}$$

$$(\exp(\overline{\Psi - V})\nabla V,\ \nabla\phi_p) - \langle\sigma\rho_v r(\Psi, V, W), \phi_p\rangle = 0, \tag{5.2.4}$$

$$(\exp(\overline{W - \Psi})\nabla W, \nabla\phi_p) + \langle\sigma\rho_w r(\Psi, V, W), \phi_p\rangle = 0, \tag{5.2.5}$$

are satisfied for all nodes $p \notin \partial\Omega_D$. Recall also that we have made the assumption the mesh is both regular and satisfies an inverse assumption.

We begin by making the simplifying assumption

$$r = 0, \tag{5.2.6}$$

where r is the recombination rate in (5.2.4),(5.2.5). This assumption is for theoretical purposes. It allows us to employ a discrete maximum principle to bound the solutions of (5.2.4), (5.2.5). It is made by all the other convergence analyses of Gummel's method of which we are aware ([38], [40]–[42]). It can be physically justified to some extent in the case where the device is a p–n diode in reverse bias, but in general the recombination/generation rate is an essential part of the physical model and is significant in size at least in part(s) of the domain Ω. It remains an open question to repeat the present analysis without the assumption (5.2.6). Also notice that we are making the harmonic average approximation to the coefficients in (5.2.4), (5.2.5). The results given in this chapter hold equally for the standard finite element method.

In this section we will discuss the convergence of the map

$$\mathcal{G} : (V, W) \longmapsto (\tilde{V}, \tilde{W}), \tag{5.2.7}$$

defined as follows.

Step 1. (Fractional Step) Find $\tilde{\Psi} \in S_h$ satisfying (5.2.2) and such that

$$\lambda^2(\nabla\tilde{\Psi}, \nabla\phi_p) + \langle \delta\{\exp(\tilde{\Psi} - V) - \exp(W - \tilde{\Psi})\} - d,\ \phi_p\rangle = 0, \quad p \notin \partial\Omega_D. \tag{5.2.8}$$

Step 2. Find $\tilde{V}$ satisfying (5.2.1) and such that

$$(\overline{\exp(\tilde{\Psi} - V)}\nabla\tilde{V}, \nabla\phi_p) = 0, \quad p \notin \partial\Omega_D. \tag{5.2.9}$$

Step 3. Find $\tilde{W}$ satisfying (5.2.1) and such that

$$(\overline{\exp(W - \tilde{\Psi})}\nabla\tilde{W}, \nabla\phi_p) = 0, \quad p \notin \partial\Omega_D. \tag{5.2.10}$$

Following [37], [40] we do this using the contraction mapping theorem in the set $B(\Omega)$ defined by (3.5.35) in Chapter 3. We equip $B(\Omega)$ with the norm

$$\|(V, W)\|_{B(\Omega)} = \{\|V\|_{H^1(\Omega)} + \|W\|_{H^1(\Omega)}\},$$

with respect to which $B(\Omega)$ is a complete metric space.

Firstly recall from Chapter 3, given any $(V, W) \in B(\Omega)$ we can use (3.5.43), (3.5.44) to solve the discretised potential equation defined by (5.2.8) to obtain $\tilde{\boldsymbol{\Psi}}$, such that the piecewise linear interpolant $\tilde{\Psi}$ associated with $\tilde{\boldsymbol{\Psi}}$ satisfies $\tilde{\Psi} \in E(\Omega)$. Recall from Chapter 3

$$E(\Omega) := \left\{\Psi \in S_h(\Omega) : x^0 \leq \Psi \leq y^0\right\},$$

where x^0, y^0 are defined by (3.5.41), (3.5.42) in Section 3.5.

In Theorem 5.2.2 below we will demonstrate the Lipschitz continuity of $\mathcal{G}$. We shall show that its Lipschitz constant is less than 1 provided $\underline{\alpha}$ and $\overline{\alpha}$, defined

by (3.5.36), are sufficiently close to zero when h is fixed. (A result analogous to this is also obtained in [40], but with a different choice of variables.) Here we shall also show that the Lipschitz constant of $\mathcal{G}$ grows only logarithmically in h as the mesh is refined.

The first step, given in Theorem 5.2.1, is to examine the continuity of the fractional step (5.2.8). Then in Theorem 5.2.2 we examine the continuity of steps (5.2.9), (5.2.10). In the next two results, (V^i, W^i), $i = 1, 2$ will denote two arbitrary elements of $B(\Omega)$. For each i, $\tilde{\Psi}^i$ will be the corresponding solutions of (5.2.8), and $(\tilde{V}^i, \tilde{W}^i)$ will be the corresponding solutions of (5.2.9), (5.2.10), with all solutions satisfying the appropriate Dirichlet boundary conditions. Our methods of proof are adapted from [40], with changes necessary to handle the different set of variables used here, as well as to deal with the approximations due to mass lumping and harmonic averaging. In addition, through the use of a discrete Sobolev inequality, we are able to obtain the behaviour of the Lipschitz constant of $\mathcal{G}$ with respect to h.

Throughout the following proofs, C will denote a generic constant which is always independent of h, but may depend on other parameters as stated.

THEOREM 5.2.1 *For each $M > 0$, there exists a constant C which is independent of h such that*

$$|\tilde{\Psi}^1 - \tilde{\Psi}^2|_{H^1(\Omega)} \leq C \, \|(V^1, W^1) - (V^2, W^2)\|_{B(\Omega)},$$

for all $(V^1, W^1), (V^2, W^2) \in B(\Omega)$, provided $\max\{\underline{\alpha}, \overline{\alpha}\} \leq M$.

Proof . By Theorem 3.5.2, given $(V^i, W^i) \in B(\Omega)$, for $i = 1, 2$, there is a unique $\tilde{\Psi}^i$ satisfying (5.2.2) and

$$\lambda^2(\nabla \tilde{\Psi}^i, \nabla \phi) + \langle \delta\{\exp(\tilde{\Psi}^i - V^i) - \exp(W^i - \tilde{\Psi}^i)\} - d, \phi \rangle = 0$$

for all test functions $\phi \in S_h$ which vanish on $\partial\Omega_D$. Subtracting the case $i = 1$ from the case $i = 2$ and putting $\phi = \tilde{\Psi}^2 - \tilde{\Psi}^1$ yields

$$\lambda^2(\nabla(\tilde{\Psi}^2 - \tilde{\Psi}^1), \nabla(\tilde{\Psi}^2 - \tilde{\Psi}^1)) +$$

$$\delta\langle \exp(\tilde{\Psi}^2 - V^2) - \exp(W^2 - \tilde{\Psi}^2) - \exp(\tilde{\Psi}^1 - V^1) + \exp(W^1 - \tilde{\Psi}^1), \tilde{\Psi}^2 - \tilde{\Psi}^1 \rangle$$
$$= 0.$$

We rewrite this as

$$\lambda^2(\nabla(\tilde{\Psi}^2 - \tilde{\Psi}^1),\ \nabla(\tilde{\Psi}^2 - \tilde{\Psi}^1)) + \delta\langle t_1,\ \tilde{\Psi}^2 - \tilde{\Psi}^1\rangle = -\delta\langle t_2,\ \tilde{\Psi}^2 - \tilde{\Psi}^1\rangle, \quad (5.2.11)$$

where

$$t_1 = \{\exp(\tilde{\Psi}^2) - \exp(\tilde{\Psi}^1)\}\exp(-V^2) + \{\exp(-\tilde{\Psi}^1) - \exp(-\tilde{\Psi}^2)\}\exp(W^2),$$
$$t_2 = \exp(\tilde{\Psi}^1)\{\exp(-V^2) - \exp(-V^1)\} + \exp(-\tilde{\Psi}^1)\{\exp(W^1) - \exp(W^2)\}.$$

It is easily shown by the mean value theorem that

$$\langle t_1,\ \tilde{\Psi}^1 - \tilde{\Psi}^2\rangle \geq 0. \quad (5.2.12)$$

Now by the definition of the discrete bilinear form, (2.3.46), and the Cauchy-Schwarz inequality,

$$|\langle t_2,\ \tilde{\Psi}^2 - \tilde{\Psi}^1\rangle| \leq \sum_T \mathcal{A}(T)\|t_2\|_{L_\infty(T)}\|\tilde{\Psi}^2 - \tilde{\Psi}^1\|_{L_\infty(T)}$$
$$\leq \left\{\sum_T \mathcal{A}(T)\|t_2\|^2_{L_\infty(T)}\right\}^{1/2} \left\{\sum_T \mathcal{A}(T)\|\tilde{\Psi}^2 - \tilde{\Psi}^1\|^2_{L_\infty(T)}\right\}^{1/2} \quad (5.2.13)$$

Now using the fact that $(V^i, W^i) \in B(\Omega)$ for each i and the bounds on $\tilde{\Psi}^i$ which follow from Theorem 3.5.2, we have

$$\|t_2\|_{L_\infty(T)} \leq C\{\|V^2 - V^1\|_{L_\infty(T)} + \|W^2 - W^1\|_{L_\infty(T)}\}.$$

But since $V^2 - V^1 \in S_h$ we can use a standard inverse inequality (see, for example [13, Theorem 3.2.6]):

$$\|V^2 - V^1\|_{L_\infty(T)} \leq C\mathcal{A}(T)^{-1/2}\|V^2 - V^1\|_{L_2(T)}.$$

Using an identical bound for $\|W^2 - W^1\|_{L_\infty(T)}$, we obtain

$$\|t_2\|_{L_\infty(T)} \leq C\mathcal{A}(T)^{-1/2}\left\{\|V^2 - V^1\|_{L_2(T)} + \|W^2 - W^1\|_{L_2(T)}\right\}.$$

Hence

$$\mathcal{A}(T)\|t_2\|^2_{L_\infty(T)} \leq 2C^2\left\{\|V^2 - V^1\|^2_{L_2(T)} + \|W^2 - W^1\|^2_{L_2(T)}\right\}.$$

Thus, recalling that C is a generic constant,

$$\begin{aligned}\sum_T \mathcal{A}(T)\|t_2\|^2_{L_\infty(T)} &\leq C\left\{\|V^2 - V^1\|^2_{L_2(\Omega)} + \|W^2 - W^1\|^2_{L_2(\Omega)}\right\} \\ &\leq C\|(V^2, W^2) - (V^1, W^1)\|^2_{B(\Omega)}. \qquad (5.2.14)\end{aligned}$$

By a similar but easier argument,

$$\begin{aligned}\sum_T \mathcal{A}(T)\|\tilde{\Psi}^2 - \tilde{\Psi}^1\|^2_{L_\infty(T)} &\leq C\sum_T \|\tilde{\Psi}^2 - \tilde{\Psi}^1\|^2_{L_2(T)} \\ = C\|\tilde{\Psi}^2 - \tilde{\Psi}^1\|^2_{L_2(\Omega)} &\leq C|\tilde{\Psi}^2 - \tilde{\Psi}^1|^2_{H^1(\Omega)}, \qquad (5.2.15)\end{aligned}$$

where the final step uses Poincaré's inequality. Substituting (5.2.14), (5.2.15) into (5.2.13) yields

$$|\langle t_2, \tilde{\Psi}^2 - \tilde{\Psi}^1\rangle| \leq C\|(V^2, W^2) - (V^1, W^1)\|_{B(\Omega)}|\tilde{\Psi}^2 - \tilde{\Psi}^1|_{H^1(\Omega)}. \qquad (5.2.16)$$

Now (5.2.11), (5.2.12) and (5.2.16) yield the required result. ■

THEOREM 5.2.2 *The solutions* $\tilde{V}^i$, $\tilde{W}^i$, $i = 1, 2$ *satisfy*

$$(\tilde{V}^i, \tilde{W}^i) \in B(\Omega).$$

Also, for each $M > 0$, *there exists a constant* C *independent of* h

$$\begin{aligned}&\|(\tilde{V}^1, \tilde{W}^1) - (\tilde{V}^2, \tilde{W}^2)\|_{B(\Omega)} \leq \\ &\quad C\max\{|\underline{\alpha}|, |\overline{\alpha}|\}(1 - \log(h))^{1/2}\|(V^1, W^1) - (V^2, W^2)\|_{B(\Omega)}, \qquad (5.2.17)\end{aligned}$$

for all $(V^1, W^1), (V^2, W^2) \in B(\Omega)$, *provided* $\max\{\underline{\alpha}, \overline{\alpha}\} \leq M$.

Proof To show that $(\tilde{V}^i, \tilde{W}^i) \in B(\Omega)$ we show that

$$\underline{\alpha} \leq \tilde{V}^i \leq \overline{\alpha}.$$

The corresponding result for $\tilde{W}^i$ is analogous. For this argument let $\overline{\alpha}$ denote the element of S_h which takes the value $\overline{\alpha}$ at every node on $\Omega \cup \partial\Omega$. Clearly then $\nabla\,\overline{\alpha} = \mathbf{0}$. So by (5.2.9),

$$(\exp(\overline{\tilde{\Psi}^i - V^i})\nabla(\tilde{V}^i - \overline{\alpha}),\ \nabla\phi_p) = 0, \quad i = 1,2, \tag{5.2.18}$$

for all $p \notin \partial\Omega_D$. Letting $\boldsymbol{x}, \boldsymbol{x}_D$ denote the values of $\tilde{V}^i - \overline{\alpha}$ at nodes in $\overline{\Omega}\backslash\partial\Omega_D$ and on $\partial\Omega_D$ respectively, we see that (5.2.18) is in the form

$$K\boldsymbol{x} + K_D\boldsymbol{x}_D\ = \mathbf{0},$$

where K represents the coupling between the nodes on $\overline{\Omega}\backslash\partial\Omega_D$ induced by the bilinear form in (5.2.9), whereas K_D represents the analogous coupling between the nodes on $\overline{\Omega}\backslash\partial\Omega_D$ and $\partial\Omega_D$. By the assumed properties of the meshes, we know that $K^{-1} > 0$ and that K_D contains only non-positive entries. Then, (since $\boldsymbol{x}_D \leq \mathbf{0}$), we have

$$\boldsymbol{x} = -K^{-1}\ K_D\ \boldsymbol{x}_D \leq \mathbf{0}.$$

This proves $\tilde{V}^i \leq \overline{\alpha}$. The proof of $\tilde{V}^i \geq \underline{\alpha}$ is analogous.

To prove the bound (5.2.17) we proceed again as in [40, Chapter 4]. Write

$$\begin{aligned}
&(\exp(\overline{\tilde{\Psi}^1 - V^1})\nabla(\tilde{V}^2 - \tilde{V}^1),\ \nabla(\tilde{V}^2 - \tilde{V}^1)) \\
&\qquad = (\{\exp(\overline{\tilde{\Psi}^1 - V^1}) - \exp(\overline{\tilde{\Psi}^2 - V^2})\}\ \nabla\tilde{V}^2,\ \nabla(\tilde{V}^2 - \tilde{V}^1)) \\
&\qquad\quad + (\exp(\overline{\tilde{\Psi}^2 - V^2})\ \nabla\tilde{V}^2,\ \nabla(\tilde{V}^2 - \tilde{V}^1)) \\
&\qquad\quad - (\exp(\overline{\tilde{\Psi}^1 - V^1})\ \nabla\tilde{V}^1,\ \nabla(\tilde{V}^2 - \tilde{V}^1)).
\end{aligned} \tag{5.2.19}$$

By (5.2.9), the last two terms on the right-hand side of (5.2.19) vanish. Hence, using the fact that $(V^1, W^1), (V^2, W^2) \in B(\Omega)$ and Theorem 3.5.2, the Cauchy-Schwarz inequality gives us

$$C|\tilde{V}^2 - \tilde{V}^1|^2_{H^1(\Omega)} \leq \|\exp(\overline{\tilde{\Psi}^1 - V^1}) - \exp(\overline{\tilde{\Psi}^2 - V^2})\|_{L_\infty(\Omega)}\ |\tilde{V}^2|_{H^1(\Omega)}\ |\tilde{V}^2 - \tilde{V}^1|_{H^1(\Omega)},$$

and hence

$$|\tilde{V}^2 - \tilde{V}^1|_{H^1(\Omega)} \leq C\|\exp(\overline{\tilde{\Psi}^1 - V^1}) - \exp(\overline{\tilde{\Psi}^2 - V^2})\|_{L_\infty(\Omega)}\ |\tilde{V}^2|_{H^1(\Omega)}. \tag{5.2.20}$$

We shall bound the two terms on the right-hand side of (5.2.20) separately. Considering the second term, recall that $\tilde{V}^2$ is defined by (5.2.9) with $V = V^2$ and $\tilde{\Psi} = \tilde{\Psi}^2$. For each i, define u_i to be the solution of the weak problem: Find $u_i \in H^1(\Omega)$ with $u_i|_{\partial\Omega_{D_j}} = \delta_{ij}$ for each j and such that

$$(\overline{\exp(\tilde{\Psi}^2 - V^2)}\nabla u_i,\ \nabla\phi) = 0, \quad \text{for all } \phi \in H^1,\ \ \phi = 0 \ \text{ on } \ \partial\Omega_D. \tag{5.2.21}$$

Let $U_i \in S_h$ be the usual finite element approximation of u_i. Then by standard theory, these finite element problems are well-posed and the solution is stable in the energy norm, i.e. $|U_i|_{H^1(\Omega)} \leq C$, with C independent of h. Then by uniqueness and linearity, $\tilde{V}^2 = \sum_i \alpha_i U_i$, and hence we obtain the bound for the second term of (5.2.20):

$$|\tilde{V}^2|_{H^1(\Omega)} \leq \sum_i |\alpha_i||U_i|_{H^1(\Omega)} \leq C \max\{|\underline{\alpha}|, |\overline{\alpha}|\}, \tag{5.2.22}$$

with C independent of h and α_i for each i .

Considering now the first term of (5.2.20), we claim that it can be bounded by

$$\|\overline{\exp(\tilde{\Psi}^1 - V^1)} - \overline{\exp(\tilde{\Psi}^2 - V^2)}\|_{L_\infty(\Omega)} \leq C\{\|\tilde{\Psi}^1 - \tilde{\Psi}^2\|_{L_\infty(\Omega)} + \|V^1 - V^2\|_{L_\infty(\Omega)}\}. \tag{5.2.23}$$

This can be proved by observing that on each triangle T,

$$\begin{aligned}
&|\overline{\exp(\tilde{\Psi}^1 - V^1)} - \overline{\exp(\tilde{\Psi}^2 - V^2)}| \\
&\qquad = \mathcal{A}(T)\left|\left\{\int_T \exp(V^1 - \tilde{\Psi}^1)\right\}^{-1} - \left\{\int_T \exp(V^2 - \tilde{\Psi}^2)\right\}^{-1}\right| \\
&\qquad = \mathcal{A}(T)\frac{|\int_T\{\exp(V^2 - \tilde{\Psi}^2) - \exp(V^1 - \tilde{\Psi}^1)\}|}{\{\int_T \exp(V^1 - \tilde{\Psi}^1)\}\{\int_T \exp(V^2 - \tilde{\Psi}^2)\}} \\
&\qquad \leq C\mathcal{A}(T)^{-1}\int_T |\exp(V^2 - \tilde{\Psi}^2) - \exp(V^1 - \tilde{\Psi}^1)| \\
&\qquad \leq C\|\exp(V^2 - \tilde{\Psi}^2) - \exp(V^1 - \tilde{\Psi}^1)\|_{L_\infty(T)}.
\end{aligned}$$

Now using the mean value theorem, the bound (5.2.23) follows.

In order to make use of Theorem 5.2.1 and to prove that the map $\mathcal{G}$ is a contraction, we would like now to bound the right-hand side of (5.2.23) in terms

of the $H^1(\Omega)$ seminorms of $\tilde{\Psi}^1 - \tilde{\Psi}^2$ and $V^1 - V^2$. The Sobolev embedding theorem would allow us to do this if the norms on the right-hand side of (5.2.23) were in $L_p(\Omega)$ for some $p < \infty$, but this embedding fails when $p = \infty$. However there is a discrete Sobolev inequality which yields a mesh-dependent bound in the case $p = \infty$. This bound is well known in the domain decomposition literature (e.g. [19, Lemma 2]), and it states that if $X \in S_h$ with $X = 0$ at any point on $\overline{\Omega}$, then

$$\|X\|_{L_\infty(\Omega)} \leq C(1 - \log(h))^{1/2} |X|_{H^1(\Omega)}.$$

Using this we obtain from (5.2.23) and using Theorem 5.2.1,

$$\begin{aligned} &\| \exp(\overline{\tilde{\Psi}^1 - V^1}) - \exp(\overline{\tilde{\Psi}^2 - V^2}) \|_{L_\infty(\Omega)} \\ &\qquad \leq C(1 - \log(h))^{1/2} \|(V^1, W^1) - (V^2, W^2)\|_{B(\Omega)}, \end{aligned}$$

with C independent of h. Using this and (5.2.22) in (5.2.20) yields

$$|\tilde{V}^2 - \tilde{V}^1|_{H^1(\Omega)} \leq C \max\{|\underline{\alpha}|, |\overline{\alpha}|\}(1 - \log(h))^{1/2} \|(V^1, W^1) - (V^2, W^2)\|_{B(\Omega)}.$$

An analogous bound is obtained for $|\tilde{W}^2 - \tilde{W}^1|_{H^1(\Omega)}$, completing the proof.

■

The following corollary is obtained using the contraction mapping theorem.

COROLLARY 5.2.3 *With $r = 0$, Gummel's method (5.2.8) - (5.2.10) converges for each fixed h provided $\max\{\underline{\alpha}, \overline{\alpha}\}$ is sufficiently small.*

This section shows overall that the convergence of Gummel's method only degrades, at worst, logarithmically with h as the mesh is refined. Each iterate of Gummel's method requires the solution of a large sparse system of equations. All these systems are symmetric positive definite, but (especially in the case of (5.2.9), (5.2.10)) they are very poorly conditioned due to severe layers in the exponential coefficients. In the following chapter we describe parallel methods of solving these systems. The rate of convergence of one of these methods degrades

only logarithmically as the mesh is refined, and its performance is independent of jump discontinuities across substructure boundaries of the coefficients of the underlying PDEs. It is this method which seems most suitable for solving the semiconductor equations.

Chapter 6

Domain decomposition methods

6.1 Introduction

In this chapter we are concerned with the *massively parallel* solution of the symmetric positive definite (SPD) linear systems arising from Gummel's method applied to the two–dimensional device problem. We do this by the conjugate gradient method (CGM) using domain decomposition as a preconditioner. Domain decomposition ideas lead to extremely natural algorithms on massively parallel computers for the solution of elliptic problems [7], [19], [67], [20], [65], [66], [5], [6], [46]. Our method is essentially one of those proposed in [66], this is an *additive Schwarz* method (see for example [19], [20], [67], [65], [66], [46], [8]).

We will focus our attention on the linear problems arising from the electron and hole continuity equations in Gummel's method. These problems suffer very severe jumps in the coefficient function appearing in the second order operator. It is for these sorts of problems that the ensuing theory is most powerful. However it should be pointed out that the linear solves required at each step of the new quasi–Newton method outlined in Chapter 3 can also be achieved by these techniques.

The steps in the iteration proposed in (5.2.8)–(5.2.10) which are specifically for the continuity equations lead to the solution by finite elements of a sequence

of linear PDE problems each of which is in the form: Find $u \in H^1(\Omega)$ satisfying Dirichlet boundary conditions on $\partial\Omega_D$ and such that

$$(a\nabla u, \nabla\phi) = (f, \phi), \tag{6.1.1}$$

for all test functions $\phi \in H^1(\Omega)$ which vanish on $\partial\Omega_D$. Here a and f are known functions which may suffer severe jumps across layers interior to the domain Ω. We assume that the coefficient function a is bounded above and below by positive constants. Note that, by Theorem 5.2.2, the linear problems resulting from Gummel's map (5.2.7), satisfy this assumption.

For convenience we shall assume zero Dirichlet conditions on $\partial\Omega_D$. Then the finite element method with mass lumping applied to (6.1.1) gives rise to a system which may be written:

$$K\boldsymbol{x} = \boldsymbol{b}. \tag{6.1.2}$$

Here $\boldsymbol{x}$ denotes the solution vector to be found and

$$K_{pq} = (a\nabla\phi_p, \nabla\phi_q), \tag{6.1.3}$$

$$b_p = \langle f, \phi_p \rangle, \tag{6.1.4}$$

where $\{\phi_p\}$ are the usual basis functions for $S_h(\Omega)$, ϕ_p being one at the pth node and zero elsewhere. Here p, q range over all fine grid nodes in $\overline{\Omega}\backslash\partial\Omega_D$. The introduction of non–zero Dirichlet conditions on $\partial\Omega_D$ just yields a different right–hand side in (6.1.2). Note that here we have used the quadrature rule (2.3.44) in order to mass lump the right hand side of (6.1.2). This is purely for the convenience of comparing this model problem to our discretised linear problems (5.2.9), (5.2.10) originating from a semiconductor model. The use of this mass lumping has no other significance in the following analysis.

6.2 Basic domain decomposition technique

Recall the decomposition of Ω into substructures $\Omega^{(i)}$ that was introduced in Section 2.2. When working on a machine with a large two dimensional array of processing elements it will be convenient to label the substructures as a two dimensional array, which then yields an obvious mapping from the substructures to the array of processing elements. However, in principle there need be no assumption of a natural ordering of the $\Omega^{(i)}$. The version of the domain decomposition technique we shall employ involves local elimination of nodes interior to the substructures. To describe it we will introduce the following notation.

- $\partial\Omega^{(i)}$ = boundary of $\Omega^{(i)}$.
- $\Gamma^{(i)} = \partial\Omega^{(i)} \backslash \partial\Omega_D$ = those parts of the substructure boundaries on which a Dirichlet condition is not present.
- $\Pi_h^{(i)}$ = nodes of the fine grid which lie on $\Gamma^{(i)}$.
- $\Pi_h = \cup_i \Pi_h^{(i)}$.
- $\Pi_H^{(i)}$ = nodes of the coarse grid which lie on $\Gamma^{(i)}$.
- $\Pi_H = \cup_i \Pi_H^{(i)}$.

For any set of nodes $\mathcal{N}$ a **nodal vector** on $\mathcal{N}$ is a vector with a unique entry for each node. The set of all nodal vectors on $\mathcal{N}$ is denoted $[\mathcal{N}]$ and the dimension of that space is denoted $|\mathcal{N}|$.

With this we can write (6.1.2) as

$$\sum_i K^{(i)} \boldsymbol{x}^{(i)} = \sum_i \boldsymbol{b}^{(i)}. \tag{6.2.5}$$

where

$$x_p^{(i)} = \begin{cases} x_p & \text{if node } p \in \Omega^{(i)} \cup \Gamma^{(i)} \\ 0 & \text{otherwise} \end{cases} \tag{6.2.6}$$

$$K_{pq}^{(i)} = (a\nabla\phi_p, \nabla\phi_q)_{\Omega^{(i)}}, \tag{6.2.7}$$

and

$$b_p^{(i)} = \langle f, \phi_p \rangle_{\Omega^{(i)}}, \tag{6.2.8}$$

In the above p, q range over all fine grid nodes in $\Omega \cup \partial\Omega_N$ and the subscript on the inner products denotes that they are taken over the substructure $\Omega^{(i)} \cup \Gamma^{(i)}$, i.e. (6.2.5) is just (6.1.2) written in "subassembly form". It is helpful to interpret (6.2.5) in a slightly different way. Remove from $K^{(i)}$ the rows and columns corresponding to nodes which are outside $\Omega^{(i)} \cup \Gamma^{(i)}$ (these are all zero of course). Similarly modify $\boldsymbol{x}^{(i)}$ and $\boldsymbol{b}^{(i)}$. Then (6.2.5) still holds, but $\sum_i$ should be interpreted to mean "extension to vectors on all the nodes in $\Omega \cup \partial\Omega_N$ by padding with zeros and then summation"

Then (as in [65] and [66] for example), for each i, we can partition the vector $\boldsymbol{x}^{(i)}$ into a part $\boldsymbol{x}_I^{(i)}$ containing its values at nodes in $\Omega^{(i)}$ and a part $\boldsymbol{x}_B^{(i)}$ containing its values on $\Gamma^{(i)}$. This induces a partition of the whole solution vector $\boldsymbol{x}$ into a part $\boldsymbol{x}_I$ of nodal values interior to substructures and a part $\boldsymbol{x}_B \in [\Pi_h]$. We can partition the right hand side vector $\boldsymbol{b}$ in the same way. With obvious notation we can also partition the matrix $K^{(i)}$ into blocks with rows and columns corresponding to interior or boundary nodes of $\Omega^{(i)}$. Then (6.2.5) may be written

$$\sum_i \begin{bmatrix} K_{II}^{(i)} & K_{IB}^{(i)} \\ K_{IB}^{(i)T} & K_{BB}^{(i)} \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_I^{(i)} \\ \boldsymbol{x}_B^{(i)} \end{bmatrix} = \sum_i \begin{bmatrix} \boldsymbol{b}_I^{(i)} \\ \boldsymbol{b}_B^{(i)} \end{bmatrix}. \tag{6.2.9}$$

Since the interior nodal values of each substructure are independent of those of any other substructure we can eliminate them, as shown in Theorem 6.2.1 below. First introduce the *Schur complements*

$$S^{(i)} = K_{BB}^{(i)} - K_{IB}^{(i)T} K_{II}^{(i)-1} K_{IB}^{(i)}. \tag{6.2.10}$$

THEOREM 6.2.1 *$\boldsymbol{x}$ solves (6.2.9) if and only if, for each i, $\boldsymbol{x}_I^{(i)}, \boldsymbol{x}_B^{(i)}$ solve the system:*

$$\sum_i S^{(i)} \boldsymbol{x}_B^{(i)} = \sum_i \left\{ \boldsymbol{b}_B^{(i)} - K_{IB}^{(i)T} K_{II}^{(i)-1} \boldsymbol{b}_I^{(i)} \right\}, \tag{6.2.11}$$

and

$$K_{II}^{(i)} \boldsymbol{x}_I^{(i)} + K_{IB}^{(i)} \boldsymbol{x}_B^{(i)} = \boldsymbol{b}_I^{(i)}. \tag{6.2.12}$$

Proof Suppose $\boldsymbol{x}$ solves (6.2.9). Then observing (6.2.9) at nodes on Γ, and using the definition of $K^{(i)}$ gives

$$\sum_i \left(K_{IB}^{(i)T} \boldsymbol{x}_I^{(i)} + K_{BB}^{(i)} \boldsymbol{x}_B^{(i)} \right) = \sum_i \boldsymbol{b}_B^{(i)}. \tag{6.2.13}$$

Also observing (6.2.9) at internal nodes of any substructure $\Omega^{(i)}$ gives (6.2.12) which implies

$$\boldsymbol{x}_I^{(i)} = K_{II}^{(i)-1} \boldsymbol{b}_I^{(i)} - K_{II}^{(i)-1} K_{IB}^{(i)} \boldsymbol{x}_B^{(i)}.$$

Now substitution for $\boldsymbol{x}_I^{(i)}$ in (6.2.13) yields (6.2.11).

Conversely, suppose $\boldsymbol{x}$ solves (6.2.11), (6.2.12). Then rearranging (6.2.11) we have

$$\sum_i \left(K_{IB}^{(i)T} K_{II}^{(i)-1} (\boldsymbol{b}_I^{(i)} - K_{IB}^{(i)} \boldsymbol{x}_B^{(i)}) + K_{BB}^{(i)} \boldsymbol{x}_B^{(i)} \right) = \sum_i \boldsymbol{b}_B^{(i)}.$$

Using (6.2.12) gives

$$\sum_i (K_{IB}^{(i)T} \boldsymbol{x}_I^{(i)} + B_{BB}^{(i)} \boldsymbol{x}_B^{(i)}) = \sum_i \boldsymbol{b}_B^{(i)}.$$

But this is just (6.2.9) observed on Γ. Moreover (6.2.12) is just (6.2.9) observed on the interior of each $\Omega^{(i)}$, so the result follows. ■

Now we can think of (6.2.11) as an equation for the unknown values of $\boldsymbol{x}_B$. We could write this system as

$$S \boldsymbol{x}_B = \boldsymbol{c}_B. \tag{6.2.14}$$

As suggested in [65], [66] we shall solve (6.2.14) by the conjugate gradient method (CGM). For this algorithm it is well–known (see, for example, [39, page 134]) that

$$\|\boldsymbol{x} - \boldsymbol{x}^k\|_S \leq 2 \left[\frac{\sqrt{\kappa(S)} - 1}{\sqrt{\kappa(S)} + 1} \right]^{k-1} \|\boldsymbol{x} - \boldsymbol{x}^1\|_S, \tag{6.2.15}$$

where $\|\boldsymbol{y}\|_S = (S\boldsymbol{y}, \boldsymbol{y})^{\frac{1}{2}}$, $\boldsymbol{x}^k$ is the kth iterate of the CGM and $\boldsymbol{x}$ solves (6.2.14). Here $\kappa(S)$ is the condition number of the matrix S and is defined to be

$$\kappa(S) = \|S\| \, \|S^{-1}\|. \tag{6.2.16}$$

If we now take the case $\|\cdot\| = \|\cdot\|_{L_2(\Omega)}$ and note the fact that S is symmetric, positive definite (SPD) (shown later in Lemma 6.2.3), then for our applications (6.2.16) is equivalent to

$$\kappa(S) = \frac{\lambda_{max}(S)}{\lambda_{min}(S)}. \tag{6.2.17}$$

In (6.2.17), $\lambda_{max}(S)$ represents the maximum eigenvalue of S and $\lambda_{min}(S)$ is the minimum eigenvalue of S. (Recall that since S is SPD, all its eigenvalues will be real and positive.) Hence our first concern if we are to use the CGM to solve (6.2.14) is that we have not increased the conditioning of our problem by reducing to the Schur complement system. The following three lemmas show that the condition number of the Schur complement matrix S is no worse than that of our original stiffness matrix K.

LEMMA 6.2.2 *If A is positive definite and symmetric $n \times n$ and if $\boldsymbol{b} \in \mathbb{R}^n$, define the quadratic functional*

$$\varphi(\boldsymbol{p}) = \boldsymbol{p}^T A \boldsymbol{p} + 2\boldsymbol{b}^T \boldsymbol{p}.$$

Then

$$\min_{p \in \mathbb{R}^n} \varphi(\boldsymbol{p}) = \varphi(\boldsymbol{p}^0),$$

where $\boldsymbol{p}^0$ is the unique solution to $A\boldsymbol{p}^0 + \boldsymbol{b} = \boldsymbol{0}$.

Proof Let $\boldsymbol{p}^0$ be the unique solution of $A\boldsymbol{p}^0 + \boldsymbol{b} = \boldsymbol{0}$. Then

$$\begin{aligned}
\varphi(\boldsymbol{p}) &= \boldsymbol{p}^T A \boldsymbol{p} + 2\boldsymbol{b}^T \boldsymbol{p}, \\
&= (\boldsymbol{p} - \boldsymbol{p}^0)^T A (\boldsymbol{p} - \boldsymbol{p}^0) + 2(\boldsymbol{b}^T + \boldsymbol{p}^{0\,T} A)\boldsymbol{p} - \boldsymbol{p}^{0\,T} A \boldsymbol{p}^0, \\
&= (\boldsymbol{p} - \boldsymbol{p}^0)^T A (\boldsymbol{p} - \boldsymbol{p}^0) - \boldsymbol{p}^{0\,T} A \boldsymbol{p}^0, \\
&\geq -\boldsymbol{p}^{0\,T} A \boldsymbol{p}^0,
\end{aligned}$$

with equality if and only if $\boldsymbol{p} = \boldsymbol{p}^0$. ■

LEMMA 6.2.3 *For all* $\boldsymbol{x}_B \in [\Pi_h]$

$$\begin{aligned} \boldsymbol{x}_B^T S \boldsymbol{x}_B &= \min_{\boldsymbol{x}_I} [\boldsymbol{x}_I^T, \boldsymbol{x}_B^T] \begin{bmatrix} K_{II} & K_{IB} \\ K_{IB}^T & K_{BB} \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_I \\ \boldsymbol{x}_B \end{bmatrix} \\ &= [\boldsymbol{x}_I^{0\,T}, \boldsymbol{x}_B] \begin{bmatrix} K_{II} & K_{IB} \\ K_{IB}^T & K_{BB} \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_I^0 \\ \boldsymbol{x}_B \end{bmatrix}, \end{aligned}$$

where

$$K_{II}\boldsymbol{x}_I^0 + K_{IB}\boldsymbol{x}_B = \mathbf{0}. \tag{6.2.18}$$

Proof For any $\boldsymbol{x}_I$

$$\boldsymbol{x}^T K \boldsymbol{x} = [\boldsymbol{x}_I^T, \boldsymbol{x}_B^T] \begin{bmatrix} K_{II} & K_{IB} \\ K_{IB}^T & K_{BB} \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_I \\ \boldsymbol{x}_B \end{bmatrix} = \boldsymbol{x}_I^T K_{II} \boldsymbol{x}_I + 2\boldsymbol{x}_B^T K_{IB}^T \boldsymbol{x}_I + \boldsymbol{x}_B^T K_{BB} \boldsymbol{x}_B$$

Hence by Lemma 6.2.2 with $A = K_{II}$, and $\boldsymbol{b} = K_{IB}\boldsymbol{x}_B$

$$\min_{\boldsymbol{x}_I} \boldsymbol{x}^T K \boldsymbol{x} = -\boldsymbol{x}_I^{0\,T} K_{II} \boldsymbol{x}_I^0 + \boldsymbol{x}_B^T K_{BB} \boldsymbol{x}_B,$$

where $K_{II}\boldsymbol{x}_I^0 + K_{IB}\boldsymbol{x}_B = \mathbf{0}$. Therefore

$$\begin{aligned} \min_{\boldsymbol{x}_I} \boldsymbol{x}^T K \boldsymbol{x} &= -(-K_{II}^{-1} K_{IB} \boldsymbol{x}_B)^T K_{II} (-K_{II}^{-1} K_{IB} \boldsymbol{x}_B) + \boldsymbol{x}_B^T K_{BB} \boldsymbol{x}_B, \\ &= \boldsymbol{x}_B^T (K_{BB} - K_{IB}^T K_{II}^{-1} K_{IB}) \boldsymbol{x}_B, \\ &= \boldsymbol{x}_B^T S \boldsymbol{x}_B. \end{aligned}$$

■

Remark 6.2.1 Even though the underlying PDE may be much more general than Laplace's equation the vector $\boldsymbol{x}_I^0$ determined by (6.2.18) is usually called the *discrete harmonic extension* of $\boldsymbol{x}_B$.

Observe that Lemma 6.2.3 implies that since K is SPD then so is S. If we now denote the maximum eigenvalue of a matrix A by $\lambda_{max}(A)$ and similarly

the minimum eigenvalue by $\lambda_{min}(A)$ then the previous two lemmas give us the following result.

LEMMA 6.2.4 $\lambda_{min}(K) \leq \lambda_{min}(S) \leq \lambda_{max}(S) \leq \lambda_{max}(K)$

Proof The middle inequality is obvious. Let $\boldsymbol{x}_B$ be an eigenvector of S corresponding to its maximum eigenvalue, $\lambda_{max}(S)$, and let $\boldsymbol{x}_B^T\boldsymbol{x}_B = 1$. Then

$$\begin{aligned} \lambda_{max}(S) &= \boldsymbol{x}_B^T S \boldsymbol{x}_B, \\ &\leq [\boldsymbol{x}_I^T, \boldsymbol{x}_B^T] K \begin{bmatrix} \boldsymbol{x}_I \\ \boldsymbol{x}_B \end{bmatrix} \qquad \text{for all } \boldsymbol{x}_I, \end{aligned}$$

by Lemma 6.2.3. Hence in particular

$$\lambda_{max}(S) \leq [\mathbf{0}^T, \boldsymbol{x}_B^T] K \begin{bmatrix} \mathbf{0} \\ \boldsymbol{x}_B \end{bmatrix},$$

and since

$$[\mathbf{0}^T, \boldsymbol{x}_B^T] \begin{bmatrix} \mathbf{0} \\ \boldsymbol{x}_B \end{bmatrix} = 1,$$

we have

$$\lambda_{max}(S) \leq \lambda_{max}(K).$$

Conversely choose $\boldsymbol{x}_B$ to be an eigenvector of S corresponding to its minimum eigenvalue, $\lambda_{min}(S)$, and let $\boldsymbol{x}_B^T\boldsymbol{x}_B = 1$. Now

$$\begin{aligned} \lambda_{min}(S) &= \boldsymbol{x}_B^T S \boldsymbol{x}_B, \\ &= [\boldsymbol{x}_I^{0\,T}, \boldsymbol{x}_B^T] K \begin{bmatrix} \boldsymbol{x}_I^0 \\ \boldsymbol{x}_B \end{bmatrix}, \end{aligned}$$

where $K_{II}\boldsymbol{x}_I^0 + K_{IB}\boldsymbol{x}_B = \mathbf{0}$. Hence

$$\lambda_{min}(S) \geq \frac{[\boldsymbol{x}_I^{0\,T}, \boldsymbol{x}_B^T] K \begin{bmatrix} \boldsymbol{x}_I^0 \\ \boldsymbol{x}_B \end{bmatrix}}{[\boldsymbol{x}_I^{0\,T}\boldsymbol{x}_I^0 + 1]}$$

$$= \frac{[\boldsymbol{x}_I^{0\,T}, \boldsymbol{x}_B^T] K \begin{bmatrix} \boldsymbol{x}_I^0 \\ \boldsymbol{x}_B \end{bmatrix}}{[\boldsymbol{x}_I^{0\,T}, \boldsymbol{x}_B^T] \begin{bmatrix} \boldsymbol{x}_I^0 \\ \boldsymbol{x}_B \end{bmatrix}}$$

$$\geq \lambda_{min}(K). \qquad \blacksquare$$

It immediately follows that

$$\kappa(S) \leq \kappa(K)$$

In the next section we will show that, in some special circumstances, S can be spectacularly well conditioned compared to K. However, in general the condition number of S has been observed to grow with the number of degrees of freedom in $[\Pi_h]$ and with the jumps of the coefficients across substructure boundaries.

For convenience we suppress the subscripts in (6.2.14) and write it as

$$S\boldsymbol{x} = \boldsymbol{c}, \tag{6.2.19}$$

which is to be solved for $\boldsymbol{x} \in [\Pi_h]$. This is done by CGM. The kernel of this algorithm (the matrix-vector products $S\boldsymbol{z}$ for any $\boldsymbol{z} \in [\Pi_h]$) are then naturally parallelised by the domain decomposition: To compute them simply break $\boldsymbol{z}$ up into parts $\boldsymbol{z}^{(i)}$, multiply these by the corresponding $S^{(i)}$ and then add together the contributions from neighbouring substructures across each substructure edge. S itself is never assembled. On massively parallel machines with thousands of processors it is then natural to assign a substructure to each processor so that this process is as parallel as possible. In this context then S can still be a very large matrix. For example suppose Ω is a square is divided up into $m \times m$ equal substructures, each of which is divided up by a uniform grid with $n \times n$ interior nodes, then the number of degrees of freedom associated with K is $O((mn)^2)$, whereas the number associated with S is still $O((m^2)n)$. When m is say $O(10^2)$

and n is say $O(10^1)$ both problems involving K and S are "large". By the remarks at the end of the last paragraph it then becomes imperative to find a good preconditioner for S. This is where the domain decomposition approach is at its most powerful: Not only does it yield fast matrix-vector products, but also allows us to define massively parallel preconditioners in a very natural way. We solve (6.2.19) by the preconditioned conjugate gradient method (which is introduced in Section 6.4) to get $\boldsymbol{x}_B^{(i)}$ for each i and then retrieve $\boldsymbol{x}_I^{(i)}$ from (6.2.12). By Theorem 6.2.1, this algorithm gives the solution to (6.2.9).

Before introducing the preconditioned conjugate gradient method we will investigate the aforementioned special circumstances in which S is far better conditioned than K.

6.3 Special cases

In the following two examples we will show that in special cases the condition number of S is independent of the coefficient function a in (6.1.1). The first example was inspired by a comment in [7, page 1104], while the second example was investigated following some unexpected numerical results.

6.3.1 Two equal subdomains

Consider the problem

$$\begin{aligned} -\nabla.(a\nabla u) &= f \quad \text{in} \quad \Omega = [0,1] \times [0,1], \\ u &= 0 \quad \text{on} \quad \partial\Omega, \end{aligned} \tag{6.3.20}$$

where

$$a = \begin{cases} k_1, & y > 0.5, \\ k_2, & y \leq 0.5, \end{cases}$$

where k_1 and k_2 are positive constants. Firstly we divide the domain into two equal substructures, namely $\Omega^{(1)} = (0,1) \times (0.5,1)$, $\Omega^{(2)} = (0,1) \times (0,0.5)$.

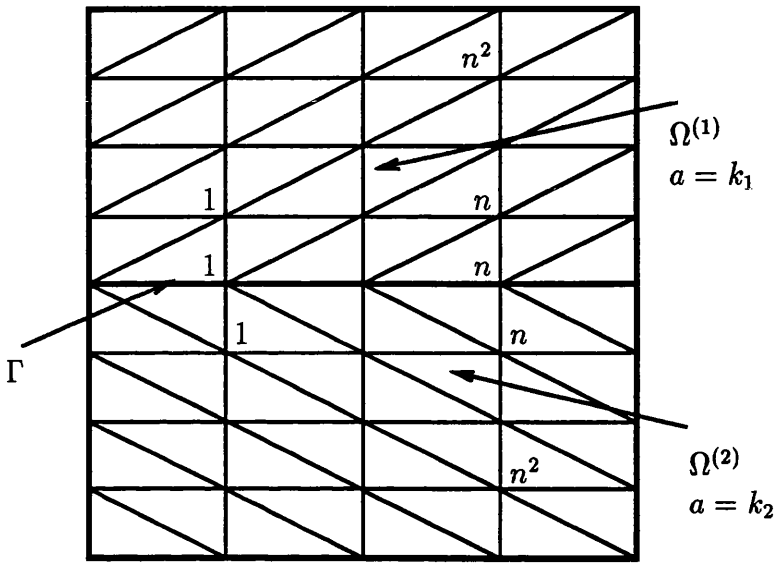


Figure 6.1: Mesh and node numbering for two equal subdomains.

We then discretise the problem by the finite element method over the uniform grid shown in Figure 6.1 with (for convenience) $n \times (2n+1)$ unknowns. Let us write $\boldsymbol{x}^{(1)}$ for the unknowns interior to $\Omega^{(1)}$, $\boldsymbol{x}^{(2)}$ for the unknowns interior to $\Omega^{(2)}$, $\boldsymbol{x}^{(3)}$ for the unknowns on $\Gamma := \overline{\Omega^{(1)}} \bigcap \overline{\Omega^{(2)}}$ and label the unknowns as in Figure 6.1. Then, exploiting the symmetry in the mesh, the resulting linear system, $K\boldsymbol{x} = \boldsymbol{b}$, in the case $k_1 = k_2 = 1$ can be expressed as

$$\begin{bmatrix} K_{II} & 0 & K_{IB} \\ 0 & K_{II} & K_{IB} \\ K_{IB}^T & K_{IB}^T & K_{BB} \end{bmatrix} \begin{bmatrix} \boldsymbol{x}^{(1)} \\ \boldsymbol{x}^{(2)} \\ \boldsymbol{x}^{(3)} \end{bmatrix} = \begin{bmatrix} \boldsymbol{b}^{(1)} \\ \boldsymbol{b}^{(2)} \\ \boldsymbol{b}^{(3)} \end{bmatrix}. \tag{6.3.21}$$

Using this notation, we can then write the general case of arbitrary k_1, k_2 as

$$\begin{bmatrix} k_1 K_{II} & 0 & k_1 K_{IB} \\ 0 & k_2 K_{II} & k_2 K_{IB} \\ k_1 K_{IB}^T & k_2 K_{IB}^T & \frac{(k_1+k_2)}{2} K_{BB} \end{bmatrix} \begin{bmatrix} \boldsymbol{x}^{(1)} \\ \boldsymbol{x}^{(2)} \\ \boldsymbol{x}^{(3)} \end{bmatrix} = \begin{bmatrix} \boldsymbol{b}^{(1)} \\ \boldsymbol{b}^{(2)} \\ \boldsymbol{b}^{(3)} \end{bmatrix}. \tag{6.3.22}$$

As above, let S be the Schur complement obtained by eliminating the unknowns $\boldsymbol{x}^{(1)}$, $\boldsymbol{x}^{(2)}$ from this system. The MATLAB results in Table 6.1 then show that, for fixed n, the condition number of K increases as the ratio $\max\{k_1, k_2\}/\min\{k_1, k_2\}$ increases (in fact it seems to increase linearly with this ratio), while the condition number of S, perhaps surprisingly, remains constant.

k_1	k_2	$\kappa(K)$	$\kappa(S)$
1	1	13.9282	4.5788
1	10	58.7365	4.5788
1	100	545.210	4.5788
1	1000	5417.88	4.5788

Table 6.1: $n = 5$.

In order to explain this behaviour, recall the definition of the Schur complement matrix S to obtain

$$\begin{aligned} S &= \frac{(k_1+k_2)}{2}K_{BB} - [k_1K_{IB}^T, k_2K_{IB}^T]\begin{bmatrix} (k_1K_{II})^{-1} & 0 \\ 0 & (k_2K_{II})^{-1} \end{bmatrix}\begin{bmatrix} k_1K_{IB} \\ k_2K_{IB} \end{bmatrix}, \\ &= \frac{(k_1+k_2)}{2}K_{BB} - (k_1+k_2)K_{IB}^TK_{II}^{-1}K_{IB}, \\ &= \frac{(k_1+k_2)}{2}(K_{BB} - 2K_{IB}^TK_{II}^{-1}K_{IB}) = \frac{(k_1+k_2)}{2}\hat{S}, \end{aligned}$$

where $\hat{S}$ is the Schur complement matrix obtained by applying the same process to (6.3.21). Therefore λ is an eigenvalue of $\hat{S}$ with associated eigenvector $\boldsymbol{y}$ if and only if $(k_1+k_2)\lambda/2$ is an eigenvalue of S with corresponding eigenvector $\boldsymbol{y}$. Hence

$$\kappa(S) = \frac{\lambda_{max}(S)}{\lambda_{min}(S)} = \frac{(k_1+k_2)\lambda_{max}(\hat{S})/2}{(k_1+k_2)\lambda_{min}(\hat{S})/2} = \kappa(\hat{S}).$$

Therefore the condition number of the Schur complement matrix for the problem with arbitrarily chosen $k_1 \neq k_2$ is independent of k_1 and k_2.

6.3.2 Four equal subdomains – checkerboard configuration.

We now consider the problem (6.3.20) with Ω divided into four equal substructures. These are

$$\begin{aligned}
\Omega^{(SW)} &= (0, 0.5) \times (0, 0.5), \\
\Omega^{(NW)} &= (0, 0.5) \times (0.5, 1), \\
\Omega^{(NE)} &= (0.5, 1) \times (0.5, 1), \\
\Omega^{(SE)} &= (0.5, 1) \times (0, 0.5).
\end{aligned}$$

Then let the coefficient function a be given by

$$a(\boldsymbol{x}) = \begin{cases} k_1, & \boldsymbol{x} \in \Omega^{(NW)} \quad \text{or} \quad \boldsymbol{x} \in \Omega^{(SE)}, \\ k_2, & \boldsymbol{x} \in \Omega^{(NE)} \quad \text{or} \quad \boldsymbol{x} \in \Omega^{(SW)}, \end{cases}$$

where k_1 and k_2 are positive constants.

The problem is then discretised by the finite element method with respect to the mesh containing $n \times n$ nodes in each subdomain as shown in Figure 6.2. The nodes are numbered as shown in Figure 6.2 and then the vector of nodal unknown values $\boldsymbol{x}$, can be partitioned as

$$\boldsymbol{x} = (\boldsymbol{x}^{NW^T}, \boldsymbol{x}^{NE^T}, \boldsymbol{x}^{SE^T}, \boldsymbol{x}^{SW^T}, \boldsymbol{x}^{N^T}, \boldsymbol{x}^{E^T}, x^C, \boldsymbol{x}^{W^T}, \boldsymbol{x}^{S^T})^T.$$

For example, $\boldsymbol{x}^{NW}$ represents the nodal values interior to $\Omega^{(NW)}$, $\boldsymbol{x}^N$ represents nodal values interior to the interface of $\Omega^{(NW)}$ with $\Omega^{(NE)}$ and x^C is the nodal value at the intersection of all four subdomains. Then in the case $k_1 = k_2 = 1$ we may represent the problem stiffness matrix, K, as

$$K = \begin{bmatrix} \tilde{K}_{II} & \tilde{K}_{IB} \\ \tilde{K}_{IB}^T & \tilde{K}_{BB} \end{bmatrix}.$$

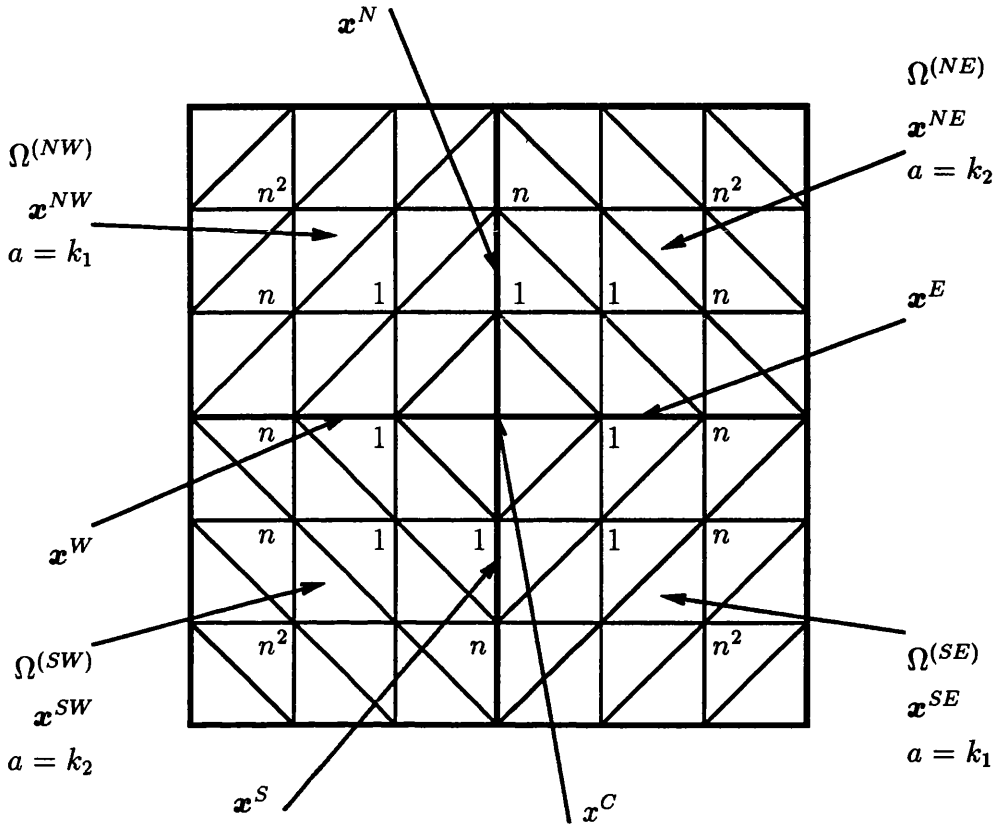


Figure 6.2: Mesh and node numbering for four subdomain problem.

Here, again exploiting the symmetry in the mesh, we have the following block structure for the components of K.

$$\tilde{K}_{II} = \begin{bmatrix} K_{II} & 0 & 0 & 0 \\ 0 & K_{II} & 0 & 0 \\ 0 & 0 & K_{II} & 0 \\ 0 & 0 & 0 & K_{II} \end{bmatrix},$$

$$\tilde{K}_{IB} = \begin{bmatrix} K_{IB}^1 & 0 & 0 & K_{IB}^2 & 0 \\ K_{IB}^1 & K_{IB}^2 & 0 & 0 & 0 \\ 0 & K_{IB}^2 & 0 & 0 & K_{IB}^1 \\ 0 & 0 & 0 & K_{IB}^2 & K_{IB}^1 \end{bmatrix},$$

$$\tilde{K}_{BB} = \begin{bmatrix} K_{BB}^1 & 0 & K_{BB}^3 & 0 & 0 \\ 0 & K_{BB}^1 & K_{BB}^3 & 0 & 0 \\ K_{BB}^{3^T} & K_{BB}^{3^T} & K_{BB}^2 & K_{BB}^{3^T} & K_{BB}^{3^T} \\ 0 & 0 & K_{BB}^3 & K_{BB}^1 & 0 \\ 0 & 0 & K_{BB}^3 & 0 & K_{BB}^1 \end{bmatrix}.$$

As a guide to our choice of notation here let us remark, for example, that K_{II} represents the coupling between the interior nodes of any of the four subdomains. As another example, K_{BB}^3 represents the coupling between the interior nodes of any of the four interior substructure edges with the centre node.

Analogously, in the general case $k_1 \neq k_2$ we have

$$K = \begin{bmatrix} \hat{K}_{II} & \hat{K}_{IB} \\ \hat{K}_{IB}^T & \hat{K}_{BB} \end{bmatrix} \tag{6.3.23}$$

where

$$\hat{K}_{II} = \begin{bmatrix} k_1 K_{II} & 0 & 0 & 0 \\ 0 & k_2 K_{II} & 0 & 0 \\ 0 & 0 & k_1 K_{II} & 0 \\ 0 & 0 & 0 & k_2 K_{II} \end{bmatrix},$$

$$\hat{K}_{IB} = \begin{bmatrix} k_1 K_{IB}^1 & 0 & 0 & k_1 K_{IB}^2 & 0 \\ k_2 K_{IB}^1 & k_2 K_{IB}^2 & 0 & 0 & 0 \\ 0 & k_1 K_{IB}^2 & 0 & 0 & k_1 K_{IB}^1 \\ 0 & 0 & 0 & k_2 K_{IB}^2 & k_2 K_{IB}^1 \end{bmatrix},$$

$$\hat{K}_{BB} = \frac{(k_1 + k_2)}{2} \begin{bmatrix} K_{BB}^1 & 0 & K_{BB}^3 & 0 & 0 \\ 0 & K_{BB}^1 & K_{BB}^3 & 0 & 0 \\ K_{BB}^{3^T} & K_{BB}^{3^T} & K_{BB}^2 & K_{BB}^{3^T} & K_{BB}^{3^T} \\ 0 & 0 & K_{BB}^3 & K_{BB}^1 & 0 \\ 0 & 0 & K_{BB}^3 & 0 & K_{BB}^1 \end{bmatrix}.$$

Now let S be the Schur complement matrix obtained by eliminating the variables $\boldsymbol{x}^{NW}$, $\boldsymbol{x}^{NE}$, $\boldsymbol{x}^{SE}$ and $\boldsymbol{x}^{SW}$ from this system. The MATLAB results given

in Table 6.2 show that, for fixed n, the condition number of K increases linearly as the ratio $\max\{k_1, k_2\}/\min\{k_1, k_2\}$ increases while the condition number of S still remains constant.

k_1	k_2	$\kappa(K)$	$\kappa(S)$
1	1	13.928	6.8239
1	10	36.470	6.8239
1	100	317.67	6.8239
1	1000	3148.7	6.8239

Table 6.2: $n = 2$.

In order to prove that the condition number of S remains independent of k_1 and k_2 we must investigate the eigenvalues of S. Firstly, in order to expedite the following analysis, we make the definitions

$$\begin{aligned} A &= \frac{1}{2}K^1_{BB} - K^{1^T}_{IB}K^{-1}_{II}K^1_{IB}, \\ B &= \frac{1}{2}K^1_{BB} - K^{2^T}_{IB}K^{-1}_{II}K^2_{IB}, \\ C &= \frac{1}{2}K^2_{BB}, \\ D &= K^{1^T}_{IB}K^{-1}_{II}K^2_{IB}, \\ E &= \frac{1}{2}K^3_{BB}. \end{aligned}$$

Then recalling that S is obtained simply by block Gaussian elimination, some tedious matrix manipulation shows that S $(= \hat{K}_{BB} - \hat{K}^T_{IB}\hat{K}^{-1}_{II}\hat{K}_{IB})$ may be written in the form

$$S = \begin{bmatrix} (k_1+k_2)A & -k_2D & (k_1+k_2)E & -k_1D & 0 \\ -k_2D^T & (k_1+k_2)B & (k_1+k_2)E & 0 & -k_1D^T \\ (k_1+k_2)E^T & (k_1+k_2)E^T & (k_1+k_2)C & (k_1+k_2)E^T & (k_1+k_2)E^T \\ -k_1D^T & 0 & (k_1+k_2)E & (k_1+k_2)B & -k_2D^T \\ 0 & -k_1D & (k_1+k_2)E & -k_2D & (k_1+k_2)A \end{bmatrix}. \tag{6.3.24}$$

Recall that S is SPD and hence has real, positive eigenvalues. Also note that S is *block persymmetric.* That is, generalising the definition in [25], if we define M as

$$M = \begin{bmatrix} 0 & 0 & 0 & 0 & I_n \\ 0 & 0 & 0 & I_n & 0 \\ 0 & 0 & I_1 & 0 & 0 \\ 0 & I_n & 0 & 0 & 0 \\ I_n & 0 & 0 & 0 & 0 \end{bmatrix}, \tag{6.3.25}$$

where I_m is the $m \times m$ identity matrix, then M is symmetric, $M^T M = I_{4n+1}$ and S has the property

$$S = MS^T M. \tag{6.3.26}$$

A persymmetric matrix is symmetric about its northeast–southwest diagonal. Here we have a matrix, S, which is both symmetric and *block* persymmetric. Persymmetric matrices are briefly discussed in [25].

With the aim of showing that the maximum and minimum eigenvalues of S, denoted $\lambda_{max}(S)$ and $\lambda_{min}(S)$ respectively, are independent of k_1 and k_2, we consider the form of the eigenvectors of S. Using the following results we will then be able to obtain expressions for $\lambda_{max}(S), \lambda_{min}(S)$ via the Rayleigh Quotient Theorem. We characterise the eigenvectors of S using the following two subspaces of $\mathbb{R}^{4n+1}$,

$$P = \left\{ \boldsymbol{x} \in \mathbb{R}^{4n+1} : \boldsymbol{x} = M\boldsymbol{x} \right\}, \tag{6.3.27}$$

$$Q = \left\{ \boldsymbol{x} \in \mathbb{R}^{4n+1} : \boldsymbol{x} = -M\boldsymbol{x} \right\}. \tag{6.3.28}$$

Then we have the following two trivial results

LEMMA 6.3.1

$$\mathbb{R}^{4n+1} = P \oplus Q,$$

$$P \perp Q.$$

Proof Let $\boldsymbol{x} \in P \bigcap Q$. Then by (6.3.27), (6.3.28)

$$\boldsymbol{x} = M^2\boldsymbol{x} = M\boldsymbol{x} = -\boldsymbol{x},$$

and hence $\boldsymbol{x} = \mathbf{0}$.

Given any $\boldsymbol{x} = (\boldsymbol{x}^{N^T}, \boldsymbol{x}^{E^T}, x^C, \boldsymbol{x}^{W^T}, \boldsymbol{x}^{S^T})^T \in \mathbb{R}^{4n+1}$, consider

$$\boldsymbol{p} := \frac{1}{2}\begin{pmatrix} \boldsymbol{x}^N + \boldsymbol{x}^S \\ \boldsymbol{x}^E + \boldsymbol{x}^W \\ 2x^C \\ \boldsymbol{x}^W + \boldsymbol{x}^E \\ \boldsymbol{x}^S + \boldsymbol{x}^N \end{pmatrix} \in P, \quad \boldsymbol{q} := \frac{1}{2}\begin{pmatrix} \boldsymbol{x}^N - \boldsymbol{x}^S \\ \boldsymbol{x}^E - \boldsymbol{x}^W \\ 0 \\ \boldsymbol{x}^W - \boldsymbol{x}^E \\ \boldsymbol{x}^S - \boldsymbol{x}^N \end{pmatrix} \in Q.$$

Then $\boldsymbol{x} = \boldsymbol{p} + \boldsymbol{q}$. Hence $\mathbb{R}^{4n+1} = P \oplus Q$.

Furthermore, given $\boldsymbol{p} \in P$, $\boldsymbol{q} \in Q$, then by (6.3.27), (6.3.28) and the properties of M we have

$$\boldsymbol{p}^T\boldsymbol{q} = -(M\boldsymbol{p})^T M\boldsymbol{q} = -\boldsymbol{p}^T M^T M\boldsymbol{q} = -\boldsymbol{p}^T\boldsymbol{q}.$$

Hence $\boldsymbol{p}^T\boldsymbol{q} = 0$ as required. ■

LEMMA 6.3.2

$$S : P \longrightarrow P, \tag{6.3.29}$$

$$S : Q \longrightarrow Q. \tag{6.3.30}$$

Proof We will show (6.3.29) only. (6.3.30) is analogous. Consider $\boldsymbol{x} \in P$, then using (6.3.26) and the symmetry of S,

$$S\boldsymbol{x} = MSM\boldsymbol{x} = MS\boldsymbol{x},$$

and hence $S\boldsymbol{x} \in P$. ■

The preceding two lemmas help furnish us with the following result.

LEMMA 6.3.3 *Any eigenvector $\boldsymbol{x}$ of S can be uniquely expressed as*

$$\boldsymbol{x} = \boldsymbol{p} + \boldsymbol{q}, \quad \textit{where} \quad \boldsymbol{p} \in P \textit{ and } \boldsymbol{q} \in Q.$$

Furthermore, if $\boldsymbol{x}$ has corresponding eigenvalue denoted by λ, then $S\boldsymbol{p} = \lambda\boldsymbol{p}$ and $S\boldsymbol{q} = \lambda\boldsymbol{q}$.

Proof It is immediate from Lemma 6.3.1 that *any* $\boldsymbol{x} \in I\!R^{4n+1}$ can be uniquely expressed as

$$\boldsymbol{x} = \boldsymbol{p} + \boldsymbol{q}, \tag{6.3.31}$$

where $\boldsymbol{p} \in P$ and $\boldsymbol{q} \in Q$. Hence for any eigenvector $\boldsymbol{x}$ with corresponding eigenvalue λ we have unique $\boldsymbol{p} \in P$, $\boldsymbol{q} \in Q$ such that (6.3.31) holds and

$$S\boldsymbol{p} + S\boldsymbol{q} = S\boldsymbol{x} = \lambda\boldsymbol{x} = \lambda\boldsymbol{p} + \lambda\boldsymbol{q}.$$

Then, using Lemmas 6.3.1 and 6.3.2, it follows that

$$S\boldsymbol{p} = \lambda\boldsymbol{p}, \quad \text{and} \quad S\boldsymbol{q} = \lambda\boldsymbol{q},$$

as required. ■

This knowledge about the form of the eigenvectors of S will allow us to study the Rayleigh Quotient of S and, given an eigenvector of S, produce an explicit representation of the associated eigenvalue. First recall that, given any normalised eigenvector of S, $\boldsymbol{x} = (\boldsymbol{x}^{N^T}, \boldsymbol{x}^{E^T}, x^C, \boldsymbol{x}^{W^T}, \boldsymbol{x}^{S^T})^T$ such that $\boldsymbol{x}^T\boldsymbol{x} = 1$, then the associated eigenvalue, $\lambda = \boldsymbol{x}^T S\boldsymbol{x}$.

Now for any $\boldsymbol{x} \in I\!R^{4n+1}$

$$\begin{aligned}\boldsymbol{x}^T S\boldsymbol{x} \;=\; & (k_1+k_2)(\boldsymbol{x}^{N^T}A\boldsymbol{x}^N + \boldsymbol{x}^{E^T}B\boldsymbol{x}^E + \boldsymbol{x}^{W^T}B\boldsymbol{x}^W + \boldsymbol{x}^{S^T}A\boldsymbol{x}^S + x^C C x^C \\ & +2\boldsymbol{x}^{N^T}Ex^C + 2\boldsymbol{x}^{E^T}Ex^C + 2\boldsymbol{x}^{W^T}Ex^C + 2\boldsymbol{x}^{S^T}Ex^C) \\ & -2k_2(\boldsymbol{x}^{N^T}D\boldsymbol{x}^E + \boldsymbol{x}^{S^T}D\boldsymbol{x}^W) - 2k_1(\boldsymbol{x}^{N^T}D\boldsymbol{x}^W + \boldsymbol{x}^{S^T}D\boldsymbol{x}^E). \end{aligned} \tag{6.3.32}$$

If $\boldsymbol{x} \in P$ then $\boldsymbol{x}$ takes the form $(\boldsymbol{x}^{N^T}, \boldsymbol{x}^{E^T}, x^C, \boldsymbol{x}^{E^T}, \boldsymbol{x}^{N^T})^T$, i.e. $\boldsymbol{x}^E = \boldsymbol{x}^W, \boldsymbol{x}^N = \boldsymbol{x}^S$, and so

$$\begin{aligned}\boldsymbol{x}^T S\boldsymbol{x} \;=\; & 2(k_1+k_2)(\boldsymbol{x}^{N^T}A\boldsymbol{x}^N + \boldsymbol{x}^{E^T}B\boldsymbol{x}^E + \frac{1}{2}x^C C x^C \\ & +2\boldsymbol{x}^{N^T}Ex^C + 2\boldsymbol{x}^{E^T}Ex^C - 2\boldsymbol{x}^{N^T}D\boldsymbol{x}^E) \end{aligned} \tag{6.3.33}$$

i.e. we can extract a factor of $(k_1 + k_2)$ from the expression for λ. However if $\boldsymbol{x} \in Q$ then $\boldsymbol{x}$ takes the form $(\boldsymbol{x}^{N^T}, \boldsymbol{x}^{E^T}, 0, -\boldsymbol{x}^{E^T}, -\boldsymbol{x}^{N^T})^T$ and

$$\boldsymbol{x}^T S \boldsymbol{x} = 2(k_1 + k_2)(\boldsymbol{x}^{N^T} A \boldsymbol{x}^N + \boldsymbol{x}^{E^T} B \boldsymbol{x}^E) + 4(k_1 - k_2)\boldsymbol{x}^{N^T} D \boldsymbol{x}^E. \qquad (6.3.34)$$

Hence if $k_1 = k_2 = k$ then (6.3.33) and (6.3.34) imply that the condition number of S will be independent of k. This is as expected as in this case S is the Schur complement arising from (6.3.20) with the coefficient function a equal to the constant k. We now concentrate on the case $k_1 \neq k_2$. We would like to show that $\lambda_{min}(S)$ and $\lambda_{max}(S)$ have corresponding eigenvectors, $\boldsymbol{x}_{min}$ and $\boldsymbol{x}_{max}$ respectively, which are elements of P or are such that their component in Q satisfies $\boldsymbol{q}^{N^T} D \boldsymbol{q}^E = 0$. We would then be able to deduce that $\kappa(S)$ is independent of k_1, k_2 when $k_1 \neq k_2$, which is a somewhat more surprising result. The following theorem allows us to do this

THEOREM 6.3.4 *$\lambda_{min} = \lambda_{min}(S)$ has corresponding eigenspace, denoted $N(S - \lambda_{min} I)$, with elements $\boldsymbol{x}_{min}$ which satisfy*

$$\boldsymbol{x}_{min} = \boldsymbol{p} + \boldsymbol{q} \quad \textit{where } \boldsymbol{p} \in P,\ \boldsymbol{q} \in Q \quad \textit{and} \quad \boldsymbol{q}^{N^T} D \boldsymbol{q}^E = 0.$$

Similarly $\lambda_{max} = \lambda_{max}(S)$ has corresponding eigenspace, denoted $N(S - \lambda_{max} I)$, with elements $\boldsymbol{x}_{max}$ which satisfy

$$\boldsymbol{x}_{max} = \boldsymbol{p} + \boldsymbol{q} \quad \textit{where } \boldsymbol{p} \in P,\ \boldsymbol{q} \in Q \quad \textit{and} \quad \boldsymbol{q}^{N^T} D \boldsymbol{q}^E = 0.$$

Proof We shall only give the proof for λ_{min}. The result for λ_{max} is achieved analogously. We assume for a contradiction that there exists a $\boldsymbol{x}_{min} \in N(S - \lambda_{min} I)$ such that

$$\boldsymbol{x}_{min} = \boldsymbol{p} + \boldsymbol{q},$$

where $\boldsymbol{p} \in P$, $\boldsymbol{q} \in Q$ and $\boldsymbol{q}^{N^T} D \boldsymbol{q}^E \neq 0$. Then $\boldsymbol{q} \neq \boldsymbol{0}$ and by Lemma 6.3.3 we know that $\boldsymbol{q} \in N(S - \lambda_{min} I)$. Hence

$$\lambda_{min}(S) = \boldsymbol{q}^T S \boldsymbol{q} / \boldsymbol{q}^T \boldsymbol{q}.$$

However by Lemma 6.3.5 (which immediately follows this proof), we can construct a vector $\boldsymbol{z} \in \mathbb{R}^{4n+1}$ from $\boldsymbol{q}$ such that

$$\frac{\boldsymbol{z}^T S \boldsymbol{z}}{\boldsymbol{z}^T \boldsymbol{z}} < \frac{\boldsymbol{q}^T S \boldsymbol{q}}{\boldsymbol{q}^T \boldsymbol{q}} = \lambda_{min}(S).$$

This is a contradiction and hence $\boldsymbol{q}^{N^T} D \boldsymbol{q}^E = 0$ as required. ■

The following technical lemma was pivotal in the proof of Theorem 6.3.4.

LEMMA 6.3.5 *Given any $\boldsymbol{q} \in Q$ with $\boldsymbol{q}^{N^T} D \boldsymbol{q}^E \neq 0$, we can construct vectors $\boldsymbol{z}_1, \boldsymbol{z}_2 \in \mathbb{R}^{4n+1}$ such that*

$$\frac{\boldsymbol{z}_1^T S \boldsymbol{z}_1}{\boldsymbol{z}_1^T \boldsymbol{z}_1} < \frac{\boldsymbol{q}^T S \boldsymbol{q}}{\boldsymbol{q}^T \boldsymbol{q}} < \frac{\boldsymbol{z}_2^T S \boldsymbol{z}_2}{\boldsymbol{z}_2^T \boldsymbol{z}_2}. \tag{6.3.35}$$

Proof We show the left hand inequality in (6.3.35) only. The right hand inequality is obtained by a similar construction. First recall using (6.3.34), that given any $\boldsymbol{q} \in Q$ we may write

$$\frac{\boldsymbol{q}^T S \boldsymbol{q}}{\boldsymbol{q}^T \boldsymbol{q}} = \frac{2(k_1 + k_2)(\boldsymbol{q}^{N^T} A \boldsymbol{q}^N + \boldsymbol{q}^{E^T} B \boldsymbol{q}^E) + 4(k_1 - k_2) \boldsymbol{q}^{N^T} D \boldsymbol{q}^E}{\boldsymbol{q}^T \boldsymbol{q}}. \tag{6.3.36}$$

If $\boldsymbol{q}^{N^T} D \boldsymbol{q}^E \neq 0$ then we have the following two cases for (6.3.36).

Case 1 Consider the case $\boldsymbol{q}^{N^T} D \boldsymbol{q}^E > 0$ and introduce the vector

$$\boldsymbol{z}_1 = \begin{pmatrix} \sqrt{2}\boldsymbol{q}^N \\ \sqrt{2}\boldsymbol{q}^E \\ 0 \\ 0 \\ 0 \end{pmatrix}.$$

Then we have $\boldsymbol{z}_1^T \boldsymbol{z}_1 = 2(\boldsymbol{q}^{N^T} \boldsymbol{q}^N + \boldsymbol{q}^{E^T} \boldsymbol{q}^E) = \boldsymbol{q}^T \boldsymbol{q}$ and recalling (6.3.32),

$$\begin{aligned} \boldsymbol{z}_1^T S \boldsymbol{z}_1 &= 2(k_1 + k_2)(\boldsymbol{q}^{N^T} A \boldsymbol{q}^N + \boldsymbol{q}^{E^T} B \boldsymbol{q}^E) - 4k_2 \boldsymbol{q}^{N^T} D \boldsymbol{q}^E \\ &< 2(k_1 + k_2)(\boldsymbol{q}^{N^T} A \boldsymbol{q}^N + \boldsymbol{q}^{E^T} B \boldsymbol{q}^E) + 4(k_1 - k_2) \boldsymbol{q}^{N^T} D \boldsymbol{q}^E \\ &= \boldsymbol{q}^T S \boldsymbol{q}, \end{aligned}$$

where the last inequality follows since $k_1 > 0$.

Case 2 Consider the case $\boldsymbol{q}^{N^T} D\boldsymbol{q}^E < 0$, and introduce the vector

$$\boldsymbol{z}_1 = \begin{pmatrix} -\sqrt{2}\boldsymbol{q}^N \\ \mathbf{0} \\ 0 \\ \sqrt{2}\boldsymbol{q}^E \\ \mathbf{0} \end{pmatrix}.$$

Then again $\boldsymbol{z}_1^T \boldsymbol{z}_1 = \boldsymbol{q}^T \boldsymbol{q}$ and using $k_2 > 0$ we have

$$\begin{aligned} \boldsymbol{z}_1^T S \boldsymbol{z}_1 &= 2(k_1 + k_2)(\boldsymbol{q}^{N^T} A \boldsymbol{q}^N + \boldsymbol{q}^{E^T} B \boldsymbol{q}^E) + 4k_1 \boldsymbol{q}^{N^T} D \boldsymbol{q}^E \\ &< 2(k_1 + k_2)(\boldsymbol{q}^{N^T} A \boldsymbol{q}^N + \boldsymbol{q}^{E^T} B \boldsymbol{q}^E) + 4(k_1 - k_2)\boldsymbol{q}^{N^T} D \boldsymbol{q}^E \\ &= \boldsymbol{q}^T S \boldsymbol{q}. \end{aligned}$$

In either case we have constructed a vector $\boldsymbol{z}_1 \in I\!R^{4n+1}$ satisfying the left hand inequality in (6.3.35). ■

COROLLARY 6.3.6 *$\kappa(S)$ is independent of k_1, k_2.*

Proof We have

$$\lambda_{max}(S) = \boldsymbol{x}_{max}^T S \boldsymbol{x}_{max},$$

where $\boldsymbol{x}_{max}^T \boldsymbol{x}_{max} = 1$. By Theorem 6.3.4 we know that $\boldsymbol{x}_{max} = \boldsymbol{p} + \boldsymbol{q}$, with $\boldsymbol{p} \in P$, $\boldsymbol{q} \in Q$ and $\boldsymbol{q}^{N^T} D \boldsymbol{q}^E = 0$. Then

$$\boldsymbol{x}_{max}^T S \boldsymbol{x}_{max} = \boldsymbol{p}^T S \boldsymbol{p} + 2\boldsymbol{p}^T S \boldsymbol{q} + \boldsymbol{q}^T S \boldsymbol{q} = \boldsymbol{p}^T S \boldsymbol{p} + \boldsymbol{q}^T S \boldsymbol{q},$$

using Lemmas 6.3.1 and 6.3.2. Hence, using (6.3.33) and (6.3.34), $\boldsymbol{x}_{max}^T S \boldsymbol{x}_{max}$ is the product of $(k_1 + k_2)$ and the Schur complement matrix for the problem (6.3.20) with $a = k_1 = k_2 = 1$. Since an analogous statement holds for $\lambda_{min}(S)$, the results follows. ■

Remark 6.3.2 At first sight, it may appear that the uniform grids used in both the above examples play a crucial role in our arguments. However this is not

entirely the case. In fact, for the two subdomain case it suffices to have a grid which is symmetric about $y = \frac{1}{2}$, while the grid used in the four subdomain example needs to be symmetric about $y = \frac{1}{2}$ and $x = \frac{1}{2}$. Then the arguments in this section will still hold and hence the Schur complement matrices in either problem will be conditioned independently of k_1 and k_2.

6.3.3 Further numerical examples

It appears that the checkerboard arrangement of the coefficients may provide us with a surprisingly well–conditioned Schur complement matrix as we increase the number of subdomains in the problem, provided each subdomain coincides with a square on the checkerboard. In this subsection we investigate this conjecture numerically.

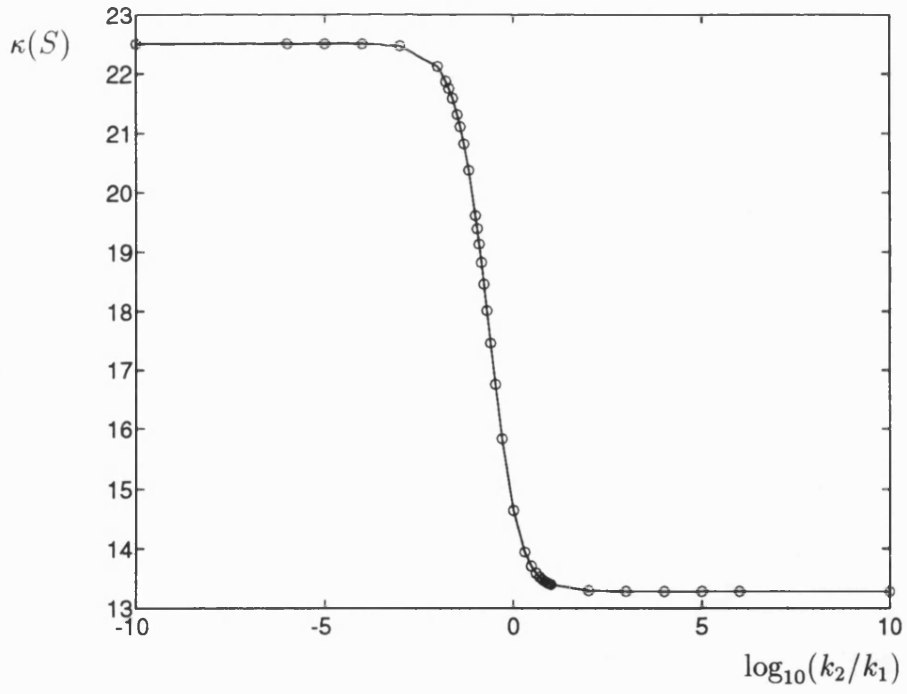


Figure 6.3: 3×3 checkerboard, $n = 2$.

In the first of our numerical examples we have divided the square domain

into 9 equal substructures and considered the PDE problem (6.3.20). Here the function a takes two distinct positive values, k_1, k_2, in a checkerboard pattern starting with k_1 in the top left corner. We triangulate each substructure using a uniform grid with n^2 interior nodes and edges orientated from bottom left to top right. Then we have constructed the Schur complement matrix for this problem with a piece of MATLAB code.

Figure 6.3 shows a plot of $\kappa(S)$ against the log of the ratio k_2/k_1. In this case $\kappa(S)$ is not constant but tends rapidly to constants independent of k_2/k_1 as $\log_{10}(k_2/k_1) \to \pm\infty$. This is not quite the independence that we have seen in our two previous examples. However here we have made no attempt to employ a symmetrical mesh or node numbering (a non–trivial task in the 3×3 case). Hence the slight dependence of the conditioning on k_2/k_1 may be caused by our "non–optimal" mesh. It remains an open question to define a mesh so that $\kappa(S)$ is independent of k_1, k_2 in this example.

$\log_{10}(k_2/k_1)$	CG iterations to convergence
-10	8
-8	8
-4	8
-2	8
0	4
2	8
4	8
8	8
10	8

Table 6.3: 3×3 checkerboard, $n = 2$.

The condition number of S is reflected in the results given in Table 6.3 where we solve the Schur complement system by CGM. We see that the number of iterations required for convergence is essentially independent of k_2/k_1.

For our second experiment we increase the size of the checkerboard still further and observe a similar behaviour. To do this we utilise the power of the parallel computer to solve the Schur complement systems by CGM (see Chapter 7 for implementation details). Here again we discretise with uniform mesh that has n^2 nodes interior to each of the m^2 substructures. Table 6.4 shows again that, for varying sizes of checkerboard, the CG method converges independently of the ratio of the coefficients provided we assign one subdomain to each of the checkerboard squares.

	CG iterations to convergence		
$\log_{10}(k_2/k_1)$	$m = 8$	$m = 16$	$m = 32$
-10	23	43	84
-8	23	43	84
-4	23	43	84
-2	23	43	83
0	18	35	67
2	23	43	83
4	23	43	84
8	23	43	84
10	23	43	84

Table 6.4: $m \times m$ checkerboard, $n = 2$.

Although these checkerboard examples represent rather artificial conditions they are sometimes used for computational tests of parallel algorithms (see, for example, [5], [6]). While it is true that these problems give rise to poorly conditioned stiffness matrices, K, we have seen that, at least with a uniform grid, this conditioning can be made independent of the coefficients if the system is first reduced to the Schur complement problem.

Having said this, we must stress that these are very special circumstances. Table 6.5 shows that even for the two–valued coefficient problem of Section 6.3.1,

if we choose to use 4 equal subdomains then $\kappa(S)$ increases with the ratio k_2/k_1.

k_1	k_2	$\kappa(S)$
1	1	6.8239
1	10	22.626
1	100	213.37
1	1000	2124.7

Table 6.5: Two–valued coefficient, 4 equal sudomains, $n = 2$.

Hence, in general, reducing to the Schur complement system merely reduces the size of the problem we wish to solve by CGM. The conditioning of that system will still be very much dependent on the jumps in the coefficient function. For this reason, Section 6.5 will address the question of finding preconditioners for CGM applied to (6.2.19). First we recall some basic facts about the CGM.

6.4 Preconditioned conjugate gradient method

In many cases which we shall encounter, to attempt to solve (6.2.19) by CGM would require a large number of iterations for the algorithm to reach a satisfactory convergence. That is, to achieve

$$\|\boldsymbol{x} - \boldsymbol{x}^k\|_S < \epsilon \|\boldsymbol{x} - \boldsymbol{x}^1\|_S,$$

would require

$$k \geq \frac{1}{2} \log\left(\frac{2}{\epsilon}\right) \sqrt{\kappa(S)} \; + \; 1.$$

Hence the performance of CGM as an iterative method depends greatly on the condition of S, and, as pointed in Section 6.2, this in general grows linearly with the number of degrees of freedom and with the ratios of the coefficients across subomain boundaries. Therefore we require a method of reducing the condition number of the iteration matrix. We do this via the preconditioned conjugate

gradient method (PCGM), with preconditioner $\hat{S}$, an SPD matrix which has to be chosen. The PCGM is:

- Choose $\boldsymbol{x}^1$.
- Set $\boldsymbol{r}^1 = \boldsymbol{c} - S\boldsymbol{x}^1$.
- Solve

$$\hat{S}\boldsymbol{z}^1 = \boldsymbol{r}^1. \tag{6.4.37}$$

- Put $\boldsymbol{p}^1 = \boldsymbol{z}^1$.
- Then for $k = 1, 2 \ldots$, iterate :

$$\boldsymbol{x}^{k+1} = \boldsymbol{x}^k + \alpha_k \boldsymbol{p}^k, \qquad \boldsymbol{r}^{k+1} = \boldsymbol{r}^k - \alpha_k S\boldsymbol{p}_k,$$

where

$$\alpha_k = (\boldsymbol{z}^k, \boldsymbol{r}^k)/(S\boldsymbol{p}^k, \boldsymbol{p}^k).$$

- Then solve

$$\hat{S}\boldsymbol{z}^{k+1} = \boldsymbol{r}^{k+1}. \tag{6.4.38}$$

- Then put

$$\boldsymbol{p}^{k+1} = \boldsymbol{z}^{k+1} + \beta_k \boldsymbol{p}^k,$$

where

$$\beta_k = (\boldsymbol{z}^{k+1}, \boldsymbol{r}^{k+1})/(\boldsymbol{z}^k, \boldsymbol{r}^k).$$

Then we have the well–known theorem (again see [39]):

THEOREM 6.4.1

(i) $\boldsymbol{r}^k = \boldsymbol{c} - S\boldsymbol{x}^k, \quad k \geq 1.$

(ii) $\| \boldsymbol{x} - \boldsymbol{x}^k \|_S \leq 2 \left[\frac{\sqrt{\kappa(\hat{S}^{-1}S)} - 1}{\sqrt{\kappa(\hat{S}^{-1}S)} + 1} \right]^{k-1} \| \boldsymbol{x} - \boldsymbol{x}^1 \|_S.$

Proof By the algorithm, *(i)* is true for $k = 1$. Suppose *(i)* is true for some $k \geq 1$. Then again by the algorithm,

$$\begin{aligned} \boldsymbol{r}^{k+1} &= \boldsymbol{r}^k - S(\alpha_k \boldsymbol{p}^k) = \boldsymbol{r}^k - S(\boldsymbol{x}^{k+1} - \boldsymbol{x}^k) \\ &= \boldsymbol{c} - S\boldsymbol{x}^k - S\boldsymbol{x}^{k+1} + S\boldsymbol{x}^k = \boldsymbol{c} - S\boldsymbol{x}^{k+1}, \end{aligned}$$

so *(i)* holds for $k+1$, and hence for all k by induction.

Since $\hat{S}$ is chosen to be SPD, we have $\hat{S} = E^T E$ with E nonsingular. By defining $\overline{\boldsymbol{x}}^k = E\boldsymbol{x}^k, \overline{\boldsymbol{p}}^k = E\boldsymbol{p}^k, \overline{\boldsymbol{r}}^k = E^{-T}\boldsymbol{r}^k$, it follows that $\overline{\boldsymbol{x}}^k, \overline{\boldsymbol{r}}^k, \overline{\boldsymbol{p}}^k$ are the iterates of the standard conjugate gradient algorithm applied to the matrix system

$$\overline{S}\overline{\boldsymbol{x}} = \overline{\boldsymbol{c}},$$

where $\overline{S} = E^{-T}SE^{-1}$, and $\overline{\boldsymbol{c}} = E^{-T}\boldsymbol{c}$. This system has solution $\overline{\boldsymbol{x}} = E\boldsymbol{x}$. Hence by (6.2.15),

$$\| \overline{\boldsymbol{x}} - \overline{\boldsymbol{x}}^k \|_{\overline{S}} \leq 2 \left[\frac{\sqrt{\kappa(\overline{S})} - 1}{\sqrt{\kappa(\overline{S})} + 1} \right]^{k-1} \| \overline{\boldsymbol{x}} - \overline{\boldsymbol{x}}^1 \|_{\overline{S}} . \tag{6.4.39}$$

But

$$\| E\boldsymbol{z} \|_{\overline{S}}^2 = (E^{-T}SE^{-1}E\boldsymbol{z}, E\boldsymbol{z}) = (S\boldsymbol{z}, \boldsymbol{z}) = \| \boldsymbol{z} \|_S^2,$$

and the eigenvalues of $\overline{S}$ are the same as the eigenvalues of $E^{-1}\overline{S}E = \hat{S}^{-1}S$. Hence (6.4.39) implies the proof of *(ii)*. ■

It is clear that the rate of convergence of PCGM decreases as $\sqrt{\kappa(\hat{S}^{-1}S)}$ increases. An *optimum* preconditioner is one for which this condition number is independent of the number of degrees of freedom in the finite element discretisation. In practice, we require not only that this condition number stay as small as possible, but also that the preconditioner is relatively easy to invert as we have the solution of the "preconditioning solves", (6.4.37) and (6.4.38), to consider at each step.

6.5 The preconditioners

This work draws heavily on the approach used by Smith, [66]. In order to estimate the condition number of the matrix $\hat{S}^{-1}S$, we prove an inequality of the form

$$\underline{\kappa}\,\boldsymbol{x}^T\hat{S}\boldsymbol{x} \;\leq\; \boldsymbol{x}^T S\boldsymbol{x} \;\leq\; \overline{\kappa}\,\boldsymbol{x}^T\hat{S}\boldsymbol{x}, \quad \boldsymbol{x}\in[\Pi_h], \tag{6.5.40}$$

with constants $\underline{\kappa},\overline{\kappa}$. From (6.5.40) it follows that

$$\kappa(\hat{S}^{-1}S) = \kappa(\hat{S}^{-1/2}S\hat{S}^{-1/2}) \;\leq\; \overline{\kappa}/\underline{\kappa}.$$

We will in fact consider two different preconditioners and an inequality of the type (6.5.40) can be shown for each.

In order to describe our preconditioners for (6.2.19) we require the following notation.

For any substructure edge E containing nodes in Π_h, define the restriction operator $R_E : [\Pi_h] \mapsto [\Pi_h]$ by

$$(R_E\boldsymbol{x})_p = \begin{cases} x_p & \text{if } p \text{ is an } \textit{interior} \text{ node of } E, \\ 0 & \text{otherwise.} \end{cases}$$

Recall that the interface between Dirichlet and Neumann boundary conditions must occur at a coarse grid node and must also be a Dirichlet node. R_E is self adjoint with respect to the usual inner product on $[\Pi_h]$ and the matrix

$$S_E = R_E S R_E^T$$

is the submatrix of S with rows and columns corresponding to interior nodes of E.

Similarly for any coarse grid vertex $V \in \Pi_H$, define $R_V : [\Pi_h] \mapsto [\Pi_h]$ by

$$(R_V\boldsymbol{x})_p = \begin{cases} x_p & \text{if } p = V \text{ or } p \text{ is an interior point of} \\ & \text{a substructure edge containing } V \text{ and} \\ & \text{nodes in } \Pi_h, \\ 0 & \text{otherwise.} \end{cases}$$

Then

$$S_V = R_V S R_V^T$$

just contains rows and columns corresponding to nodes $p \in \Pi_h$ which are "adjacent" to V.

Define the operator $R_H^T : [\Pi_h] \mapsto [\Pi_h]$ by linear interpolation at the nodes of Π_H. That is, for $\boldsymbol{x} \in [\Pi_h]$ take the function on $\Gamma \bigcup \partial\Omega_D$ which has the value x_p at each $p \in \Pi_H$, has the value 0 at $p \in \partial\Omega_D$ and is linear on each edge of Γ. Then, $R_H^T \boldsymbol{x}$ is the restriction of this function to Π_h. R_H^T is not self adjoint, and its adjoint is denoted R_H. Now set

$$S_H = R_H S R_H^T.$$

The preconditioner $\hat{S}$ which is used in (6.4.37) and (6.4.38) is, in its most general form, defined by

$$\hat{S}^{-1} = \sum_{Edges\ E} R_E^T S_E^{-1} R_E + \sum_{Vertices\ V} R_V^T S_V^{-1} R_V + R_H^T S_H^{-1} R_H. \tag{6.5.41}$$

The calculation of $\hat{S}^{-1}\boldsymbol{r}$ for $\boldsymbol{r} \in [\Pi_h]$ thus requires the solution of many local independent subproblems corresponding to edges/vertices, together with a problem of size $|\Pi_H|$ (the "coarse grid problem"). To examine the optimality properties of $\hat{S}$ we need to examine the condition number of

$$\hat{S}^{-1} S =: \sum_E P_E + \sum_V P_V + P_H, \tag{6.5.42}$$

where for $i = E, V$ or H we have

$$P_i = R_i^T S_i^{-1} R_i S.$$

The following lemma shows that P_i is the orthogonal projection onto

$$[\Pi_h]_i := \mathrm{Im}\{R_i^T\} = \{\boldsymbol{x} = R_i^T \boldsymbol{y} : \boldsymbol{y} \in [\Pi_h]\},$$

with respect to the inner product

$$(\boldsymbol{x}, \boldsymbol{y})_S = (S\boldsymbol{x}, \boldsymbol{y}). \tag{6.5.43}$$

LEMMA 6.5.1 *For $i = E, V$ or H and $\boldsymbol{x} \in [\Pi_h]$,*

$$((I - P_i)\boldsymbol{x}, R_i^T \boldsymbol{y})_S = 0, \quad \text{for all } \boldsymbol{y} \in [\Pi_h].$$

Proof Using the fact that $(R_i^T)^2 = R_i^T$ (twice), we have

$$\begin{aligned}(P_i\boldsymbol{x}, R_i^T\boldsymbol{y})_S &= (SR_i^T S_i^{-1} R_i S\boldsymbol{x}, R_i^T\boldsymbol{y}) = (R_i S R_i^T S_i^{-1} R_i S\boldsymbol{x}, R_i^T\boldsymbol{y}) \\ &= (R_i S\boldsymbol{x}, R_i^T\boldsymbol{y}) = (S\boldsymbol{x}, R_i^T\boldsymbol{y}) = (\boldsymbol{x}, R_i^T\boldsymbol{y})_S. \end{aligned}$$ ■

Thus (6.5.42) is a sum of orthogonal projections onto subspaces of $[\Pi_h]$. Such sums are examined in abstract in Section 6.6.1 and this theory is applied in Section 6.6.2 to two particular cases of (6.5.41):

$$\hat{S}^{-1} = \sum_{Vertices\ V} R_V^T S_V^{-1} R_V + R_H^T S_H^{-1} R_H. \tag{6.5.44}$$

$$\hat{S}^{-1} = \sum_{Edges\ E} R_E^T S_E^{-1} R_E + R_H^T S_H^{-1} R_H. \tag{6.5.45}$$

6.6 Convergence theory

6.6.1 Abstract theory of additive Schwarz methods

Let V be an finite-dimensional vector space with inner product $(\cdot,\cdot)_a$ and induced norm $\|\cdot\|_a = (\cdot,\cdot)_a^{1/2}$. If U is a subspace of V and $U^\perp := \{\boldsymbol{v} \in V : (\boldsymbol{v}, \boldsymbol{u})_a = 0, \ \boldsymbol{u} \in U\}$, then $V = U \oplus U^\perp$ (see for instance [35, page 129]) and each $\boldsymbol{v} \in V$ then has a unique representation as $\boldsymbol{v} = \boldsymbol{u} + \boldsymbol{u}^\perp$, where $\boldsymbol{u} \in U$, and $\boldsymbol{u}^\perp \in U^\perp$. The map $\boldsymbol{v} \mapsto P\boldsymbol{v} =: \boldsymbol{u}$ is a well–defined linear transformation on U, called the

orthogonal projection of V onto U (with respect to $(\cdot,\cdot)_a$). It is easily seen that a linear mapping $P : V \to U$ is the orthogonal projection onto U if and only if

$$(P\boldsymbol{v}, \boldsymbol{u})_a = (\boldsymbol{v}, \boldsymbol{u})_a, \ \boldsymbol{v} \in V, \ \boldsymbol{u} \in U.$$

We collect the important properties of P in the following proposition.

PROPOSITION 6.6.1 *Let P be the orthogonal projection of V onto U. Then*

(i) $P^2 = P$,

(ii) $(P\boldsymbol{v}, \boldsymbol{v}')_a = (\boldsymbol{v}, P\boldsymbol{v}')_a, \ \boldsymbol{v}, \ \boldsymbol{v}' \in V$,

(iii) $(\boldsymbol{v}, \boldsymbol{v})_a \geq (P\boldsymbol{v}, \boldsymbol{v})_a \geq 0, \ \boldsymbol{v} \in V$.

Proposition 6.6.1 shows that the linear transformation $P : V \to V$ is positive semidefinite with maximum eigenvalue 1. In fact the only possible eigenvalues are 0 and 1. Lemma 6.6.2 extends this result to the case when P is a sum of projections onto mutually orthogonal subspaces of V. (A set $U_1, \ldots, U_k$ of subspaces of V are called *mutually orthogonal* if for all $i \neq j$ we have $(\boldsymbol{u}_i, \boldsymbol{u}_j)_a = 0, \ \boldsymbol{u}_i \in U_i, \ \boldsymbol{u}_j \in U_j$).

LEMMA 6.6.2 *Let $U_1, \ldots, U_k$ be mutually orthogonal subspaces of V, and, for each i, let P_i be the orthogonal projection of V onto U_i. Then $P := \sum_{i=1}^{k} P_i$ is the orthogonal projection of V onto $U := U_1 \oplus \ldots \oplus U_k$. Consequently*

$$(\boldsymbol{v}, \boldsymbol{v})_a \geq (P\boldsymbol{v}, \boldsymbol{v})_a \geq 0, \quad \boldsymbol{v} \in V.$$

Proof Observe that if $\boldsymbol{u}_i \in U_i$ and if $j \neq i$ then for all $\boldsymbol{v} \in V$, $(P_j \boldsymbol{u}_i, \boldsymbol{v})_a = (\boldsymbol{u}_i, P_j \boldsymbol{v})_a = 0$. Hence $P_j \boldsymbol{u}_i = \boldsymbol{0}$. Now let $\boldsymbol{u} \in U$, i.e. $\boldsymbol{u} = \sum_i \boldsymbol{u}_i$ with $\boldsymbol{u}_i \in U_i$ for each i. Then if $\boldsymbol{v} \in V$, we have

$$(P\boldsymbol{v}, \boldsymbol{u})_a = \sum_j \sum_i (P_j \boldsymbol{v}, \boldsymbol{u}_i)_a = \sum_j \sum_i (\boldsymbol{v}, P_j \boldsymbol{u}_i)_a$$

$$= \sum_j (\boldsymbol{v}, \boldsymbol{u}_j)_a = (\boldsymbol{v}, \boldsymbol{u})_a.$$

Thus P is the orthogonal projection onto U and the inequalities now follow from Proposition 6.6.1. ■

In Lemma 6.6.3 we give the generalisation of Lemma 6.6.2 to the sum of projections onto a sum of subspaces which, although not themselves mutually orthogonal, can be decomposed into subsets of mutually orthogonal subspaces. Lemma 6.6.3 gives an upper bound on the spectrum of P. Lower bounds are given in Lemma 6.6.4.

In Lemmas 6.6.3 and 6.6.4 we suppose that U_i, $i = 1, \cdots, s$ are subspaces of V, that P_i is the orthogonal projection of V onto U_i and we set $P = \sum_{i=1}^{s} P_i$.

LEMMA 6.6.3 *Suppose the subspaces $\{U_j\}$ can be distributed into p subsets $\mathcal{S}_1, \ldots, \mathcal{S}_p$ such that each U_j belongs to one and only one $\mathcal{S}_i$ and such that each $\mathcal{S}_i$ is a mutually orthogonal set of subspaces. Then*

$$p(\boldsymbol{v}, \boldsymbol{v})_a \geq (P\boldsymbol{v}, \boldsymbol{v})_a \geq 0, \quad \boldsymbol{v} \in V.$$

Proof . For each i, define $\mathcal{P}_i = \sum_j P_j$, and $\mathcal{U}_i = \sum_j U_j$ where the sums are over all j such that $U_j \in \mathcal{S}_i$. Then by Lemma 6.6.2,

$$(\boldsymbol{v}, \boldsymbol{v})_a \geq (\mathcal{P}_i \boldsymbol{v}, \boldsymbol{v})_a \geq 0, \quad \boldsymbol{v} \in V.$$

Since $P = \sum_{i=1}^{p} \mathcal{P}_i$, we have

$$p(\boldsymbol{v}, \boldsymbol{v})_a \geq \sum_{i=1}^{p} (\mathcal{P}_i \boldsymbol{v}, \boldsymbol{v})_a = (P\boldsymbol{v}, \boldsymbol{v})_a \geq 0, \quad \boldsymbol{v} \in V,$$

as required. ■

LEMMA 6.6.4 (P.L. Lions' Lemma, see [45] or [19, Lemma 2.1]). *Suppose that each $\boldsymbol{v} \in V$ has a representation $\boldsymbol{v} = \sum_{i=1}^{s} \boldsymbol{u}_i$ with $\boldsymbol{u}_i \in U_i$ for each i, and such that*

$$\sum_{i=1}^{s} (\boldsymbol{u}_i, \boldsymbol{u}_i)_a \leq c_o^2 (\boldsymbol{v}, \boldsymbol{v})_a.$$

Then

$$(P\boldsymbol{v},\ \boldsymbol{v})_a \geq c_0^{-2}(\boldsymbol{v},\ \boldsymbol{v})_a, \qquad \boldsymbol{v} \in V.$$

Proof We have, on utilising Proposition 6.6.1,

$$\|\boldsymbol{v}\|_a^2 = (\boldsymbol{v},\boldsymbol{v})_a = \sum_i (\boldsymbol{v},\boldsymbol{u}_i)_a = \sum_i (P_i\boldsymbol{v},\boldsymbol{u}_i)_a.$$

Hence applying the Cauchy–Schwarz inequality in V and then in $\mathbb{R}^s$, we have

$$\begin{aligned}\|\boldsymbol{v}\|_a^2 \leq \sum_i \|P_i\boldsymbol{v}\|_a\ \|\boldsymbol{u}_i\|_a &\leq \left\{\sum_i \|P_i\boldsymbol{v}\|_a^2\right\}^{1/2} \left\{\sum_i \|\boldsymbol{u}_i\|_a^2\right\}^{1/2} \\ &\leq c_0\|\boldsymbol{v}\|_a \left\{\sum_i \|P_i\boldsymbol{v}\|_a^2\right\}^{1/2},\end{aligned}$$

using the hypothesis. Hence

$$\|\boldsymbol{v}\|_a^2 \leq c_0^2 \sum_i (P_i\boldsymbol{v}, P_i\boldsymbol{v})_a = c_0^2 \sum_i (P_i\boldsymbol{v},\boldsymbol{v})_a = c_0^2 (P\boldsymbol{v},\boldsymbol{v})_a,$$

which implies the result. ■

6.6.2 Properties of the preconditioners

Let us now return to (6.5.41). By (6.5.42) and Lemma 6.5.1, $\hat{S}^{-1}S$ is a sum of orthogonal projections (and hence is symmetric) with respect to $(\cdot,\cdot)_S$. The spectrum of $\hat{S}^{-1}S$ can now be bounded using the abstract theory of Section 6.6.1, with upper and lower bounds obtained from Lemmas 6.6.3 and 6.6.4 respectively. The following two results can be deduced from [19]. (There the original system (6.1.2) and not the Schur complement system (6.2.19) was considered, and some minor modifications are necessary to produce the following results. However the proofs in [66] suggest the necessary modifications.) The upper bounds are relatively trivial.

THEOREM 6.6.5 *With $\hat{S}$ given by (6.5.41),*

$$\lambda_{max}(\hat{S}^{-1}S) \leq C,$$

with C independent of h and H.

Proof First observe that the edge spaces $[\Pi_h]_E$ can be distributed into a number of subsets of mutually orthogonal subspaces of $[\Pi_h]$, and the number of such subsets is bounded independently of h and H. To see this, consider the undirected graph with a node for each edge space and a connection between any two nodes if the edge spaces that those nodes represent are not mutually orthogonal. It is immediate from our construction of the substructures that any edge is not orthogonal to, at most, 6 others. Hence any node in our graph is connected to, at most, 6 others. Now consider colouring the graph in such a way that no two nodes that are connected are the same colour. It follows from the above argument that we need, at most, 7 colours to do this. Now distribute the edge spaces into subsets which contain all the edge spaces of the same colour and we have, at most, 7 subsets which, by definition, contain mutually orthogonal subspaces

Hence, by Lemma 6.6.3 applied in the space $[\Pi_h]$, the sum of the edge projections in (6.5.42) has maximum eigenvalue which is bounded independently of h and H. Similarly the maximum eigenvalue of the sum of the vertex projections in (6.5.42) is also bounded independently of h and H. Since the maximum eigenvalue of P_H is 1, the maximum eigenvalue of $\hat{S}^{-1}S$ is bounded independently of h and H. ■

Lower bounds are somewhat more technical to prove. To apply Lemma 6.6.4 we have to find the smallest number c_0 such that any vector $\boldsymbol{x} \in [\Pi_h]$ can be represented as a sum of vectors in the subspaces $[\Pi_h]_i$, $i = E, V, H$ in such a way that the energy increases at most by a factor of c_0^2. Then the smallest eigenvalue of the sum of projections (6.5.42) is bounded below by c_0^{-2}. The first step in doing this is to recall Lemma 6.2.3, which shows, for any $\boldsymbol{x} \in [\Pi_h]$

$$(\boldsymbol{x}, \boldsymbol{x})_S = \boldsymbol{x}^T S \boldsymbol{x} = \tilde{\boldsymbol{x}}^T K \tilde{\boldsymbol{x}},$$

where $\tilde{\boldsymbol{x}}$ is the discrete harmonic extension of $\boldsymbol{x}$ as defined by Remark 6.2.1.

Case A We first consider the case of the preconditioner defined by (6.5.44). In [19] (and also in [66]) it is shown for this preconditioner how $\tilde{\boldsymbol{x}}$ may be expressed in terms of discrete harmonic extensions of vectors in $[\Pi_h]_i$, $i = V, H$. The corresponding increases in energy proved there together with Lemma 6.6.4 lead to the following result.

THEOREM 6.6.6 *If the sum over the edges is deleted from (6.5.41), then*

$$\lambda_{min}(\hat{S}^{-1}S) \geq C,$$

with C independent of h and H.

This immediately gives us the following corollary.

COROLLARY 6.6.7 *With $\hat{S}$ defined by (6.5.44) we have*

$$\kappa(\hat{S}^{-1}S) \leq C,$$

where C is a constant independent of h, H.

Thus we have shown that if the preconditioner is constructed from solves in vertex spaces together with the coarse grid solve, then it is optimal, in the sense that the eigenvalues of $\hat{S}^{-1}S$ remain bounded with respect to changes in h or H.

Case B We now consider the preconditioner defined by (6.5.45). We now proceed by using a refinement of the arguments in [19], [65] and [66] to show that this preconditioner is "weakly sub–optimal". Moreover our arguments also show that $\kappa(\hat{S}^{-1}S)$ is independent of jumps in the coefficient function a. We show this by using the decomposition of Ω into the substructures $\Omega^{(i)}$. Observe that if $\boldsymbol{x} \in [\Pi_h]$ then

$$\boldsymbol{x}^T S \boldsymbol{x} = \boldsymbol{x}^T \sum_i S^{(i)} \boldsymbol{x}^{(i)} = \sum_i \boldsymbol{x}^{(i)^T} S^{(i)} \boldsymbol{x}^{(i)}, \quad \boldsymbol{x} \in [\Pi_h]. \tag{6.6.46}$$

Then, in the procedures that follow, we may first define a preconditioner $\hat{S}^{(i)}$ for each $S^{(i)}$ and then set

$$\hat{S}\boldsymbol{x} = \sum_i \hat{S}^{(i)} \boldsymbol{x}^{(i)}, \quad \boldsymbol{x} \in [\Pi_h], \tag{6.6.47}$$

where, as before, summation means "extension by zero then summation". Thus

$$\boldsymbol{x}^T \hat{S} \boldsymbol{x} = \sum_i \boldsymbol{x}^{(i)T} \hat{S}^{(i)} \boldsymbol{x}^{(i)}, \quad \boldsymbol{x} \in [\Pi_h], \tag{6.6.48}$$

and so if $\hat{S}^{(i)}$ is a "good" preconditioner for $S^{(i)}$, we expect that $\hat{S}$ will be a "good" preconditioner for S.

We begin by showing how the edge space plus coarse grid preconditioner can be viewed in terms of a change to a hierarchical basis.

Consider any vector $\boldsymbol{x}$ defined at the fine grid nodes on all of $\cup_i \partial\Omega^{(i)}$. We can partition it into $\boldsymbol{x}^T = (\boldsymbol{x}_E^T, \boldsymbol{x}_H^T)$, with $\boldsymbol{x}_E$ containing the values of $\boldsymbol{x}$ at *interior* nodes of substructure edges and $\boldsymbol{x}_H$ containing values at substructure corners (i.e. coarse grid nodes). We can also express $\boldsymbol{x}$ in terms of the *hierarchical basis* obtained simply by taking the standard basis vectors at interior nodes of substructure edges and adding those vectors which are standard basis vectors on the coarse grid nodes and linear between them. If $\boldsymbol{y}$ is the vector of coordinates of $\boldsymbol{x}$ with respect to this new basis, then

$$\begin{bmatrix} \boldsymbol{x}_E \\ \boldsymbol{x}_H \end{bmatrix} = \begin{bmatrix} I & \mathcal{R}_H^T \\ 0 & I \end{bmatrix} \begin{bmatrix} \boldsymbol{y}_E \\ \boldsymbol{y}_H \end{bmatrix}. \tag{6.6.49}$$

The matrix appearing on the right-hand side is the change of basis matrix from standard to hierarchical basis. In fact $\mathcal{R}_H^T \boldsymbol{y}_H$ is the linear interpolant to $\boldsymbol{y}_H$ evaluated at interior nodes of substructure edges. The relationship (6.6.49) is for all $\boldsymbol{x}$ and $\boldsymbol{y}$ defined at the fine grid nodes on $\cup_i \partial\Omega^{(i)}$, but it can also be used for $\boldsymbol{x}$ and $\boldsymbol{y} \in [\Pi_h]$, by simply understanding $\boldsymbol{x}$ and $\boldsymbol{y}$ to be extended by zero at nodes on $\partial\Omega_D$. Analogously, for each i, a nodal vector $\boldsymbol{x}^{(i)}$ on $\partial\Omega^{(i)}$ may be partitioned as $\boldsymbol{x}^{(i)T} = (\boldsymbol{x}_E^{(i)T}, \boldsymbol{x}_H^{(i)T})$, and we have the corresponding *local* change to hierarchical basis:

$$\begin{bmatrix} \boldsymbol{x}_E^{(i)} \\ \boldsymbol{x}_H^{(i)} \end{bmatrix} = \begin{bmatrix} I & \mathcal{R}_H^{(i)T} \\ 0 & I \end{bmatrix} \begin{bmatrix} \boldsymbol{y}_E^{(i)} \\ \boldsymbol{y}_H^{(i)} \end{bmatrix}. \tag{6.6.50}$$

The matrix in (6.6.50) is simply the minor of that in (6.6.49), obtained by selecting only the rows and columns corresponding to fine grid nodes on $\partial\Omega^{(i)}$. With respect to the hierarchical basis, the bilinear form induced by the matrix $S^{(i)}$ is represented by

$$\begin{bmatrix} I & 0 \\ \mathcal{R}_H^{(i)} & I \end{bmatrix} \begin{bmatrix} S_{EE}^{(i)} & S_{EH}^{(i)} \\ S_{EH}^{(i)T} & S_{HH}^{(i)} \end{bmatrix} \begin{bmatrix} I & \mathcal{R}_H^{(i)T} \\ 0 & I \end{bmatrix}. \tag{6.6.51}$$

Since the change to hierarchical basis often produces a better conditioned stiffness matrix (see, for example [74]), (6.6.51) is approximated by discarding the off-diagonal blocks and replacing the block $S_{EE}^{(i)}$ by an approximation $\hat{S}_{EE}^{(i)}$ which neglects coupling between nodes on different edges of $\Omega^{(i)}$. This yields the block diagonal matrix:

$$\begin{bmatrix} \hat{S}_{EE}^{(i)} & 0 \\ 0 & \hat{S}_{HH}^{(i)} \end{bmatrix} := \begin{bmatrix} \hat{S}_{EE}^{(i)} & 0 \\ 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ \mathcal{R}_H^{(i)} & I \end{bmatrix} \begin{bmatrix} S_{EE}^{(i)} & S_{EH}^{(i)} \\ S_{EH}^{(i)T} & S_{HH}^{(i)} \end{bmatrix} \begin{bmatrix} 0 & \mathcal{R}_H^{(i)T} \\ 0 & I \end{bmatrix}, \tag{6.6.52}$$

which contains four independent diagonal blocks, one for the interior nodes of each of the edges of $\Omega^{(i)}$ plus a diagonal block which relates the values at the four corners of $\Omega^{(i)}$. Transforming back to standard basis we obtain the matrix

$$\hat{S}^{(i)} := \begin{bmatrix} I & 0 \\ -\mathcal{R}_H^{(i)} & I \end{bmatrix} \begin{bmatrix} \hat{S}_{EE}^{(i)} & 0 \\ 0 & \hat{S}_{HH}^{(i)} \end{bmatrix} \begin{bmatrix} I & -\mathcal{R}_H^{(i)T} \\ 0 & I \end{bmatrix}, \tag{6.6.53}$$

which we use as a preconditioner for $S^{(i)}$. Then, defining $\hat{S}$ by (6.6.47) yields

$$\hat{S} = \begin{bmatrix} I & 0 \\ -\mathcal{R}_H & I \end{bmatrix} \begin{bmatrix} \hat{S}_{EE} & 0 \\ 0 & \hat{S}_{HH} \end{bmatrix} \begin{bmatrix} I & -\mathcal{R}_H^T \\ 0 & I \end{bmatrix}.$$

Here $\hat{S}_{EE}$ is block diagonal with the blocks containing the restrictions of S to the interiors of each of the edges of the substructures and $\hat{S}_{HH}$ is just a coarse grid approximation to S, using linear interpolation and its adjoint as grid transfer

operators. The inverse of $\hat{S}$ is, explicitly,

$$\hat{S}^{-1} = \begin{bmatrix} \hat{S}_{EE}^{-1} & 0 \\ 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & \mathcal{R}_H^T \\ 0 & I \end{bmatrix} \begin{bmatrix} 0 & 0 \\ 0 & \hat{S}_{HH}^{-1} \end{bmatrix} \begin{bmatrix} 0 & 0 \\ \mathcal{R}_H & I \end{bmatrix}, \tag{6.6.54}$$

which coincides with (6.5.45). Note that

$$\begin{bmatrix} 0 & \mathcal{R}_H^T \\ 0 & I \end{bmatrix} = R_H^T,$$

where R_H^T is the interpolation operator introduced earlier. Therefore each preconditioning step consists of independent local solves at interior nodes of substructure edges plus a global coarse grid solve. By a refinement of the arguments in [19], [65] and [66] we can now prove the following theorem.

THEOREM 6.6.8 *For each i there exists a constant $C^{(i)}$ such that*

$$(C^{(i)})^{-1}(1+\log(H/h))^{-2}\boldsymbol{x}^{(i)T}\hat{S}^{(i)}\boldsymbol{x}^{(i)} \leq \boldsymbol{x}^{(i)T}S^{(i)}\boldsymbol{x}^{(i)} \leq 5\boldsymbol{x}^{(i)T}\hat{S}^{(i)}\boldsymbol{x}^{(i)}, \tag{6.6.55}$$

for all $\boldsymbol{x}^{(i)} \in [\Pi_h^{(i)}]$. The constants $C^{(i)}$ depend only on the restrictions of the coefficient a to the subdomain $\Omega^{(i)}$.

Proof

Case I: If $\partial\Omega^{(i)} \cap \partial\Omega_D \neq \emptyset$, then $S^{(i)}$ and $\hat{S}^{(i)}$ are SPD and

$$\hat{S}^{(i)^{-1}} := \begin{bmatrix} \hat{S}_{EE}^{(i)^{-1}} & 0 \\ 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & \mathcal{R}_H^{(i)T} \\ 0 & I \end{bmatrix} \begin{bmatrix} 0 & 0 \\ 0 & \hat{S}_{HH}^{(i)^{-1}} \end{bmatrix} \begin{bmatrix} 0 & 0 \\ \mathcal{R}_H^{(i)} & I \end{bmatrix}. \tag{6.6.56}$$

Let E denote the interior of any of the edges of $\Gamma^{(i)}$ and define subspaces of $[\Pi_h^{(i)}]$ by

$$\begin{aligned} \mathcal{V}_E^{(i)} &= \{\boldsymbol{x}^{(i)} \in [\Pi_h^{(i)}] : x_p^{(i)} = 0,\ p \notin E\}, \\ \mathcal{V}_H^{(i)} &= \mathrm{Im} \begin{bmatrix} 0 & \mathcal{R}_H^{(i)T} \\ 0 & I \end{bmatrix}. \end{aligned}$$

Let $\mathcal{P}_E^{(i)}, \mathcal{P}_H^{(i)}$ denote the orthogonal projections (with respect to the inner product induced by $S^{(i)}$) of $[\Pi_h^{(i)}]$ onto $\mathcal{V}_E^{(i)}$ and $\mathcal{V}_H^{(i)}$. A short calculation, analogous to that

done in the proof of Lemma 6.5.1, then shows that $\hat{S}^{(i)^{-1}} S^{(i)} = \sum_E \mathcal{P}_E^{(i)} + \mathcal{P}_H^{(i)}$, where the sum is over the edges E of $\Gamma^{(i)}$. The well-known additive Schwarz theory (e.g. [19], [20], [66]) can now be applied. Since $\mathcal{P}_E^{(i)}, \mathcal{P}_H^{(i)}$ are orthogonal projections, the largest eigenvalue of $\hat{S}^{(i)^{-1}} S^{(i)}$ is no more than 5. To bound the smallest eigenvalue we combine the estimate in [19, §5] with the procedure in [66, Theorem 4.1] (which transforms energies induced by $K^{(i)}$ to energies induced by $S^{(i)}$) to show that each $\boldsymbol{x}^{(i)} \in [\Pi_h^{(i)}]$ has the representation $\boldsymbol{x}^{(i)} = \sum_E \boldsymbol{x}_E^{(i)} + \boldsymbol{x}_H^{(i)}$, where $\boldsymbol{x}_E^{(i)} \in \mathcal{V}_E^{(i)}$, $\boldsymbol{x}_H^{(i)} \in \mathcal{V}_H^{(i)}$ and where

$$\sum_E \boldsymbol{x}_E^{(i)T} S^{(i)} \boldsymbol{x}_E^{(i)} + \boldsymbol{x}_H^{(i)T} S^{(i)} \boldsymbol{x}_H^{(i)} \leq C^{(i)} (1 + \log(H/h))^2 \boldsymbol{x}^{(i)T} S^{(i)} \boldsymbol{x}^{(i)},$$

where $C^{(i)}$ depends on the variation of the coefficients a in $\overline{\Omega^{(i)}}$ and not on H or h. Then a simple application of Lemma 6.6.4 shows that the smallest eigenvalue of $\hat{S}^{(i)^{-1}} S^{(i)}$ is bounded below by $C^{(i)^{-1}} (1 + \log(H/h))^{-2}$. Expressing this using the Rayleigh quotient induced by $\hat{S}^{(i)}$ yields (6.6.55).

Case II: If $\partial\Omega^{(i)} \cup \partial\Omega_D = \emptyset$, then $K^{(i)}$ is the stiffness matrix corresponding to a pure Neumann problem for a PDE with only order second terms. Letting $\mathbf{1}_H^{(i)}$, $\mathbf{1}_h^{(i)}$ denote the unit vectors in $[\Pi_H^{(i)}]$, $[\Pi_h^{(i)}]$ respectively, it is easily seen that both $S^{(i)}$ and $\hat{S}^{(i)}$ are SPD on the restricted space $\mathrm{Im}(S^{(i)}) = \mathrm{Im}(\hat{S}^{(i)}) = \{\mathrm{span}(\mathbf{1}_h^{(i)})\}^{\perp}$. Similarly $\hat{S}_{HH}^{(i)}$ is SPD on $\{\mathrm{span}(\mathbf{1}_H^{(i)})\}^{\perp}$. The matrix on the right-hand side of (6.6.56) is well defined on $\mathrm{Im}(S^{(i)}) = \mathrm{Im}(\hat{S}^{(i)})$, and is a left-inverse for $\hat{S}^{(i)}$. The arguments of Case I can then be repeated for this case with $[\Pi_h^{(i)}]$, $\mathcal{V}_E^{(i)}$ and $\mathcal{V}_H^{(i)}$ being replaced by their orthogonal projections onto $\mathrm{Im}(S^{(i)})$. This yields inequalities (6.6.55) for all $\boldsymbol{x}^{(i)} \in \mathrm{Im}(S^{(i)})$. But since $\mathrm{Ker}(S^{(i)}) = \mathrm{Ker}(\hat{S}^{(i)}) = \{\mathrm{Im}(S^{(i)})\}^{\perp}$, (6.6.55) is also true in this case for all $\boldsymbol{x}^{(i)} \in [\Pi_h^{(i)}]$. ■

Now we can sum (6.6.55) over all substructures and recall (6.6.46), (6.6.47) to obtain the following corollary.

COROLLARY 6.6.9 *With $\hat{S}$ given by (6.5.45) we have*

$$\kappa(\hat{S}^{-1}S) \leq 5 \max_i(C^{(i)})(1+\log(H/h))^2.$$

Remark. The two-dimensional analogue of the results in [65] shows that $\kappa(\hat{S}^{-1}S)$ is independent of the coefficient jumps and also that it grows logarithmically with H/h. Corollary 6.6.9 provides a more precise statement of these facts, with both results provided by a single statement.

Chapter 7

The MasPar MP-1

7.1 Introduction

In the last decade, all branches of numerical analysis have been affected by the ideas and the realisation of vector and parallel computing. Early machines by Cray and IBM allowed vector operations to be carried out. Recently the emphasis has been mainly on parallel processing, with a large number of manufacturers and architectures to choose from.

The main aims of parallel processing lie in the following three areas :-

- **Speed.** There is a continuing desire to solve existing problems in ever decreasing times. The optimum speed–up on a machine with n processors is achieved when the code executes n times faster than on a single processor. Many of the algorithms in numerical linear algebra are currently being implemented in parallel in order to achieve, hopefully, near optimal speed–up.

- **Size.** Many of the applications in the worlds of computational fluid dynamics and semiconductor device modelling lead to problems with very large numbers of unknowns (typically $\mathcal{O}(10^6)$). In many of these examples this is

simply too many for one machine and hence a parallel architecture becomes necessary.

- **Complexity.** Although the advent of parallel computers is a new phenomenon, potentially parallel algorithms most certainly are not. The architectures now available make the implementation of these algorithms a viable possibility. This is most notable in the area of domain decomposition.

This intensive period of interest in parallel processing has seen many different parallel architectures offered to the scientific community. At present, we are still confronted with two, basically different, types of parallel architecture.

There are MIMD (multiple instruction multiple data) machines which, for simplicity, can be viewed as an interconnected cluster of workstations. Each is free to execute its own code on its own data and then must send or receive information when it requires it from another processor. Clearly such machines can perform several independent tasks at once. However, care is required to ensure that their asynchronous behaviour does not allow bottle–necks in any parallel implementations.

Alternatively there are SIMD (single instruction multiple data) machines. These typically have a large number of relatively small processors which work in lockstep, executing the same instructions on their individual data. Clearly for these types of machines to work effectively the task they are required to do must be inherently parallel. The MasPar MP-1 is an example of a SIMD architecture.

7.2 The MasPar system

7.2.1 Machine architecture

The MasPar MP-1 is a fine grained, massively data–parallel processing system. Each model has at least 1024 simple parallel data processor elements (PE's). The

machine comprises two major pieces: a front end and a data–parallel unit (DPU). Figure 7.1 below shows a schematic representation of the major components in the MasPar system.

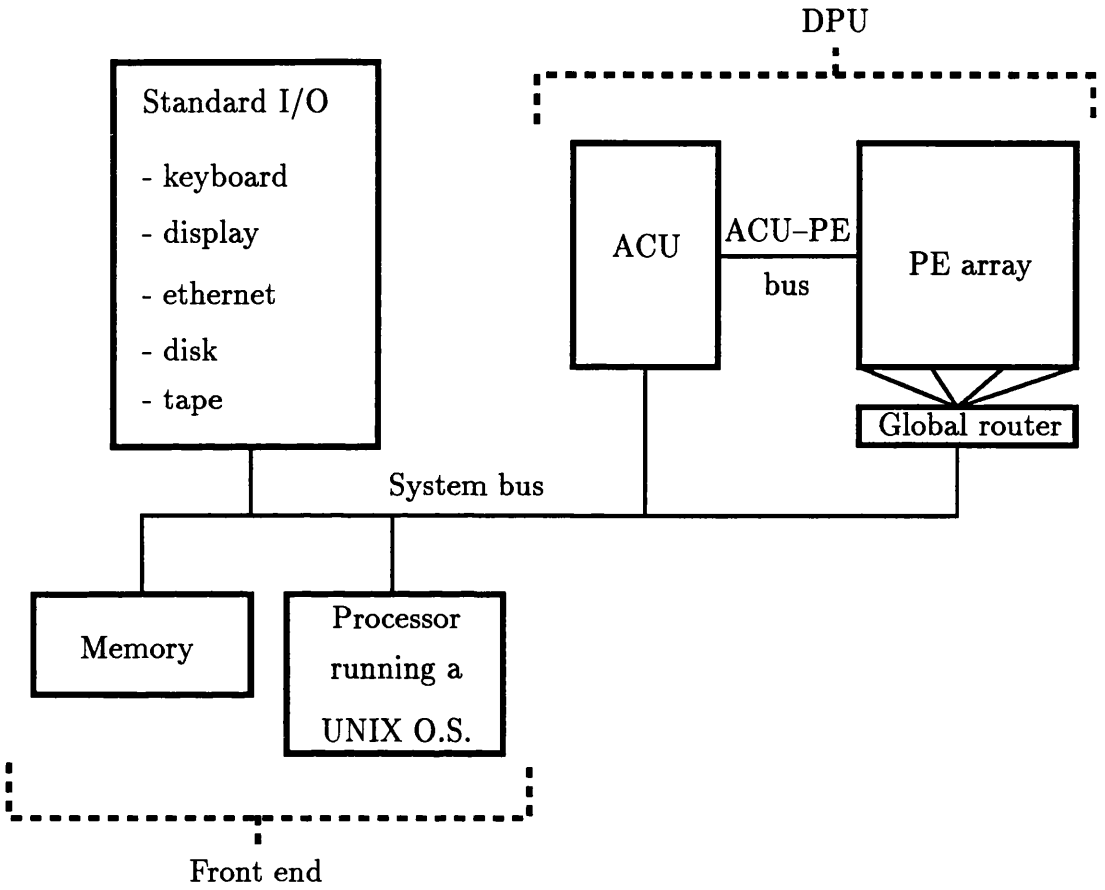


Figure 7.1: MasPar MP–1.

The Front End. This is a processor that runs an implementation of a UNIX operating system that provides services to the data-parallel system. The front end performs all operations that cannot be implemented in parallel on the array of PE's.

The Data Parallel Unit (DPU). This is the part of the system that performs all the parallel processing. The DPU consists of an array of PE's, an array control unit (ACU), and PE communications mechanisms.

The ACU. This is a load/store processor with its own registers, data and in-

struction memory. It controls the PE array, sending data and instructions to each PE simultaneously. It also has the capability of performing serial operations. This means that, using MasPar Parallel Application Language (MPL), it is possible to write a program that executes entirely on the DPU. However this is not practical in real applications as the ACU is not as powerful as the front end machine. In MasPar Fortran the compiler decides whether to use the DPU or the front end in a manner that is transparent to the user.

The PE array. Each PE is a load/store arithmetic processor with dedicated register space and RAM. Each has a 1.8 MIPS control processor, forty 32–bit registers and 16 KBytes of RAM. They are arranged in a 32×32 matrix. Each non-overlapping 4×4 group of PE's is called a cluster. The 1K PE array has 64 such clusters which are important in Global Router communications. The PE array is where all the parallel data is processed. Variables that are declared to be parallel (either by default or by the user) are located on the PE array. The PE's work in lockstep, all receive the same instruction simultaneously from the ACU. The user has the ability to mask sections of the array so that certain PE's are disabled. However for optimal strategies all PE's should be employed as often as possible.

PE Communications. There are three communication paths within the MasPar system. The use of these different types of communication has a marked effect on the performance of any parallel code.

X–Net Communication. Each PE in the array has the ability to communicate with one of its 8 nearest neighbours. These are called X–Net communications and the array is joined in a toroidal wrap so that every processor has 8 neighbours. X-Net communications can be incremented so that a processor can communicate with processors in a straight line along the directions of its nearest neighbours. X–Net communications are the quickest communication path in the MasPar system.

Global Router. The global router network allows random communication be-

tween any two processors on the PE array. PE clusters are involved with global router communications. The system can communicate with all PE clusters simultaneously but only one PE per cluster at any one time. A PE communicating with itself or any other PE in the same cluster means that cluster cannot communicate with any other at that time. Hence, in general, global router communication is slightly slower than X–Net communication but in many applications may be more general purpose. When programming in MPL the user can explicitly select between X-Net or Global Router communications. This control is lost when using MasPar Fortran but the mechanism chosen by the compiler is, in general, transparent in the source code.

ACU–PE Bus. This communication is most often used from the ACU to the PE's. In this case the ACU broadcasts instructions and data to the PE's. It is also required in the transition from parallel to serial execution and back again. The nature of the front end machine means that serial execution is extremely slow and hence should be avoided to obtain efficient use of the machine.

When programming in MPL the user may make explicit statements to communicate between the ACU and the PE array. MPL allows low level access to the DPU. When using MasPar Fortran (an adaptation of FORTRAN90) the user requires less knowledge of the DPU, however this is at the expense of some control over the communication mechanisms employed on execution.

The MasPar has a sophisticated debugging facility called the MSD (MasPar Symbolic Debugger). This allows the user to control and observe a program that is executing on both the front end and the DPU. The debugger shows the user a listing of the source code, allows the insertion of user specified breakpoints and shows the values of any desired variables. It can also show which of the PE's are enabled at any one time.

For a more detailed discussion of the MasPar machine architecture we refer the reader to [52].

7.3 Programming languages on the MP-1

7.3.1 MasPar parallel applications language (MPL)

MPL is the lowest level programming language that the MasPar supports. It is based on Kernighan and Ritchie C and can be used for DPU system programming similar to the way that C can be used on traditional machines. It gives the programmer the ability to explicitly assign variables to the DPU and to choose between communication mechanisms. This is in addition to the advantages of data structures already inherent in C. There are also library routines supplied to support MPL.

7.3.2 MasPar Fortran

MasPar Fortran is based on the widely used FORTRAN77 programming language with array processing features added from the FORTRAN90 ANSI standard and other enhancements from DEC's VAX Fortran. These extensions allow the programmer to effectively use the data-parallel processing capabilities of the MasPar system. FORTRAN77 is a subset of MasPar Fortran and as such FORTRAN77 code (under certain restrictions) can be run on the MasPar system. However such code does not make use of the parallel processing capabilities and will essentially run on the front end with an execution time reflecting this. To use the DPU effectively the user must modify his code with FORTRAN90 array statements to take advantage of the data-parallel hardware.

7.3.3 Programming in MasPar Fortran

Typically, FORTRAN77 operations on arrays are executed element by element using iterative DO loops. FORTRAN90 allows the user to write simple statements to perform array calculations and matrix operations. MasPar Fortran includes additional array assignments using the WHERE and FORALL statements, both

of which have limited parallel capability, and which are not in FORTRAN90. In MasPar Fortran the user does not need to explicitly state which variables are stored on the front end and which are stored on the DPU. The compiler will determine which parts of the code can be executed in parallel and which in serial by the way they are used in the program and allocate storage accordingly. Recall that it is undesirable to execute on the front end as this is very slow compared with the DPU and involves PE-ACU communication. The compiler will issue a warning when it comes across a section of code that cannot be executed in parallel and then, hopefully, the user will be able to make alterations to the code to make it run in parallel. This, of course, requires knowledge of the architecture of the machine and how MasPar Fortran works. The following is an example of how MasPar Fortran executes array calculations in parallel.

Example 7.3.1. Recall that, in FORTRAN77, to add two matrices we would need the following coded loop

```
REAL A(10,10), B(10,10), C(10,10)
....
DO 30 I = 1, 10
   DO 30 J = 1, 10
      A(I,J) = B(I,J) + C(I,J)
30 CONTINUE
```

Hence this iteration involves 100 serial steps before we have completed the matrix addition. In FORTRAN90 this matrix addition is coded in the following way

```
REAL A(10,10), B(10,10), C(10,10)
....
A = B + C
```

On a serial machine with a FORTRAN90 compiler this would still be executed

in 100 steps. However, on the MasPar system, compiled with the MasPar Fortran compiler this operation would be carried out on the DPU in one parallel step. In order to understand which MasPar Fortran commands will be executed in parallel the user must first know a little about how arrays are mapped to the DPU.

Default MasPar Fortran Array Mapping. Any array used in FORTRAN90 syntax is mapped to the DPU using *canonical* array allocation. This proves to be efficient for many array operations. On the MasPar machine arrays are mapped onto the PE grid by columns and rows. If the array is one–dimensional then it is mapped in a serpentine fashion onto the PE grid (starting at the top left processor). If an array is two–dimensional then the first dimension is mapped onto columns of the PE grid and the second dimension is mapped onto the rows. Hence the array A(3,2) is mapped as follows

A(1,1)	A(2,1)	A(3,1)
A(1,2)	A(2,2)	A(3,2)

If a two–dimensional array exceeds 32 by 32 then the overspill is placed into layers of the PE memory. Similarly if the array has more than two dimensions then the first two dimensions will be mapped across the PE grid and the remaining dimensions will be allocated to PE memory. The user can override this default mapping by using mapping directives but for our application the default mapping was found to be adequate. Hence returning to *Example 7.3.1*, we see that the matrices A, B and C would be mapped to the top left 10×10 square of processors and then each processor would add its element value of B and C and store the results in A.

Masked and Element Array Assignment. In addition to performing operations in parallel on entire arrays, as in *Example 7.3.1*, MasPar Fortran includes two extensions which allow the programmer to perform operations on array sections or elements. These are *not* part of the FORTRAN90 ANSI standard.

Firstly, in array assignment statements, the assignment of values can be masked according to the value of a logical expression appearing in a WHERE statement. The logical expression is evaluated first and the assignment statement is executed at elements that have the value TRUE. If an ELSEWHERE statement is also included then this is then executed at elements having the value FALSE. For example we could evaluate sign(A) as follows

```
WHERE (A .GE. 0.0)
   A = 1.0
ELSEWHERE
   A = -1.0
END WHERE
```

Secondly, a parallel array assignment can be specified in terms of array elements or array sections using a FORALL statement. The statement can take up to 3 integer subscripts, each of which can be incremented in a desired step size. The default step size is 1. The version of the compiler available to us would only generate parallel code provided the assignment statement following the FORALL command was sufficiently simple and involved each of the subscripts only once. For instance, the following code will be executed in parallel

```
FORALL(I=1:N,J=1:N) H(I,J) = 1.0 / REAL(I+J)
```

If the FORALL statement cannot be executed in parallel then a warning will be issued at compile time. The FORALL assignment will not execute in parallel for intrinsic functions, user written functions, triple FORALL indices, functions of FORALL indices, non-scalar array references or transformational intrinsics.

We found that if a FORALL statement had to be executed in serial because the compiler could not make it parallel, then its execution time increased markedly. The following example illustrates this.

Example 7.3.2. First consider storing two N by M matrices in the memory

of each PE and then trying to add the two matrices. The matrices would be declared as follows

```
REAL A(IX,IY,N,M), B(IX,IY,N,M), C(IX,IY,N,M)
```

The FORALL statement gives a very quick and simple way of doing this.

```
FORALL(I=1:IX,J=1:IY) A(I,J,:,:) = B(I,J,:,:) + C(I,J,:,:)
```

Alternatively we could execute this operation in a DO loop stepping through the elements of the matrices. This is written as

```
DO I = 1, N
   DO J = 1, M
      A(:,:,I,J) = B(:,:,I,J) + C(:,:,I,J)
   ENDDO
ENDDO
```

Experiments were carried out with the above code and execution times are as follows

IX	IY	N	M	FORALL	DO loop
32	32	50	10	15 mS	40 mS

As the table above shows, since the FORALL statement could be implemented fully in parallel by the compiler, it is a quicker method than the DO loop. Notice also that the DO loop has a certain degree of parallelism as all the processors are doing a single addition at any one time. This is an example of a successful use of the FORALL statement. However we now examine an example where FORALL is not suitable.

Example 7.3.3. Consider storing an N by N matrix and a N by 1 vector in the memory of each PE. Hence we have two arrays assigned by

```
REAL A(IX,IY,N,N), X(IX,IY,N)
```

Now suppose we wish to form the product of the matrix A(I,J,:,:) with the vector X(I,J,:) on each of the processors. The FORALL statement would appear to be the correct command to use with the intrinsic function MATMUL which multiplies any two conforming matrices together. Hence we would write

```
FORALL(I=1:IX,J=1:IY) Y(I,J,:) = MATMUL(A(I,J,:,:),X(I,J,:))
```

Alternatively we could implement a DO loop again

```
DO I = 1, N
   DO J = 1, N
      Y(:,:,I) = Y(:,:,I) + (A(:,:,I,J)*X(:,:,J))
   ENDDO
ENDDO
```

Various experiments were run with different size matrices and different amounts of the PE grid used. Timings were taken for the two different methods of performing the matrix-vector multiplications. In the following table S.O indicates that the machine crashed due to PE stack overflow. This means that in the course of execution a PE has exceeded its memory limit.

		n=10		**n=20**	
IX	**IY**	**DO loop**	**FORALL**	**DO loop**	**FORALL**
5	2	10 mS	210 mS	40 mS	634 mS
10	10	10 mS	1740 mS	40 mS	6130 mS
25	20	10 mS	8530 mS	40 mS	S.O
25	30	10 mS	S.O	40 mS	S.O
32	32	10 mS	S.O	41 mS	S.O

In all of these cases the FORALL statement is executed in serial because it is being used with an intrinsic function. A warning to this effect is given at compile

time. Hence all calculations are done on the (slow) front end. Now compare this with the DO loop version which is executed on the DPU. We see that as long as we do not exceed 1024 matrices this calculation will be done in N^2 steps. Hence the execution times are only increasing when N increases and not as we use more of the PE array. The speed–up is quite remarkable and we see that for problems covering a large proportion of the PE grid the FORALL statement is unable to cope with the instruction and causes a stack overflow.

These observations have proved extremely useful in the implementation that we now have running on our machine. By understanding how arrays are allocated and the limitations of some of the MasPar Fortran commands we have been able to reduce the amount of serial execution to a bare minimum.

For a more detailed discussion of MasPar FORTRAN programming features we refer the reader to [51] and [54].

7.4 Implementation of domain decomposition algorithms

Our implementations have all been in MasPar Fortran. On a machine such as the MasPar it is natural to assign one or more processors to each of the substructures in our problem. For ease of exposition we will assume that each substructure is associated with just one processor. Each processor assembles its own substructure stiffness matrix as a preprocessing step. The solution of the local Dirichlet problems to obtain the local Schur complement matrices, $S^{(i)}$, is at present done by the conjugate gradient method. This can be done in a highly parallel fashion, with the matrix–vector multiplies being performed by the DO loop outlined in *Example 7.3.3* above. As the philosophy of these domain decomposition algorithms is to keep the sub–problems as small as possible, this conjugate gradient loop will converge in a handful of steps. The construction of the local modified

right–hand side vectors, $\boldsymbol{c}^{(i)}$, can be achieved at the same time by appending the vectors $\boldsymbol{b}_I^{(i)}$ to the matrices $K_{IB}^{(i)}$ in the conjugate gradient solver.

In order that the implementation is efficient we must ensure that all the local Schur complement matrices are approximately the same size. A small number being much larger than the rest would cause a considerable bottle–neck in the code as the DO loops on the associated processors would be significantly longer. Ideally all the $S^{(i)}$ should be the same size. In practice this may mean differing physical sizes of substructure if we intend to grade the mesh.

Now, given any $\boldsymbol{x} \in [\Pi_h]$, we can store the local vectors $\boldsymbol{x}^{(i)}$ (containing the nodal values of $\Gamma^{(i)}$) on each processor. $S^{(i)}\boldsymbol{x}^{(i)}$ can be formed in parallel as described in *Example 7.3.3*, then for each node on Γ, add up the results from each substructure containing that node. Clearly, if the substructures are mapped to the PE grid in a sensible manner, this addition involves only X–Net communication and is therefore inexpensive. In practice a global vector is never stored. The added values are returned to the associated locally stored vectors. Inner products can also be evaluated in a similarly efficient way. There is just one addition across the whole PE array and this is done using the built–in function SUM. The only trick required is in taking account of the fact that all vertex values of a vector are stored up to 4 times and all edge values up to twice on the PE array.

The remaining implementation issue is the efficient solution of the preconditioning problems defined in Section 6.4. To find and explicitly invert the local edge preconditioners (as suggested in [66]) would be a very significant preprocessing step and would require far too much of the PE memory. Likewise the coarse grid matrix is large (in our implementation as large as 961×961) and hence would be far too expensive to construct and then explicitly invert. Instead our implementation involves iterative solution of the preconditioning problem. That is, each preconditioning step is done by a sequence of "inner iterations".

Vertex space preconditioner.

By our definition of the discretisation in Chapter 2, each substructure vertex is surrounded by, at most, 4 substructures. Here we shall consider the problem with entirely Dirichlet boundary, in which case every vertex is surrounded by exactly 4 substructures. Problems with a partly Neumann boundary can be handled in an analogous manner with some small amount of additional work to handle the vertices on the Neumann boundary. A typical vertex space is shown in Figure 7.2.

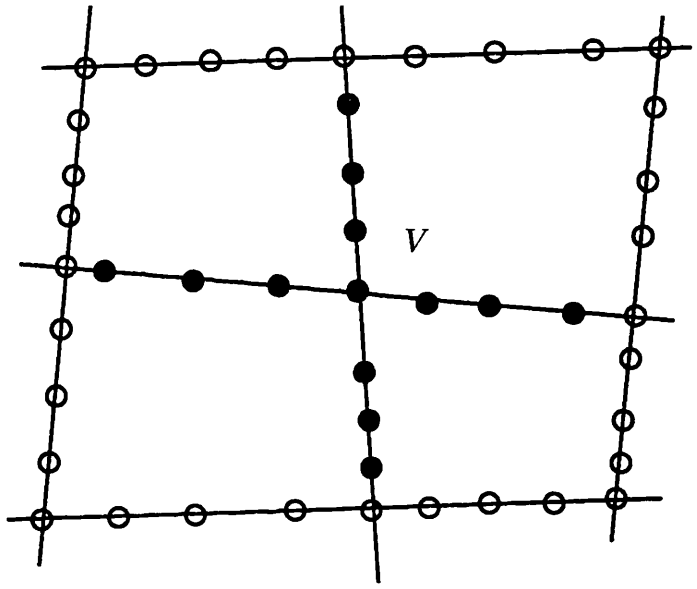


Figure 7.2: A typical vertex space.

Following the notation in Chapter 6, for a vertex $V \in \Pi_H$, the associated vertex space $[\Pi_H]_V$ consists of vectors in $[\Pi_h]$ which are zero except at the black nodes depicted in Figure 7.2.

Then on each vertex space, we can consider the solution of the local problem,

$$S_V \boldsymbol{z}_V = R_V \boldsymbol{r}$$

for $\boldsymbol{z}_V \in [\Pi_h]_V$, where $R_V \boldsymbol{r}$ is the restriction of $\boldsymbol{r} \in [\Pi_h]$ to vertex space $[\Pi_H]_V$, and S_V is the minor of S containing rows and columns corresponding to the

nonzero nodes lying on $[\Pi_H]_V$. If we denote the minor of $S^{(i)}$ containing rows and columns corresponding to nonzero nodes of $[\Pi_H]_V$ by $S_V^{(i)}$ then it is clear that

$$S_V = \sum_i S_V^{(i)},$$

where the sum is just over the four substructures surrounding V. Hence the multiplication of any vector $\boldsymbol{x}_V \in [\Pi_H]_V$ by S_V is equivalent to $\sum_i S_V^{(i)} \boldsymbol{x}_V^{(i)}$ where $\boldsymbol{x}_V^{(i)}$ is just those elements of $\boldsymbol{x}_V$ which also lie on $\Gamma^{(i)}$. Also the product $S_V^{(i)} \boldsymbol{x}_V^{(i)}$ is equivalent to $S^{(i)} \tilde{\boldsymbol{x}}_V^{(i)}$ at nodes of $[\Pi_H]_V$ where $\tilde{\boldsymbol{x}}_V^{(i)}$ is the extension by zero of $\boldsymbol{x}_V^{(i)}$ to a vector of nodal values on $\Gamma^{(i)}$. As we already have the matrices $S^{(i)}$ available to us, then the product $S_V \boldsymbol{x}_V$ is easily formed. Furthermore, instead of performing all four matrix–vector multiplications in parallel, if we step around the vertex V performing the multiplications individually, all the vertex spaces can be dealt with at once. Hence all the products $S_V \boldsymbol{x}_V$ are evaluated in highly parallel steps. Therefore on an $m \times m$ array of processors we can solve up to $(m-1)^2$ vertex space problems at once. We stop the inner CG loops when the largest residual from the individual problems drops below some user specified tolerance.

Edge space preconditioner.

Recall, by the definition of the discretisation in Chapter 2, each substructure has exactly four edges and four vertices. On each edge, E, containing unknown nodal values we require the solution to the local problem

$$\hat{S}_{EE} \boldsymbol{z}_E = R_E \boldsymbol{r}.$$

where $R_E \boldsymbol{r}$ is the restriction of $\boldsymbol{r} \in [\Pi_h]$ to edge E and $\hat{S}_{EE}$ is the sum of the local matrices $\hat{S}_{EE}^{(i)}$ from the substructures containing edge E.

It is clear from Section 6.5 that $\hat{S}_{EE}^{(i)}$ is the minor of $S^{(i)}$ containing rows and columns corresponding to the nodes lying on edge E. Hence, although it would be expensive in terms of memory to actually form and factor $\hat{S}_{EE}$, it is extremely

cheap to multiply by $\hat{S}_{EE}$ given that we already have the local Schur complement matrices $S^{(i)}$.

Given $\boldsymbol{x}_E$, a vector of values corresponding to nodes on edge E, then by our discretisation E lies in at most two substructures, $\Omega^{(j)}$ and $\Omega^{(k)}$ say. We first extend $\boldsymbol{x}_E$ with zeros to a vector of nodal values on $\Gamma^{(j)}$ and multiply this by $S^{(j)}$. The values of the result at nodes of E are then just $\hat{S}^{(j)}_{EE}\boldsymbol{x}_E$. Repeat this procedure on $\Omega^{(k)}$, and then add the two vectors $\hat{S}^{(j)}_{EE}\boldsymbol{x}_E$ and $\hat{S}^{(k)}_{EE}\boldsymbol{x}_E$ to obtain the required product. This addition requires only X–Net communication. Furthermore, all processors can be performing the local multiplications at once and hence the whole operation only requires two steps. In the case of an entirely Dirichlet boundary on a quadrilateral domain, by storing information about the southern and eastern edges in each associated processor, we can complete all the local multiplications in two sweeps. Each sweep containing, at most, two parallel steps. When we introduce some Neumann boundaries we may need one extra sweep to cope with the multiplications in the boundary substructures.

With the ability to perform these multiplications cheaply and in parallel it is easy to solve the edge space problems by an inner iterative method such as the conjugate gradient method.

Coarse Grid Preconditioner.

As mentioned earlier, the coarse grid matrix $\hat{S}_{HH}$ is potentially large and is certainly not suited to direct inversion on a massively parallel SIMD architecture. However, once again we find that multiplication by $\hat{S}_{HH}$ can be achieved relatively cheaply and with a high degree of parallelism.

In the preprocessing stage that generates the local Schur complement matrices $S^{(i)}$ on each processor we also derive the local coarse grid matrices $\hat{S}^{(i)}_{HH}$. On each processor we generate the relevant matrix $\mathcal{R}^{(i)T}_H$ which linearly interpolates from nodes of the coarse grid to nodes in the edge spaces on $\Gamma^{(i)}$. The $\hat{S}^{(i)}_{HH}$ are then formed by premultiplying $S^{(i)}$ by $\mathcal{R}^{(i)}_H$ and postmultiplying by $\mathcal{R}^{(i)T}_H$. Each of these

multiplications can be done on all the processors simultaneously. The stored $\hat{S}_{HH}^{(i)}$ are just 4×4 matrices and the $\mathcal{R}_H^{(i)T}$ can subsequently be overwritten when we are pushed for memory space.

With the local coarse grid matrices we can then locally and in parallel multiply by a local coarse grid vector and then add at the coarse grid vertices to obtain the global product. This again only requires X–Net communication and because of the fixed, small size of the $\hat{S}_{HH}^{(i)}$ this is guaranteed to be an extremely quick operation. Hence with the cheap multiplication, the coarse grid problem is also solved with an inner conjugate gradient iteration.

7.4.1 Stopping criterion.

Firstly we need a stopping criterion for the outer PCGM loop. To do this consider the error at the k^{th} step, defined by

$$\boldsymbol{e}^k = \boldsymbol{x} - \boldsymbol{x}^k$$

where $\boldsymbol{x}$ is the exact solution of $S\boldsymbol{x} = \boldsymbol{c}$. Then using the definition of $\boldsymbol{z}^k$ in the preconditioned conjugate gradient algorithm,

$$S\boldsymbol{e}^k = \boldsymbol{c} - S\boldsymbol{x}^k = \boldsymbol{r}^k = \hat{S}\boldsymbol{z}^k$$

where $\hat{S}$ is the preconditioner. Hence

$$\|\boldsymbol{e}^k\| \leq \|S^{-1}\hat{S}\| \, \|\boldsymbol{z}^k\|.$$

Therefore if $\|S^{-1}\hat{S}\|$ is bounded independently of the number of substructures and the mesh diameter of the triangulation then monitoring $\|\boldsymbol{z}^k\|_2$ will provide a robust estimate of the error. If no preconditioning is used then we monitor $\|\boldsymbol{r}^k\|_2$. This is not as good an estimate of the error at the k^{th} step, but is readily available to us. Hence, in order to achieve a reasonably accurate solution, the

stopping criterion for the unpreconditioned problem will have to be smaller than that for the preconditioned problem.

We also require a condition to halt the inner CG loops which are approximately solving the preconditioning problems. Obviously we need a fairly accurate solution but at the same time do not want to waste time solving to too high an accuracy in the initial stages of the outer PCGM loop. So far we have used the following criterion

$$\text{If} \quad \text{TOL} \times \|\boldsymbol{r}_{outer}\|_2 \geq 3 \times 10^{-4} \quad \text{then}$$

$$\|\boldsymbol{r}_{inner}\|_2 \leq 3 \times 10^{-4}$$

else

$$\|\boldsymbol{r}_{inner}\|_2 \leq \max\{10^{-7}, \text{TOL} \times \|\boldsymbol{r}_{outer}\|_2\}$$

end if

where $\boldsymbol{r}_{inner}$ is the residual from the inner iteration, $\boldsymbol{r}_{outer}$ is the residual from the outer PCGM iteration and TOL is some user specified tolerance.

7.5 Preliminary numerical results

So far, in our implementations of this algorithm, we have made some further simplifications mainly to reduce the storage overheads on our machine and also to ease the coding. Firstly we have divided the domain into $m \times m$ equal square substructures and assigned each substructure to a PE. Hence 32 is the maximum value of m we have run. The domain has then been triangulated with a uniform mesh consisting of $n \times n$ nodes internal to each substructure. Therefore the fine mesh diameter, h, is equal to $1/(m(n+1))$. This gives us an equal load balance on each of the PE's throughout the computation. Subsequently no processors are left idle, waiting for others to finish calculations.

In our first implementation of the algorithm we had the ability to run the code with either no preconditioning at all, vertex preconditioning only, vertex space plus coarse grid preconditioning or edge space plus coarse grid preconditioning. In all of the following preliminary examples the (outer) iteration was stopped when the two–norm of the monitored residual dropped below 3×10^{-4}.

Example 7.5.4 Poisson's equation

$$-\triangle u = 4 \quad \text{in} \quad \Omega = [0,1] \times [0,1],$$
$$u = 0 \quad \text{on} \quad \partial\Omega.$$

m	n	H	h	**No precon**	**Vertex Only**	**Vertex + Coarse**	**Edge + Coarse**
8	1	1/8	1/16	14	9	6	5
8	3	1/8	1/32	21	10	7	6
8	5	1/8	1/48	26	10	7	7
8	7	1/8	1/64	30	10	7	8
8	9	1/8	1/80	34	10	7	8
8	11	1/8	1/96	36	11	7	9
16	1	1/16	1/32	27	17	6	5
16	3	1/16	1/64	40	18	7	6
16	5	1/16	1/96	50	18	8	7
32	1	1/32	1/64	52	32	7	5
32	3	1/32	1/128	78	33	8	5
32	5	1/32	1/192	95	33	9	6
32	10	1/32	1/352	127	34	14	8

Table 7.1: Poisson's equation

In all cases the inner loop tolerance (TOL) was fixed at 10^{-3}. Here the condition number of S grows only with the number of unknowns in the problem.

Hence in Table 7.1 we see that, for a fixed number of substructures, the number of iterations to convergence without any preconditioning increases as h decreases. Notice also that if we fix the total unknowns and vary the number of substructures that we use, then convergence is quickest for the smallest number of substructures. This is because we have fewer substructure boundary unknowns in this problem. However we have larger problems to solve in order to find the Schur complement matrix in this case and hence the preprocessing step will take longer.

We see that if we use the vertex space preconditioner alone, convergence is independent of the number of mesh points in each substructure, but slows as the number of substructures increases. Also the number of iterations is still rather high. On the other hand, if we use vertex space plus coarse grid preconditioning then we see that we have convergence independent of both the number of substructures and the fine mesh diameter. The slight rise in the number of iterations required as we increase n is due to our inexact solution of the preconditioning problems. Indeed, if we use a TOL of 10^{-5} for the case $m = 32, n = 10$ then we find that the iteration converges after 7 steps. This is in full accordance with the theory of Chapter 6.

Finally we see that convergence using the edge space plus coarse grid preconditioner is weakly effected by the ratio H/h. We find that we cannot improve the convergence by solving the preconditioning problems more accurately. Having said this, convergence would still actually appear quicker than the vertex space equivalent in most of the above cases.

Example 7.5.5 Two–valued rough coefficient (see, e.g. [27])

$$-\nabla.(a\nabla u) = 4 \quad \text{in} \quad \Omega = [0,1] \times [0,1],$$
$$u = 0 \quad \text{on} \quad \partial\Omega,$$

where

$$a = \begin{cases} k_1, & y > 0.5, \\ k_2, & y \leq 0.5. \end{cases}$$

In this example, $\kappa(S)$ not only grows with the number of unknowns in the problem, it is also proportional to $\max\{k_1, k_2\}/\min\{k_1, k_2\}$. Table 7.2 gives results obtained with an inner loop tolerance of 10^{-3}.

m	n	H	h	k_1	k_2	**No precon**	**Vertex + Coarse**	**Edge + Coarse**
32	1	1/32	1/64	1	10^2	413	9	5
32	3	1/32	1/128	1	10^2	657	9	5
32	1	1/32	1/64	1	10^3	983	9	5
32	3	1/32	1/128	1	10^3	1767	17	5

Table 7.2: Two–valued rough coefficient

We now begin to see the effect of the discontinuous coefficient. Without preconditioning the CG method takes a prohibitive number of iterations to converge. It has now become essential to precondition the CG method. Both the preconditioning strategies result in a large reduction in the number of iterations required for convergence. We also see that the so–called "optimal" vertex space plus coarse grid preconditioning method is weakly effected by the jump in the coefficient. Alternatively, the edge space plus coarse grid preconditioner provides a method which is robust to the jumps in a. Again, by choosing a tighter inner loop tolerance, we may have improved upon the result for the vertex space plus coarse grid preconditioner in the last row of Table 7.2.

Obviously this inner loop tolerance is unsatisfactory and Table 7.3 shows even more clearly what a crucial role it can play in determining the number of iterations required for convergence.

We conclude from Table 7.3 that if we do not solve the preconditioning problems accurately then, at best we can expect to perform more outer iterations to

reach convergence, and at worst, we lose orthogonality of the search directions and fail to converge in a reasonable number of iterations at all. In the light of the extremely large jumps occurring in the coefficients in the semiconductor equations it was decided to concentrate on an implementation of the edge space plus coarse grid preconditioner. Although the convergence of this method is weakly effected by the ratio H/h, the results in examples 7.5.4 and 7.5.5 indicated that this effect would be less than that of the jump in the coefficient in the vertex space plus coarse grid method.

m	n	k_1	k_2	**Inner Tolerance**	**No Precon**	**Edge + Coarse**
32	3	10^{-3}	10^3	10^{-4}	6824	18
32	3	10^{-3}	10^3	10^{-5}	6824	12
32	3	10^{-3}	10^3	10^{-6}	6824	11
32	3	10^{-3}	10^3	10^{-8}	6824	10
32	3	10^{-3}	10^3	10^{-9}	6824	10

Table 7.3: Two–valued rough coefficient

Timings We have taken timings for the solution of $S\boldsymbol{x} = \boldsymbol{c}$ in the above examples. For the small or relatively well conditioned problems we find that the code is quickest without any preconditioning. This is mainly due to the overhead in solving coarse grid problems which are generally large. However when we progress to the very badly conditioned problems, such as the last example, then preconditioning is necessary to obtain adequate turn around times. In the last example the code was twice as quick when preconditioning as it was when no preconditioner was used. Having said this, the code was still considered too slow given that our goal was to recursively solve problems with much larger jumps in the coefficients such as those arising in the Gummel iteration (see Chapter 5). The bottle–neck in the implementation was the solution of the coarse grid

problems. Clearly these are crucial in order to obtain convergence in a small number of outer iterations and hence a more efficient method of inverting the coarse grid matrix was sought.

7.6 Further improvements

The first modification to the code was to eliminate the user specified inner tolerance. We first tried a fixed number of steps for the inner iterative solver. This proved unsatisfactory as the choice of this number was somewhat of a "black art" (as the choice of the inner tolerance had been), and the solution to the coarse grid problems would become inaccurate as the jumps in the coefficient became significant. Instead we chose to solve the preconditioning problems to a fixed accuracy. *Example 7.5.5* above had shown us that the stopping tolerance needed to be very small if the convergence properties outlined in Chapter 6 were to be observed when the jump became large. Because of this, we opted to iterate the inner loops until the two–norm of the residual fell below 10^{-10}.

The one remaining task was to speed up the coarse grid solve. Several improvements in the coding of the coarse grid loop had meant that each step of the iteration was executed extremely quickly. It seemed unlikely that we could extract the desired speed–up just by a more efficient syntax. The real problem lay in the fact that, although the coarse grid problem is smaller than the original problem, it is still relatively large and potentially ill–conditioned when the jumps in the coefficients are large. Hence the CG method applied to this problem was taking a prohibitively large number of iterations to converge. Coupled with this, we were estimating the error at each step of this inner iteration by the two–norm of the coarse grid residual. This is not a very reliable error estimator and hence we would sometimes exit the coarse grid solver with a poor solution to the coarse grid problem.

The answer was to precondition the coarse grid CG method. However, since

we would now be "preconditioning the preconditioner" it was imperative that the preconditioner be as cheap and as easy to implement as possible. Our experiences with the one–dimensional semiconductor problems had already shown that diagonal scaling was very successful when solving the linearised electron and hole continuity equations. Also using the diagonal of your matrix as a preconditioner in the CG method is a well–known cheap and often effective strategy. Now recall that we have already formed and stored the local coarse grid matrices $S_H^{(i)}$ in the existing version of the code. Hence, for the cost of an extra 4×1 vector on each processor, we could easily evaluate and store the diagonal of S_H. This merely involves adding 4 values from substructures (and hence processors) surrounding each vertex. This operation already existed as a subroutine for use in calculating the coarse grid residual.

Hence we have updated our version of the algorithm, solving the preconditioning problems to within a fixed tolerance and using a CG loop with diagonal preconditioner for the solution of the coarse grid problem. The results given in the following section will hopefully expose the advantages of this improved implementation.

7.7 More model problems

Example 7.7.6 Semiconductor geometry.

Both the old and the new version of the code were run on the model problem

$$-\nabla.(a\nabla u) = 0$$

over the domain shown in Figure 7.3 with Dirichlet conditions on the parts of the boundary indicated with arrows and homogeneous Neumann boundary conditions elsewhere. We take u to be 0 on the upper Dirichlet boundary and 0.2 on the lower Dirichlet boundary. The coefficient function a is taken to be piecewise constant with value c_1 in region **A** and c_2 in region **B**.

In the tables that follow, "time" is the wall clock time in seconds that the solver took, "E_{tot}" is the total number of inner edge space iterations and "C_{tot}" is the total number of inner coarse grid iterations for that run. Both cases 1 and 2 refer to preconditioning with edge space plus coarse grid preconditioner. However case 1 is the original version of the code which does not employ any preconditioning of the inner coarse grid CG iteration whereas case 2 is the improved version of the code with diagonal preconditioner for the inner coarse grid CG iteration. In Tables 7.4 and 7.5 all inner solves are iterated until the monitored residual becomes less than 10^{-10}.

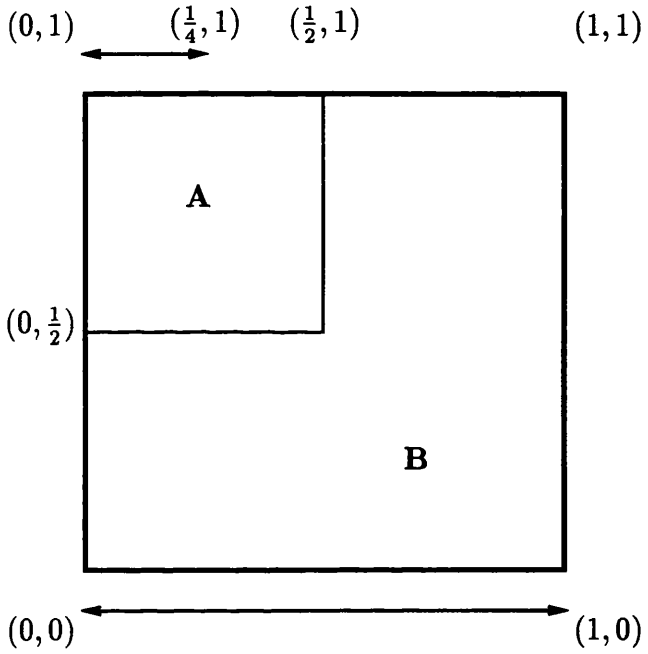


Figure 7.3: Domain of computation.

Table 7.4 again shows the effect of increasing the ratio H/h. This causes a slight increase in the number of preconditioned iterates required. The table also shows only a modest speed–up gained by preconditioning the coarse grid problem. At first this appears disappointing, but it should be realised that since there is only a relatively small jump in the coefficient in this example then the coarse grid problem will not be too badly conditioned. Hence diagonal preconditioning

of the coarse grid problem is not having a great effect.

				Case 1				Case 2			
m	n	c_1	c_2	its	time	E_{tot}	C_{tot}	its	time	E_{tot}	C_{tot}
8	3	10^{-1}	10^{+1}	7	44.9	24	575	7	36.7	24	222
16	3	10^{-1}	10^{+1}	7	78.6	24	1460	7	52.2	24	438
32	3	10^{-1}	10^{+1}	8	161.5	27	3615	8	96.0	27	958

Table 7.4: Outer loop stopping tolerance of 3×10^{-4}

				Case 1				Case 2			
m	n	c_1	c_2	its	time	E_{tot}	C_{tot}	its	time	E_{tot}	C_{tot}
32	1	10^{-3}	10^{+3}	12	397	13	14175	12	67	13	1288
32	1	10^{-4}	10^{+4}	13	435	14	15565	12	69	13	1316
32	1	10^{-5}	10^{+5}	12	404	13	14445	12	70	13	1317

Table 7.5: Outer loop stopping tolerance of 3×10^{-7}

Table 7.5 shows how the edge space plus coarse grid preconditioner is robust to the jump in the coefficient provided the jump occurs along substructure edges. In these examples the coarse grid problem is becoming badly conditioned and we see that by diagonal preconditioning the inner solve we can have a dramatic effect on the solution time. In fact, we are seeing a speed–up of nearly 6 times over our original implementation. This is due entirely to the reduction in the required number of coarse grid iterations. Note also that, although we reduce this number by a factor of about 11, we only obtain a speed–up of 6 since each inner coarse grid iterate in case 2 will be slower than that in case 1 due to the extra preconditioning operation.

In Table 7.6 we perform all inner solves to a tolerance of 10^{-12}. This example is reflective of the size of jump one would expect to see in the continuity equations for a semiconductor model. We were very encouraged to see that our

improved code again performed nearly 6 times faster than the original version. Furthermore, to solve this problem without any preconditioning at all took 1058 seconds. Obviously, in this case the outer stopping tolerance was considerably reduced as we were only monitoring the residual of the CG iteration. This is some 13 times slower than using our latest implementation of the preconditioner. With these results we could proceed to an implementation of Gummel's method for a two–dimensional problem with the code developed above performing the linear solves.

				Case 1				**Case 2**			
m	n	c_1	c_2	its	time	E_{tot}	C_{tot}	its	time	E_{tot}	C_{tot}
32	1	10^{-8}	10^{+8}	12	458	19	16446	12	83	19	1715

Table 7.6: Outer loop stopping tolerance of 3×10^{-7}

Table 7.7 gives numbers of unknowns for problems of the size that we have discussed above.

m	n	**Total unknowns**	**Substructure boundary unknowns**
8	1	225	161
8	3	961	385
8	5	2209	609
8	7	3969	833
8	9	6241	1057
8	11	9025	1281
16	1	961	705
16	3	3969	1665
16	5	9025	2625
32	1	3969	2945
32	3	16129	6913
32	5	36481	10881
32	10	123201	20800

Table 7.7: Problem sizes

Chapter 8

Massively parallel solution of a semiconductor problem

8.1 Introduction

In this chapter we discuss the solution of the model semiconductor problem

$$-\lambda^2\Delta\psi + \delta\{\exp(\psi - v) - \exp(w - \psi)\} - d = 0, \tag{8.1.1}$$

$$-\nabla.(\exp(\psi - v)\nabla v) - \sigma\rho_v r(\psi, v, w) = 0, \tag{8.1.2}$$

$$-\nabla.(\exp(w - \psi)\nabla w) + \sigma\rho_w r(\psi, v, w) = 0, \tag{8.1.3}$$

on the domain Ω shown in Figure 8.1 with boundary conditions and doping profile, d, as indicated. This represents a two–dimensional p–n diode with contacts indicated by the arrows at the boundary. The boundary conditions given are such that the diode is in reverse bias, with an applied voltage of αU_T. Once again we have used the statistics given in [62], which give the values $\lambda^2 = 1.6715 \times 10^{-7}$, $\delta = 1.22 \times 10^{-8}$, $\beta = 18.2218$ and $\sigma = 3.17 \times 10^{-15}$. We also use $\rho_v = \rho_w = 1/450$.

For simplicity, we divide Ω into $m \times m$ equal square substructures , where m is a multiple of 4. This ensures that the collision point between the Dirichlet

and Neumann boundary conditions on the top boundary is a node of the coarse grid and that the jump in d occurs entirely along substructure edges. We do this because, in many practical examples, we have seen a sharp jump in the potential at doping profile interfaces. Each substructure is then further divided with a uniform triangulation containing $n+2$ nodes along each substructure edge. We then discretise (8.1.1)–(8.1.3) by the finite element method with piecewise linear elements.

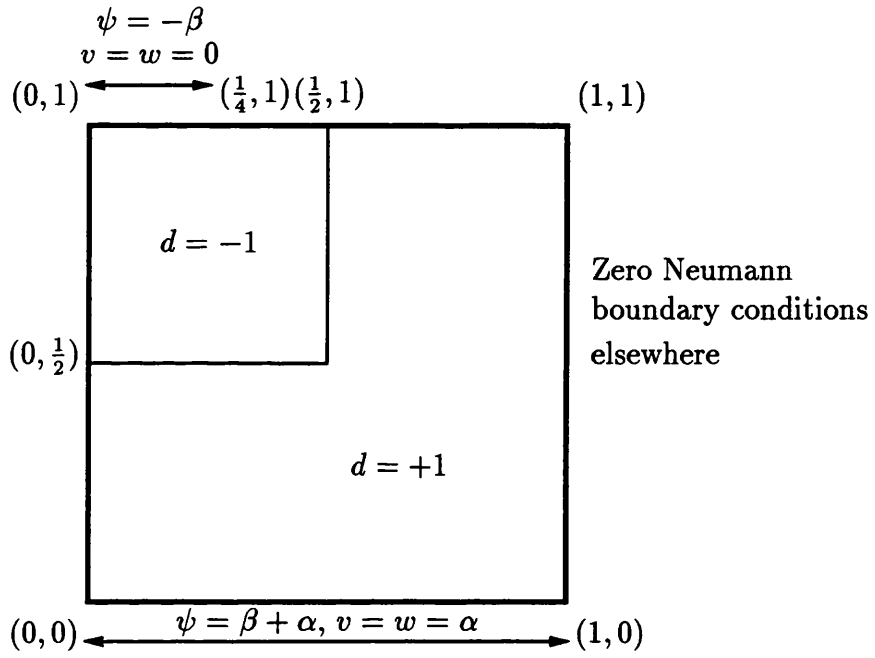


Figure 8.1: Model diode problem.

As discussed in Chapter 2, we use mass lumping for the zeroth order terms and take the harmonic average of the exponential coefficients in (8.1.2) and (8.1.3). A more detailed account of how this is achieved computationally is given in Section 8.3. The system (8.1.1)–(8.1.3) discretised in this way can be expressed as

$$\lambda^2(K(0)\boldsymbol{\Psi}+K_D(0)\boldsymbol{\Psi}_D)+\mathrm{diag}\{\boldsymbol{w}\}[\delta(\exp(\boldsymbol{\Psi}-\boldsymbol{V})-\exp(\boldsymbol{W}-\boldsymbol{\Psi}))-\boldsymbol{d}] = \boldsymbol{0}, \quad (8.1.4)$$

$$K(\Psi - V)\boldsymbol{V} + K_D(\Psi - V)\boldsymbol{V}_D - \sigma\rho_v\boldsymbol{r}(\boldsymbol{\Psi}, \boldsymbol{V}, \boldsymbol{W}) = \boldsymbol{0}, \quad (8.1.5)$$

$$K(W - \Psi)\boldsymbol{W} + K_D(W - \Psi)\boldsymbol{W}_D + \sigma\rho_w\boldsymbol{r}(\boldsymbol{\Psi}, \boldsymbol{V}, \boldsymbol{W}) = \boldsymbol{0}. \quad (8.1.6)$$

Here $\boldsymbol{w}$ is the vector of weights obtained from the quadrature rule (2.3.45) and $\boldsymbol{r}$ is the Shockley-Read-Hall recombination rate, (1.4.46) discretised by the Galerkin method with a nodal quadrature rule.

We then employ the Gummel iteration which was introduced in Chapter 2 and analysed (for $r = 0$) in Chapter 5. This implementation is executed on the MasPar MP–1 which was discussed in Chapter 7. Much of the coding of this algorithm in an efficient parallel manner is non–trivial. An overview of the Gummel algorithm is shown in Figure 8.2. In the following two sections we discuss how the potential and continuity equations in the Gummel loop are solved on a massively parallel machine. Finally we give numerical results obtained from extensive use of the parallel Gummel solver.

8.2 Solution of the potential equation

Recall from Chapter 3, given $(V, W) \in B(\Omega)$, the discretised potential equation may be written in the form

$$\boldsymbol{F}(\boldsymbol{\Psi}) = \lambda^2(K\boldsymbol{\Psi} + K_D\boldsymbol{\Psi}_D) + \boldsymbol{g}(\boldsymbol{\Psi}) = \boldsymbol{0}. \tag{8.2.7}$$

Then the Jacobian of (8.2.7) can be written

$$J(\boldsymbol{\Psi}) = \lambda^2 K + \boldsymbol{g}_{\boldsymbol{\Psi}}(\boldsymbol{\Psi}), \tag{8.2.8}$$

where $\boldsymbol{g}_{\boldsymbol{\Psi}}$ is the diagonal Jacobian matrix of $\boldsymbol{g}$. We solve (8.2.7) by the quasi–Newton method introduced in Chapter 3. That is, given lower and upper solutions, $\boldsymbol{x}^k, \boldsymbol{y}^k$ respectively, the updated solutions are defined by

$$\boldsymbol{x}^{k+1} = \boldsymbol{x}^k - (A(\boldsymbol{x}^k, \boldsymbol{y}^k))^{-1}\boldsymbol{F}(\boldsymbol{x}^k), \tag{8.2.9}$$

$$\boldsymbol{y}^{k+1} = \boldsymbol{y}^k - (A(\boldsymbol{x}^k, \boldsymbol{y}^k))^{-1}\boldsymbol{F}(\boldsymbol{y}^k), \tag{8.2.10}$$

where

$$A(\boldsymbol{x}^k, \boldsymbol{y}^k) = \max\left\{J(\boldsymbol{x}^k), J(\boldsymbol{y}^k)\right\} = \lambda^2 K + \max\left\{\boldsymbol{g}_{\boldsymbol{\Psi}}(\boldsymbol{x}^k), \boldsymbol{g}_{\boldsymbol{\Psi}}(\boldsymbol{y}^k)\right\}. \tag{8.2.11}$$

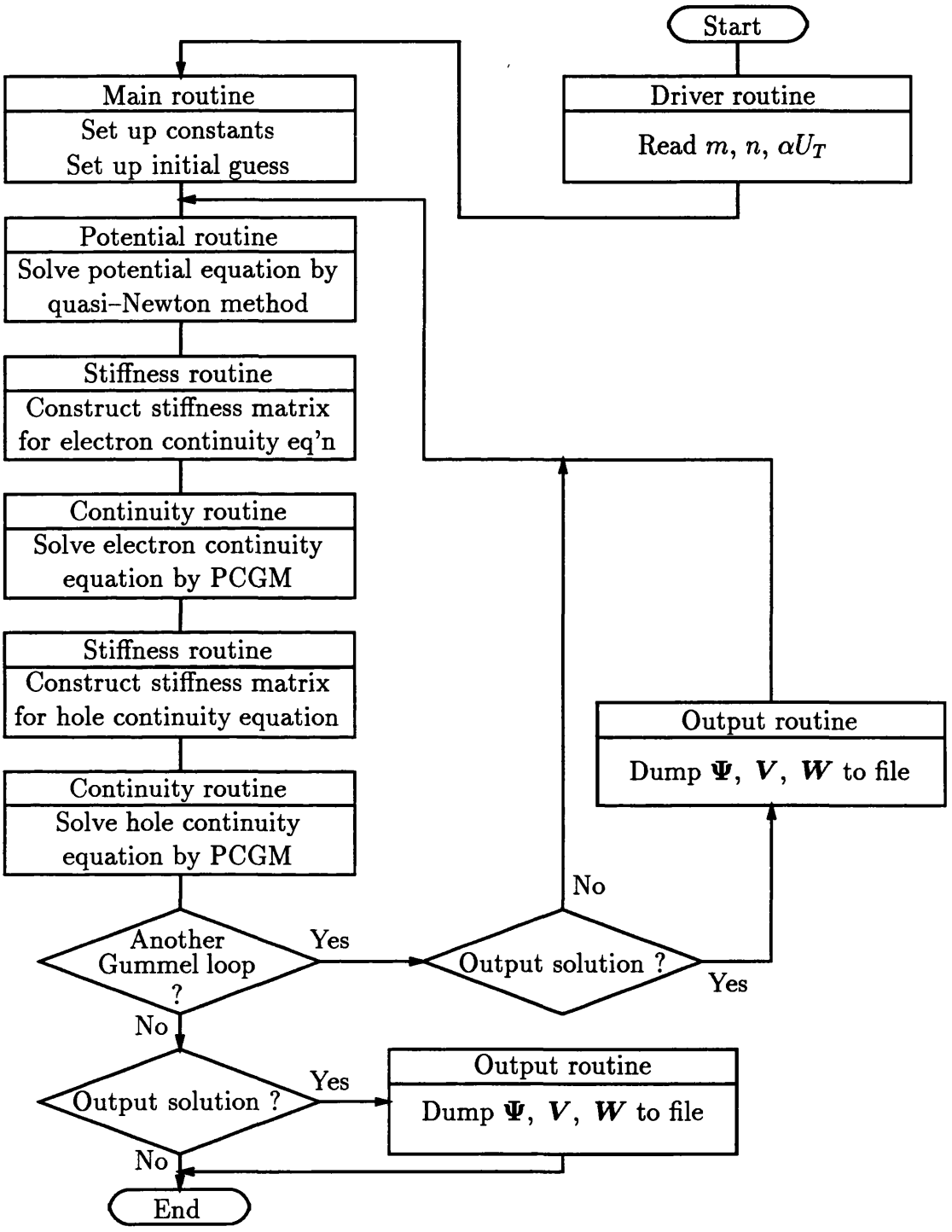


Figure 8.2: Overall flow of control.

Hence at each step of the iteration we have two problems of the form

$$A(\boldsymbol{x}^k, \boldsymbol{y}^k)\boldsymbol{z} = \boldsymbol{F}, \tag{8.2.12}$$

to solve. In order to achieve this with the domain decomposition solver described in Chapter 7 we must first reduce (8.2.12) to a Schur complement problem. To do this write

$$G^k := \max\left\{\boldsymbol{g}_\Psi(\boldsymbol{x}^k), \boldsymbol{g}_\Psi(\boldsymbol{y}^k)\right\},$$

and recall that G^k is a diagonal matrix. Then (8.2.12) can be expressed as

$$\left(\lambda^2 \begin{bmatrix} K_{II} & K_{IB} \\ K_{IB}^T & K_{BB} \end{bmatrix} + \begin{bmatrix} G_I^k & 0 \\ 0 & G_B^k \end{bmatrix}\right) \begin{pmatrix} \boldsymbol{z}_I \\ \boldsymbol{z}_B \end{pmatrix} = \begin{pmatrix} \boldsymbol{F}_I \\ \boldsymbol{F}_B \end{pmatrix}, \tag{8.2.13}$$

where the notation follows that introduced in Chapter 6. For example, G_I^k is the (diagonal) submatrix of G^k corresponding to nodes interior to the substructures. We can then reduce (8.2.13) to the Schur complement form

$$S\boldsymbol{z}_B = \boldsymbol{c} \tag{8.2.14}$$

where

$$S = \left(\lambda^2 K_{BB} + G_B^k\right) - \lambda^2 K_{IB}^T \left(\lambda^2 K_{II} + G_I^k\right)^{-1} \lambda^2 K_{IB}, \tag{8.2.15}$$

$$\boldsymbol{c} = \boldsymbol{F}_B - \lambda^2 K_{IB}^T \left(\lambda^2 K_{II} + G_I^k\right)^{-1} \boldsymbol{F}_I. \tag{8.2.16}$$

The fact that the quasi–Jacobian matrix, A, may be decomposed into a constant part, $\lambda^2 K$, and a diagonal part, G^k, that depends on $\boldsymbol{x}^k, \boldsymbol{y}^k$, is extremely useful in its construction on the MP–1. Firstly it means that K need only be constructed once and then, at each step of the quasi–Newton iteration, we need only consider the elements of $\boldsymbol{g}_\Psi$ evaluated at $\boldsymbol{x}^k$ and $\boldsymbol{y}^k$ in order to construct G^k. Secondly, when using a uniform mesh, the local stiffness matrices, $K^{(i)}$, will be identical on each substructure and hence only one copy need be constructed. This will greatly reduce storage overheads. A graded or non–uniform mesh will, of course, disallow this but the same philosophy may be applied to any group of

substructures that are identically triangulated. This may well occur in a refinement of an initially uniform mesh.

A flow chart indicating the algorithm used to solve the potential equation is given in Figure 8.3. In that diagram r_{up}, r_{low} denote the residuals produced by the present upper and lower solutions respectively. The solution of the Neumann problems to form the Schur complements is done by a CGM routine with a stopping tolerance of 10^{-7}. This ensures reasonable accuracy for these small, relatively well–conditioned problems. In practice we have found that the linear solves required in the quasi–Newton method are best performed without preconditioning. This is because the systems involved are strongly diagonally dominant and hence only require a few CG steps to converge. As we have already seen in Chapter 7, in this case the quickest solution method is not to precondition. This CG loop had a stopping criterion of 10^{-8}. The $(k+1)$th quasi–Newton iteration ceased when

$$\max\{\|(A(\boldsymbol{x}^k,\boldsymbol{y}^k))^{-1}\boldsymbol{F}(\boldsymbol{x}^k)\|_2, \|(A(\boldsymbol{x}^k,\boldsymbol{y}^k))^{-1}\boldsymbol{F}(\boldsymbol{y}^k)\|_2\} < 3\times 10^{-4}.$$

8.3 Solution of the continuity equations

Given any $\boldsymbol{\Psi}^{k+1}$, $\boldsymbol{V}^k$, $\boldsymbol{W}^k$, to solve the continuity equations (8.1.5), (8.1.6) we must construct the stiffness matrices arising from a finite element discretisation of (8.1.5), (8.1.6) incorporating the harmonic average of the exponential coefficient function. First recall the definition of the harmonic average: Given $X \in S_h(\Omega)$ we define $\overline{X} \in \Sigma_h(\Omega)$, given on each triangle T by

$$\exp(\overline{X}|_T) = \left\{\frac{1}{\mathcal{A}(T)}\int_T \exp(-X)\right\}^{-1}.$$

Hence given any vector $\boldsymbol{X}$ of nodal values of X, for each triangle we wish to calculate $\int_T \exp(-X)$. As $X \in S_h(\Omega)$ this can be easily done analytically. It is

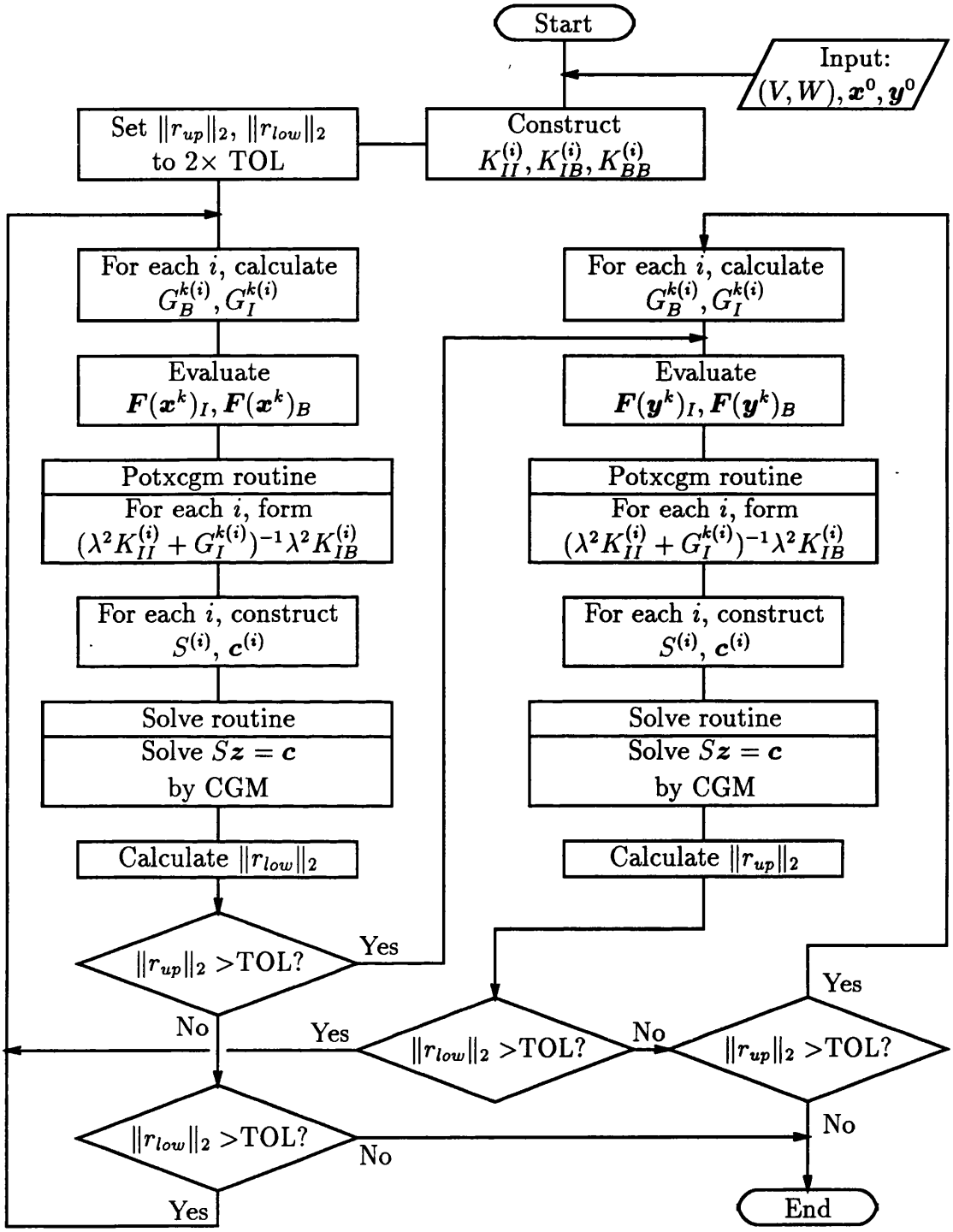


Figure 8.3: Potential equation solver.

most convenient to map each triangle T to the standard triangle K with vertices (0,0), (0,1) and (1,0) and perform the integration over K.

We denote the vertices of any triangle T as N_1, N_2, N_3 with coordinates $\boldsymbol{x}_1 = (x_{11}, x_{12})^T$, $\boldsymbol{x}_2 = (x_{21}, x_{22})^T$, $\boldsymbol{x}_3 = (x_{31}, x_{32})^T$ respectively. Then, referring to Figure 8.4, consider the mapping M from the standard triangle K to any triangle T in Ω given by

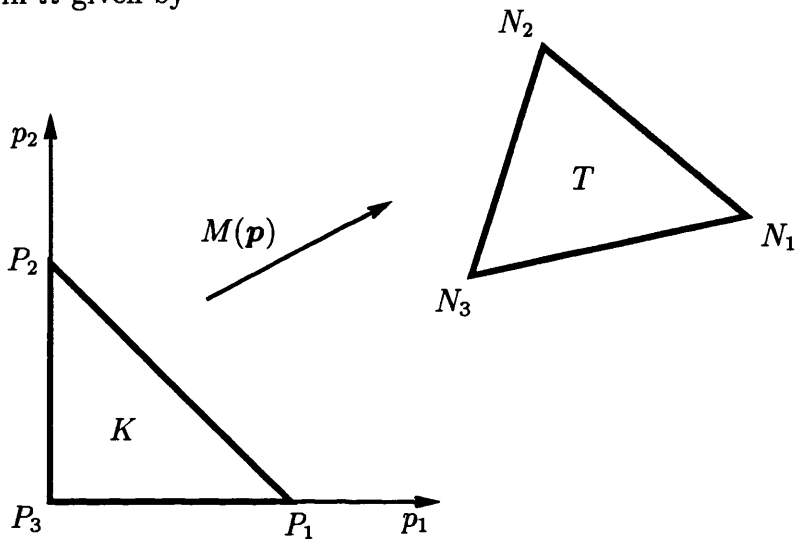


Figure 8.4: The mapping M.

$$M(\boldsymbol{p}) := p_1\boldsymbol{x}_1 + p_2\boldsymbol{x}_2 + (1 - p_1 - p_2)\boldsymbol{x}_3. \tag{8.3.17}$$

M takes K into T such that $P_i \rightarrow N_i$ for all i. Then, denoting the Jacobian of M as J we have,

$$J = \begin{bmatrix} x_{11} - x_{31} & x_{21} - x_{31} \\ x_{12} - x_{32} & x_{22} - x_{32} \end{bmatrix} \tag{8.3.18}$$

and hence

$$\begin{aligned} \det(J) &= \det \begin{bmatrix} 1 & x_{11} & x_{12} \\ 1 & x_{21} & x_{22} \\ 1 & x_{31} & x_{32} \end{bmatrix} \\ &= 2\mathcal{A}(T). \end{aligned} \tag{8.3.19}$$

Hence for any $X \in S_h(\Omega)$

$$\begin{aligned}\int_T \exp(-X(\boldsymbol{x}))\, dx_1 dx_2 &= \int_K 2\exp(-X(\boldsymbol{x}))\mathcal{A}(T)\, dp_1 dp_2 \\ &= 2\mathcal{A}(T)\int_K \exp(-X(M(\boldsymbol{p})))\, dp_1 dp_2\end{aligned}$$

Therefore at each new Gummel iterate we are left with two calculations of the form,

$$\int_K \exp(\Phi(M(\boldsymbol{p})))dp_1 dp_2 \quad \text{for each } T \text{ in } \Omega. \tag{8.3.20}$$

For the electron continuity equation $\Phi = V^k - \Psi^k$ and for the hole continuity equation $\Phi = \Psi^k - W^k$. If we denote the value of Φ at N_i by Φ_i, $i = 1, 2, 3$, then we have

$$\Phi(M(\boldsymbol{p})) = \Phi_1 p_1 + \Phi_2 p_2 + (1 - p_1 - p_2)\Phi_3 = \Phi_3 + p_1(\Phi_2 - \Phi_3) + p_2(\Phi_1 - \Phi_3).$$

If we now make the assumption that Φ_1, Φ_2, Φ_3 are pairwise disjoint then we have

$$\begin{aligned}&\int_T \exp(\Phi)d\boldsymbol{p} \\ &= \int_0^1 \int_0^{1-p_1} \exp(\Phi_3 + p_1(\Phi_2 - \Phi_3) + p_2(\Phi_1 - \Phi_3))\, dp_2\, dp_1 \\ &= \int_0^1 \left[\frac{1}{\Phi_1 - \Phi_3}\exp(\Phi_3 + p_1(\Phi_2 - \Phi_3) + p_2(\Phi_1 - \Phi_3))\right]_0^{1-p_1} dp_1 \\ &= \frac{1}{\Phi_1 - \Phi_3}\int_0^1 \exp(\Phi_1 + p_1(\Phi_2 - \Phi_1)) - \exp(\Phi_3 + p_1(\Phi_2 - \Phi_3))dp_1 \\ &= \frac{1}{\Phi_1 - \Phi_3}\left[\frac{1}{\Phi_2 - \Phi_1}\exp(\Phi_1 + p_1(\Phi_2 - \Phi_1))\right. \\ &\qquad\qquad \left. -\frac{1}{\Phi_2 - \Phi_3}\exp(\Phi_3 + p_1(\Phi_2 - \Phi_3))\right]_0^1 \\ &= \frac{1}{\Phi_1 - \Phi_3}\left(\frac{1}{\Phi_2 - \Phi_1}(\exp(\Phi_2) - \exp(\Phi_1)) - \frac{1}{\Phi_2 - \Phi_3}(\exp(\Phi_2) - \exp(\Phi_3))\right) \\ &= \frac{\exp(\Phi_1)}{(\Phi_1 - \Phi_2)(\Phi_1 - \Phi_3)} + \frac{\exp(\Phi_2)}{(\Phi_2 - \Phi_1)(\Phi_2 - \Phi_3)} + \frac{\exp(\Phi_3)}{(\Phi_3 - \Phi_1)(\Phi_3 - \Phi_2)}.\end{aligned}$$

If Φ attains the same value at any two of the three nodes of T we can express (8.3.20) in a similar way using an appropriate limit. Furthermore, to ensure accurate computation, we should also consider the cases when the difference in Φ

at any two nodes is very small as this will induce rounding error in our calculation of (8.3.20). This analysis was done in practice but the code produced was so long that our somewhat limited front end machine found it impossible to compile. Therefore we have made the approximation that whenever

$$|\Phi_i - \Phi_j| < 10^{-3} \text{ for } i,j = 1,2,3 \ \ i \neq j,$$

we set $\Phi_i = \Phi_j$ and use the appropriate calculation for (8.3.20). This does away with the need for complicated Taylor expansions but will also lead to some (small) inaccuracies. Recall from earlier numerical experiments in Chapter 4 that in many cases Φ is of the order of 18.

With the harmonic average on each triangle T computed and stored on the appropriate processor, stiffness matrix construction is then a simple task of constructing the element stiffness matrices for $K(0)$, multiplying each entry by its harmonic average, and then adding up to create the global stiffness matrix. This can then be reduced to the Schur complement matrix as we have done for our model problems in Chapter 7. A flow chart to illustrate this process is given in Figure 8.5

In contrast to the potential equation solver, it was found essential to use preconditioning on the solutions of the linearised continuity equations. As these are particularly ill–conditioned problems with large jumps in the coefficient functions expected interior to the domain, we have used the edge space plus coarse grid preconditioner (6.5.45) detailed in Chapter 6. This PCG loop was halted when the residual fell below 10^{-5}. This may seem a more generous stopping tolerance than that used for the linear solves in the potential equation, but it should be noted that here we are monitoring the preconditioned residual which is a (much) better estimate of the error.

Therefore the implementation of the continuity equation solver is identical to that given for the edge space plus coarse grid preconditioner in Chapter 7. It is the construction of the stiffness matrices that has required some additional work.

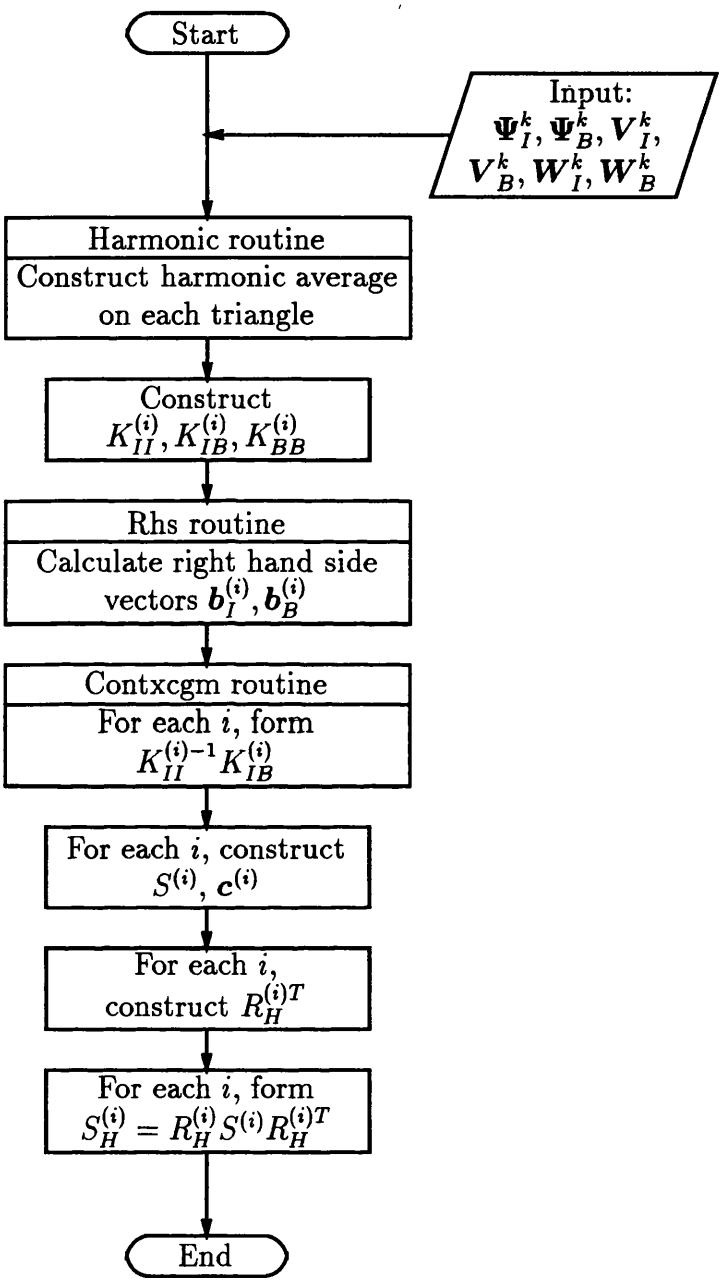


Figure 8.5: Stiffness matrix construction.

For this reason we do not give a flow chart for the solution of the continuity equation, referring the reader instead to the details given in Chapter 7.

Finally before we present some numerical results, we comment that the Gummel iteration was continued until the solutions at the kth step gave residuals less than their respective specified tolerances at the $(k+1)$th step.

8.4 Numerical experiments

Example 8.4.1 Small applied voltage (20mV)

In Table 8.1 below we give results obtained with the recombination rate, r in (8.1.2), (8.1.3), switched off. Here "time" refers to the wall clock time in seconds that each solver took. "E_{tot}" and "C_{tot}" are the total numbers of inner edge space and coarse grid iterations respectively."G it" is the Gummel iteration number.

The first thing to notice is that, in each case, after the first Gummel iteration the value of Ψ obtained was extremely close to the eventual solution and hence only a negligible amount of work was required in the quasi–Newton solver at subsequent Gummel iterates. For this reason we have not included solution times for the potential equation as this essentially constitutes a one–off overhead at the beginning of the Gummel process. As we can see the number of quasi–Newton iterates in the first Gummel iterate is unaffected by the value of $h = 1/m(n+1)$. For the potential equation in subsequent Gummel iterations we use the converged upper and lower solutions from the previous Gummel iterate as starting guesses. Although these cannot be guaranteed to be upper and lower starting values for the new potential equation, our new quasi–Newton method appears to be robust enough to cope with this.

Secondly note that the overall number of Gummel iterations required for convergence appears to be independent of H or h. The theoretical results of Chapter 5 tell us that, at worst, we can expect the convergence to deteriorate logarithmically with h. The times per linear solve, although reasonably fast for

such a difficult problem, are a little slower than the solve times from the model problems in Chapter 7. This is because the coefficients in the continuity equations do not now have simple jump discontinuities across substructure boundaries, but rather they exhibit a layer behaviour, where the variation may be smeared over several mesh widths. The preconditioner, (6.5.45), described in Chapter 6 therefore does not completely remove their influence. This is also reflected in the number of PCG iterates required for each linear solve. These can now be quite significant. This number also increases as h is decreased and is probably due to the improved resolution of the layers.

Finally for this example, in Table 8.2 we include results for the same problem with the recombination rate switched on. Obviously the number of quasi–Newton iterates required at the first Gummel step is unaffected, and again we see that this is independent of h as predicted theoretically in Chapter 3. We also see that, for this example near equilibrium, the recombination rate has no affect on the total number of Gummel iterations required, but so far a proof of this fact has eluded us. The extra work required to solve the systems with non–trivial right hand side is reflected in the increased number of PCG iterates and hence execution times.

Example 8.4.2 Increased applied voltage (100mV)

Table 8.3 shows results obtained with an increased applied voltage. The information has now been summarised, so that "Quasi–Newton its" is those required in the first Gummel iterate. Once again, the quasi–Newton iterates in subsequent Gummel iterations are negligible. "Time" now refers to the cumulative wall clock solve times for both the continuity equations.

The increased applied voltage has inflated the contraction constant for the Gummel mapping. This is in full concurrence with the theory of Chapter 5. Hence we now require 25 Gummel iterations for convergence when the recombination rate is switched off. We also see a very modest increase in the required quasi–Newton iterates, presumably because our initial guesses have altered in relation

to the solution. The overall increase in Gummel iterations has brought about an increased execution time also. Finally we see that the scheme still converges if we turn on the recombination rate, although in this case the required number of Gummel iterations and hence the execution time has modestly increased.

Table 8.4 gives results obtained from executing the code on the MP–2 available to us in Sunnyvale, California. This is a 16K machine, i.e. it has a 128×128 array of processors. Furthermore each processor has 64K of RAM. However, due to share arrangements, we are allocated a 1K segment of the PE array. The results again show that there is only a modest increase in the required number of iterations when the mesh size is decreased. It should be pointed out that, since the MP–2 incorporates different chips than the MP–1 the floating point arithmetic may be executed slightly differently, which in itself could result in a slight change in the number of iterations required. Although the timings show that the code runs reasonably efficiently on a larger machine, they should not be compared with the times obtained from the MP–1 as it was not possible to obtain solitary access to the MP–2. This meant that the code would occasionally be swapped out of the DPU, resulting in longer run–times.

Figures 8.6, 8.7 and 8.8 are plots of the converged potential Ψ, electron quasi–Fermi level V, and hole quasi–Fermi level W respectively. They are the solutions for the problem of the reverse bias diode with 100mV applied voltage, recombination switched on and $m = 16$, $n = 1$.

The plots are obtained by downloading the results file obtained from the MP–1 at Bath to a MATLAB code which outputs the results as a mesh plot of a matrix. Hence in Figures 8.6, 8.7 and 8.8 the mesh appears to be rectangular, but this is just an effect from the plotting routine. The solutions have also been rotated in these figures so that the point $(0,0)$ represents $(0,1)$ in the domain of computation.

In Figure 8.6 we see the sharp interior layer that is present in the electrostatic potential. In fact our rather coarse mesh does not resolve this layer very well at

all. We see that there is a need for mesh refinement about the doping interface. This would involve some significant alterations to the present code. We also see that the solution does not suffer any of the instabilities which would be expected if standard finite element methods were used to solve the semiconductor equations in natural variables. Figures 8.7 and 8.8 show the layer behaviour of the quasi–Fermi levels. Again there are regions of rapid change within the domain although this is somewhat more smeared than in the case of the electrostatic potential. We see that for this relatively small applied voltage example, the incorporation of a recombination term does not result in the loss of a maximum principle for the quasi–Fermi levels. However a proof of this observation has yet to be given.

Finally, in Figure 8.9 we have taken the trouble to reconstruct the electron current from the electrostatic potential and electron quasi–Fermi level shown in Figures 8.6 and 8.7 respectively. The current is computed as a post–processing step in the parallel code and then plotted using a MATLAB graphics routine. The domain is oriented as in Figure 8.1. We see that, despite the large variation in both Ψ and V, the resulting current is small and relatively smooth. This is consistent with the reverse bias configuration of the device. The slightly increased current about the point (0.25,1) is probably due to the rather coarse mesh and the singularity at that point. Again, some mesh refinement would be preferable. A similar plot for the hole current shows very little action anywhere in the domain and is therefore not included.

		G	Pot Eqn	Electron Cont Eqn				Hole Cont Eqn			
m	n	it	its	its	time	E_{tot}	C_{tot}	its	time	E_{tot}	C_{tot}
8	1	1	53	113	147	116	3820	44	45	45	1078
		2	4	25	25	26	581	28	30	29	711
		3	0	20	21	21	485	23	24	24	570
		4	0	10	10	11	239	13	13	14	316
		5	0	8	8	9	191	7	8	8	172
		6	0	1	2	2	42	2	3	3	63
		7	0	0	1	1	22	1	2	2	42
		8	0	0	1	1	22	0	1	1	21
Totals			57	177	215	187	5402	118	126	126	2973
16	1	1	54	207	463	210	13941	45	78	46	2226
		2	3	27	48	28	1331	29	52	30	1459
		3	0	21	36	22	994	23	41	24	1150
		4	0	11	19	12	529	9	16	10	451
		5	0	9	16	10	438	8	14	9	389
		6	0	1	3	2	89	1	3	2	87
		7	0	0	2	1	46	1	3	2	86
		8	0	0	2	1	45	0	2	1	42
Totals			57	276	589	286	17413	116	209	124	5890
32	1	1	55	238	1004	242	32513	46	140	47	4372
		2	2	66	204	67	6363	28	89	29	2801
		3	0	19	55	20	1697	23	71	24	2213
		4	0	5	16	6	497	7	22	8	690
		5	0	3	11	4	346	9	27	10	830
		6	0	1	6	2	177	1	5	2	149
		7	0	1	6	2	187	1	5	2	148
		8	0	0	3	1	94	0	2	1	72
Totals			57	333	1305	344	41874	115	361	123	11275

Table 8.1: Reverse bias applied voltage 20mV, recombination off.

		G	Pot Eqn	Electron Cont Eqn				Hole Cont Eqn			
m	n	it	its	its	time	E_{tot}	C_{tot}	its	time	E_{tot}	C_{tot}
16	1	1	54	207	467	210	13941	100	185	101	5288
		2	3	115	207	116	5911	28	50	29	1409
		3	0	21	36	22	1001	23	41	24	1160
		4	0	10	18	11	488	9	16	10	444
		5	0	7	13	8	350	9	16	10	443
		6	0	1	3	2	90	1	3	2	88
		7	0	1	3	2	90	1	3	2	86
		8	0	0	2	1	45	0	2	1	43
Totals			57	276	749	372	21916	171	316	179	8961
16	2	1	54	222	626	451	14719	81	196	164	4242
		2	3	65	157	132	3392	28	68	58	1440
		3	0	20	47	42	957	22	53	46	1109
		4	0	12	29	26	583	13	32	28	656
		5	0	8	20	18	401	8	20	18	415
		6	0	1	4	4	91	1	4	4	91
		7	0	0	2	2	46	1	4	4	90
		8	0	0	2	2	46	0	2	2	44
Totals			57	328	887	677	20235	154	379	160	8087

Table 8.2: Reverse bias applied voltage 20mV, recombination on.

m	n	Quasi–Newton its	Gummel its	Time (s)	r
16	1	61	25	2284	off
16	1	61	27	3006	on

Table 8.3: Reverse bias applied voltage 100mV.

m	n	**Quasi–Newton its**	**Gummel its**	**Time (s)**	r
32	1	61	30	2512	off
32	1	61	28	3856	on

Table 8.4: Reverse bias applied voltage 100mV, execution at Sunnyvale.

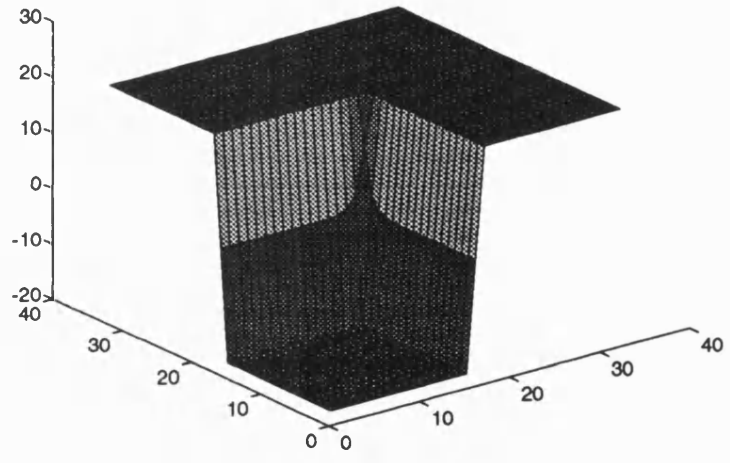


Figure 8.6: Electrostatic potential, Ψ, 100mV applied voltage.

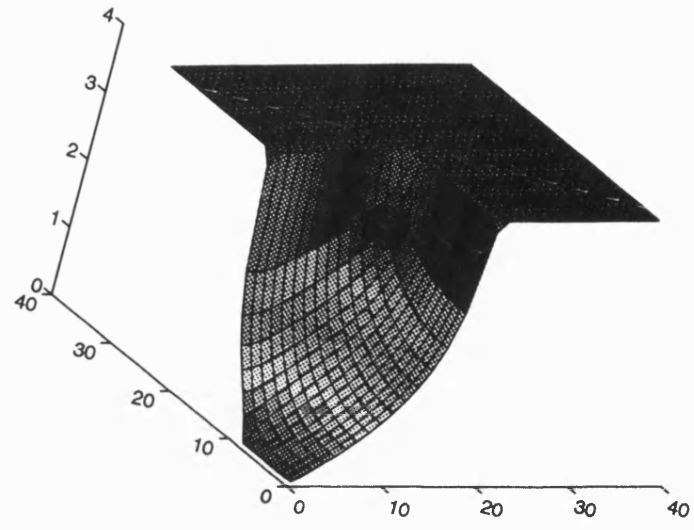


Figure 8.7: Electron quasi–Fermi level, V, 100mV applied voltage.

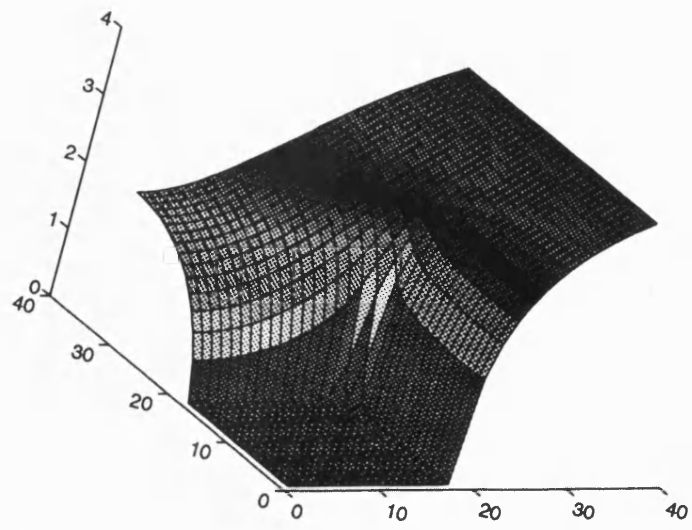


Figure 8.8: Hole quasi–Fermi level, W, 100mV applied voltage.

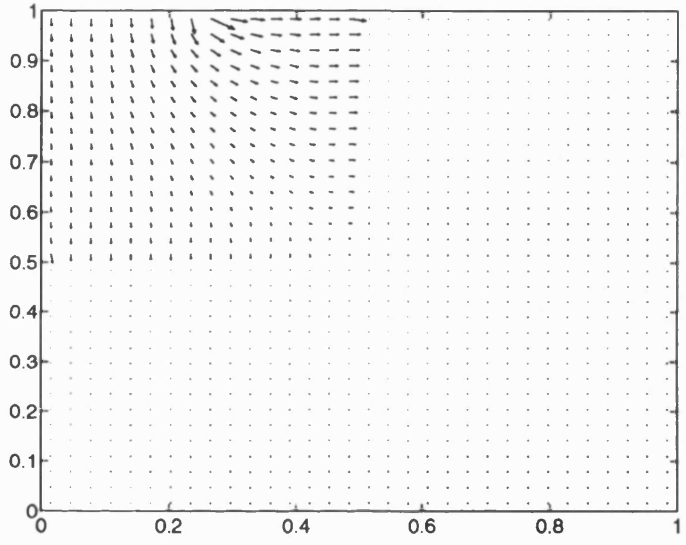


Figure 8.9: Electron current, 100mV applied voltage.

Appendix A

Miscellaneous results

A.1 Bounding the 2–norm of K^{-1}

We require an upper bound for $||K^{-1}||_2$ defined in Chapter 3. First recall since K is symmetric positive definite we have

$$||K^{-1}||_2 = \rho(K^{-1}) = \lambda_{\max}(K^{-1}) = 1/\lambda_{\min}(K).$$

Hence, bounding $||K^{-1}||_2$ above is equivalent to bounding $\lambda_{\min}(K)$ below. We do this via the following lemma.

LEMMA A.1.1 *Assuming our mesh satisfies the assumption (2.2.8), then there exists constants c and C only depending on γ_1 such that for all $V \in S_h(\Lambda)$, $V = \sum_i v_i\phi_i$ we have*

$$ch|\boldsymbol{v}|^2 \leq ||V||^2_{L_2(\Lambda)} \leq Ch|\boldsymbol{v}|^2, \tag{A.1.1}$$

where $|.|$ denotes the 2–norm on $\mathbb{R}^n$.

Proof It is sufficient to show that for each interval, $I_i = [x_{i-1}, x_i]$, $i = 1, \ldots, n+1$, and each $V \in P_1(I_i)$ we have

$$ch_i(v_{i-1}^2 + v_i^2) \leq ||V||^2_{l_2(I_i)} \leq Ch_i(v_{i-1}^2 + v_i^2), \tag{A.1.2}$$

where $P_1(I_i)$ denotes the space of linear functions on I_i. Result (A.1.1) then follows directly from (A.1.2) by summation over all intervals.

Firstly we show (A.1.2) holds when $I_i = [0,1]$. Let λ_1, λ_2 be the basis functions for $P_1([0,1])$ and define

$$\begin{aligned} f_1(\boldsymbol{\eta}) &= \int_0^1 (\hat{V})^2 dx \\ f_2(\boldsymbol{\eta}) &= \eta_1^2 + \eta_2^2 \end{aligned}$$

where $\boldsymbol{\eta} = (\eta_1, \eta_2)^T$ and $\hat{V}(x) = \eta_1\lambda_1(x) + \eta_2\lambda_2(x)$. Note that f_1 and f_2 are continuous functions of $\boldsymbol{\eta} \in \mathbb{R}^2$. Now consider the quotient

$$f_3(\boldsymbol{\eta}) = \frac{f_1(\boldsymbol{\eta})}{f_2(\boldsymbol{\eta})}, \quad \boldsymbol{\eta} \in \mathbb{R}^2, \quad \boldsymbol{\eta} \neq \mathbf{0}.$$

We wish to prove that there is a constant C such that

$$f_3(\boldsymbol{\eta}) \leq C, \quad \boldsymbol{\eta} \in \mathbb{R}^2, \quad \boldsymbol{\eta} \neq \mathbf{0}. \tag{A.1.3}$$

This corresponds to the right–hand inequality in (A.1.2) when $I_i = [0,1]$. To do this note that

$$f_3(\mu\boldsymbol{\eta}) = f_3(\boldsymbol{\eta}), \quad \text{for all} \quad \mu \in \mathbb{R}, \quad \mu \neq 0,$$

i.e. f_3 is homogeneous of degree zero. Hence it is sufficient to prove that

$$f_3(\hat{\boldsymbol{\eta}}) \leq C, \quad \hat{\boldsymbol{\eta}} \in B, \quad B = \left\{\boldsymbol{\eta} \in \mathbb{R}^2 : |\boldsymbol{\eta}| = 1\right\}. \tag{A.1.4}$$

But f_3 is continuous on B (in particular $f_2 \neq 0$ for $\boldsymbol{\eta} \in B$) and B is closed and bounded in $\mathbb{R}^2$, and thus f_3 has a maximum on B. This gives us (A.1.4) and thus (A.1.3) and (A.1.2) in the case $I_i = [0,1]$. The left–hand inequality in (A.1.2) can be proved similarly for the case $I_i = [0,1]$.

We now show the right–hand inequality in (A.1.2) for an arbitrary interval $[x_{i-1}, x_i]$. The left–hand inequality is analogous. Consider the mapping F which takes our unit interval to $[x_{i-1}, x_i]$, and set

$$x = F(t) = x_{i-1} + th_i, \quad t \in [0,1].$$

Then, given $V \in P_1(I_i)$ we have

$$V(x) = V(F(t)) := \hat{V}(t), \quad t \in [0,1],$$

where clearly $\hat{V} \in P_1([0,1])$ and

$$\begin{aligned} \int_{x_{i-1}}^{x_i} (V(x))^2 dx = \int_{x_{i-1}}^{x_i} (\hat{V}(t))^2 dx &= \int_0^1 h_i (\hat{V}(t))^2 dt \\ &\leq Ch_i(\eta_1^2 + \eta_2^2) = Ch_i(v_{i-1}^2 + v_i^2) \end{aligned}$$

as required. ■

COROLLARY A.1.2 $\lambda_{min}(K) \geq ch$

Proof Recall that if $V = \sum_i v_i \phi_i \in S_h(\Lambda)$ then

$$a(V,V) := \int_0^1 (V')^2 dx = \boldsymbol{v}^T K \boldsymbol{v}.$$

Hence by ellipticity of $a(\cdot,\cdot)$ on $H_0^1(\Lambda)$ and Lemma A.1.1,

$$\frac{\boldsymbol{v}^T K \boldsymbol{v}}{|\boldsymbol{v}|^2} = \frac{a(V,V)}{|\boldsymbol{v}|^2} \geq \frac{c||V||_2^2}{|\boldsymbol{v}|^2} \geq ch,$$

Hence by the Rayleigh Quotient Theorem we have

$$\lambda_{\min}(K) \geq ch$$

■

A.2 Properties of φ

LEMMA A.2.1 *Let $\varphi(x) = (e^x - 1)/x$, $x \in \mathbb{R}$. Then $\varphi : \mathbb{R} \longmapsto \mathbb{R}$ is positive, monotonic increasing and convex on $\mathbb{R}$.*

Proof We first show that φ is positive and monotonically increasing. Note that $\varphi(x) \to \infty$ as $x \to \infty$ and $\varphi(x) \to 0$ as $x \to -\infty$. Also

$$\varphi'(x) = ((x-1)e^x + 1)/x^2.$$

Therefore $\varphi'(x) = 0$ if and only if $e^x = 1/(1-x)$ and

$$\varphi'(x) \to \infty \quad \text{as} \quad x \to \infty,$$
$$\varphi'(x) \to 0 \quad \text{as} \quad x \to -\infty,$$

However if $1 > x > 0$ then

$$e^x = \sum_{k=0}^{\infty} \frac{x^k}{k!} < \sum_{k=0}^{\infty} x^k = (1-x)^{-1}.$$

Also, if $x > 1$, then $1/(1-x) < 0 < e^x$. Finally, if $x < 0$ and $e^x = (1-x)^{-1}$ then consequently

$$e^{-x} = 1 + (-x) + \frac{(-x)^2}{2!} + \ldots,$$

a contradiction. So $e^x = (1-x)^{-1}$ has a single route at $x = 0$. Hence $\varphi'(x)$ can only be zero at $x = 0$. But expanding φ' as a power series we obtain

$$\begin{aligned} \varphi'(x) &= \left((x-1)\sum_{j=0}^{\infty}\frac{x^j}{j!} + 1\right)/x^2 = \left(\sum_{j=0}^{\infty}\frac{x^{j+1}}{j!} - \sum_{j=1}^{\infty}\frac{x^j}{j!}\right)/x^2 \\ &= ((x-x) + (x^2 - x^2/2) + \mathcal{O}(x^3))/x^2 \to 1/2 \text{ as } x \to 0. \end{aligned}$$

So $\varphi'(x) > 0$ for all x and hence $\varphi : \mathbb{R} \longmapsto \mathbb{R}$ is positive and monotonically increasing. Finally we show that φ is convex on $\mathbb{R}$.

$$\varphi(x) = \frac{1}{x}\left(\sum_{j=1}^{\infty}\frac{x^j}{j!}\right) = \sum_{j=0}^{\infty}\frac{1}{(j+1)}\frac{x^j}{j!}.$$

So

$$\varphi' = \sum_{j=1}^{\infty}\frac{1}{(j+1)}\frac{x^{j-1}}{(j-1)!} = \sum_{j=0}^{\infty}\frac{1}{(j+2)}\frac{x^j}{j!}.$$
$$\varphi'' = \sum_{j=2}^{\infty}\frac{1}{(j+1)}\frac{x^{j-2}}{(j-2)!} = \sum_{j=0}^{\infty}\frac{1}{(j+3)}\frac{x^j}{j!}. \qquad \text{(A.2.5)}$$

So $\varphi'' > 0$ on $x \geq 0$ and hence φ is convex on $x \geq 0$. Also $\varphi'(x) = ((x-1)e^x + 1)/x^2$ and so

$$\varphi''(x) = \frac{x^2((x-1)e^x + e^x) - ((x-1)e^x + 1)2x}{x^4} = \frac{e^x(x^2 - 2x + 2) - 2}{x^3}$$

Now by (A.2.5) $\varphi''(x) = 1/3$ at $x = 0$ and by the above expression $\varphi''(x) \to 0^+$ as $x \to -\infty$. Hence $\varphi''(x) > 0,\ x \in (-\infty, 0)$ unless $\varphi''(x) = 0$ for some $x \in (-\infty, 0)$. However $\varphi'' = 0$ if and only if $e^x = 2/(x^2 - 2x + 2)$ for some $x < 0$ which occurs if and only if

$$e^{-x} = 1 - x + x^2/2 \text{ for some } x < 0. \tag{A.2.6}$$

Putting $y = -x$ then (A.2.6) is true if and only if

$$e^y = 1 + y + y^2/2 \text{ for some } y > 0. \tag{A.2.7}$$

But this is a contradiction. Hence $\varphi'' > 0,\ x \in \mathbb{R}$ and thus φ is convex. ■

Bibliography

[1] I. BABUSKA and J.E. OSBORNE. Generalised finite element methods : Their performance and their relation to mixed methods. *SIAM. J. Numer. Anal.*, 20:510–536, 1983.

[2] R. E. BANK. *PLTMG: A Software Package for Solving Elliptic Partial Differential Equations, User's Guide 6.0.* SIAM, Philadelphia, 1990.

[3] R.E. BANK, T. DUPONT, and H. YSERENTANT. The hierarchical basis multigrid method. *Numer. Math.*, 52:427–458, 1988.

[4] R.E. BANK, D.J. ROSE, and W. FICHTNER. Numerical methods for semiconductor device simulation. *Siam. J. Sci. Stat. Comp.*, 4:416–435, 1983.

[5] P.E. BJØRSTAD and M.D. SKOGEN. A new, massively parallel algorithm for the solution of elliptic equations with discontinuous coefficients. Technical report, University of Bergen, 1991.

[6] P.E. BJØRSTAD and M.D. SKOGEN. Domain decomposition algorithms of Schwarz type designed massively parallel computers. Technical report, University of Bergen, March 1991.

[7] P.E. BJØRSTAD and O.B. WIDLUND. Iterative methods for the solution of elliptic problems on regions partitioned into substructures. *Siam. J. Numer. Anal*, 23:1097–1120, 1986.

[8] J.H. BRAMBLE, J.E. PASCIAK, and A.H. SCHATZ. An iterative method for elliptic problems on regions partitioned into substructures. *Math. Comp.*, 46:361–369, 1986.

[9] F. BREZZI, P. CAPELO, and L. GASTALDI. A singular perturbation analysis of reverse-biased semiconductor diodes . *Siam. J. Numer. Anal*, 20:372–387, 1989.

[10] F. BREZZI, L.D. MARINI, and P. CAPELO. Two-dimensional exponential fitting and application to drift-diffusion models . *Siam. J. Numer. Anal*, 26:1342–1355, 1989.

[11] F. BREZZI, L.D. MARINI, and P. PIETRA. Numerical simulation of semiconductor devices . *Comp. Meth. Appl. Mech. Engrg.*, 75:493–514, 1989.

[12] T. CHAN, R. GLOWINSKI, J. PÉRIAUX, and O. B. WIDLUND, editors. *Second International Symposium on Domain Decomposition Methods for Partial Differential Equations.* SIAM, 1988.

[13] P.G. CIARLET. *The Finite Element Method for Elliptic Problems.* North Holland, Amsterdam, 1978.

[14] R. K. COOMER and I. G. GRAHAM. Domain decomposition methods for device modelling. Technical Report (To appear in the proceedings of the Seventh International Conference on Domain Decomposition Methods in Science and Engineering, Penn State University, October, 1993), University of Bath, 1993.

[15] R. K. COOMER and I. G. GRAHAM. Massively parallel methods for semiconductor device modelling. Technical Report Mathematics Preprint Number 93/02 (submitted for publication), University of Bath, 1993.

[16] P.M. DE ZEEUW. Nonlinear multigrid applied to a 1D stationary semiconductor model. *SIAM J. Sci. Stat. Comp*, 13:512–530, 1992.

[17] P. DEUFLHARD and F.A. POTRA. Asymptotic mesh independence of Newton-Galerkin methods via a refined Mysovskii theorem. *SIAM J. Numer. Anal.*, 29:1395–1412, 1992.

[18] M. DRYJA, B.J. SMITH, and O.B. WIDLUND. Schwartz analysis of iterative substructuring algorithms for elliptic problems in three dimensions. Technical Report 638, Courant Institute of Mathematical Sciences, New York, June 1993.

[19] M. DRYJA and O.B. WIDLUND. Some domain decomposition algorithms for elliptic problems. In L. Hayes and D. Kincaid., editor, *Iterative methods for large linear systems.* Academic Press, Orlando, Florida, 1989.

[20] M. DRYJA and O.B. WIDLUND. Multilevel additive methods for elliptic finite element problems. In W. Hackbusch, editor, *Parallel algorithms for PDEs.* Vieweg, Braunschweig, Germany, 1991.

[21] G. A. DUFFET and M. S. TOWERS. *EVEREST doping generator*, 1990.

[22] N. FERGUSON, C. J. FITZSIMONS, M. S. TOWERS, J. V. ASHBY, and C. GREENOUGH. *EVEREST pre-processor*, 1990.

[23] W.F. FICHTNER, D.J. ROSE, and R.E. BANK. Semiconductor device simulation. *Siam. J. Sci. Stat. Comp.*, 4:391–415, 1983.

[24] R. GLOWINSKI, G. H. GOLUB, G. A. MEURANT, and J. PÉRIAUX, editors. *First International Symposium on Domain Decomposition Methods for Partial Differential Equations.* SIAM, 1987.

[25] G.H. GOLUB and C.F. VAN LOAN. *Matrix Computations, Second Edition.* John Hopkins Press, London, 1989.

[26] A. GRAHAM. *Nonnegative Matrices and Applicable Topics in Linear Algebra.* Ellis Horwood Limited, Chichester, 1987.

[27] A. GREENBAUM, C. LI, and H.Z. CHAO. Parallelizing preconditioned conjugate gradient algorithms. *Computer Physics Communications*, 53:295–309, 1989.

[28] H.K. GUMMEL. A self-consistent iterative scheme for one-dimensional steady state transistor calculations . *IEEE Transactions on Electron Devices*, ED-11:455–465, 1964.

[29] D. GUNASEKERA, R. F. FOWLER, and C. GREENOUGH. *EVEREST solver module*, 1990.

[30] D. GUNASEKERA, R.F. FOWLER, and C. GREENOUGH. Everest Solver Module, User Manual, Version 3.0 . Technical report, Rutherford Appleton Laboratory, November 1990.

[31] D. GUNASEKERA and C. GREENOUGH. *Comprehensive testing of the solver module*, 1988.

[32] D. GUNASEKERA and C. GREENOUGH. *Transient benchmark tests*, 1989.

[33] W. HACKBUSCH. *Multi–grid methods and applications.* Springer series in computational mathematics : 4. Springer-Verlag, 1985.

[34] C. A. HALL and T. A. PORSCHING. *Numerical Analysis of Partial Differential Equations.* Prentice Hall, New Jersey, 1990.

[35] P.R. HALMOS. *Finite–Dimensional Vector Spaces.* Springer–Verlag, Wien - New York, 1978.

[36] P.W. HEMKER. A nonlinear multigrid method for one–dimensional semiconductor device simulation: results for the diode. *J. Computational and Applied Mathematics*, 30:117–126, 1990.

[37] J.W. JEROME. Consistency of semiconductor modelling: an existence/stability analysis for the stationary Van Roosbroeck system. *SIAM. J. Appl. Math.*, 45(4):565–590, 1985.

[38] J.W. JEROME and T. KERKHOVEN. A finite element approximation theory for the drift diffusion semiconductor model. *SIAM J. Numer. Anal.*, 28(2):403–422, 1991.

[39] C. JOHNSON. *Numerical Solutions of Partial Differential Equations by the Finite Element Method.* Cambridge University Press, Cambridge, 1987.

[40] T. KERKHOVEN. *Coupled and decoupled algorithms for semiconductor simulation.* PhD thesis, Department of Computer Science, Yale University, 1985. (Yale Research Report YALEU/DCS/RR-429).

[41] T. KERKHOVEN. On the dependence of the convergence of Gummel's algorithm on the regularity of the solution. Technical Report YALEU/DCS/RR-366, Department of Computer Science, Yale University, March 1985.

[42] T. KERKHOVEN. On the effectiveness of Gummel's method. *SIAM J. Sci. Stat. Comput.*, 9(1):48–60, 1988.

[43] T. KERKHOVEN. A spectral analysis of the decoupling algorithm for semiconductor simulation. *SIAM J. Numer. Anal.*, 25(6):1299–1312, 1988.

[44] D. E. KEYES and W. D. GROPP. A comparison of domain decomposition techniques for elliptic partial differential equations and their parallel implementation. *SIAM J. Sci. Stat. Comput.*, 8(2), 1987.

[45] P.L. LIONS. On the Schwarz alternating method I. In R. Glowinski, G.H. Golub, G.A. Meurant and J. Périaux., editor, *First International Symposium on Domain Decomposition Methods for Partial Differential Equations.* SIAM, Philadelphia, 1988.

[46] J. MANDEL. Hybrid domain decomposition with unstructured subdomains. *Preprint*, 1993.

[47] P. A. MARKOWICH and C. A. RINGHOFER. A singulalry perturbed boundary value problem modelling a semiconductor device. *SIAM J. Numer. Anal.*, 26:507–516, 1989.

[48] P.A. MARKOWICH. *The Stationary Semiconductor Device Equations.* Springer, Wien - New York, 1986.

[49] P.A. MARKOWICH, C.A. RINGHOFER, and C. SCHMEISER. *Semiconductor Equations.* Springer, Wien - New York, 1990.

[50] P.A. MARKOWICH and M.A. ZLÁMAL. Inverse-average-type finite element discretizations of selfadjoint second-order elliptic problem. *Mathematics of Computation*, 51(184):431–449, 1988.

[51] MasPar Computer Corporation. *MasPar FORTRAN programming manuals.*

[52] MasPar Computer Corporation. *MasPar system overview.*

[53] P. A. MAWBY, M. S. TOWERS, G. A. DUFFET, J. ZHANG, and G. J. HUANG. *EVEREST post–processor*, 1990.

[54] M. METCALF and J. REID. *Fortran90 explained.* Oxford Science Publications, Oxford, 1990.

[55] J. MILLMAN and A. GRABEL. *Microelectronics.* McGraw–Hill, New York, London, 1987.

[56] M.S. MOCK. *Analysis of Mathematical Models of Semiconductor Devices.* Boole Press, Dublin, 1983.

[57] J. MOLENAAR. *Multigrid methods for semiconductor device simulation.* PhD thesis, Center for Mathematics and Computer Science, Amsterdam, 1992.

[58] J. MOLENAAR and P.W. HEMKER. A multigrid approach for the solution of the 2D semiconductor equations. Technical Report NM-R9003, Centre for Mathematics and Computer Science, Amsterdam, February 1990.

[59] C. MOLER, J. LITTLE, and S. BANGERT. *PRO-MATLAB User's Guide.* Mathworks Inc, South Natick, Mass., 1987.

[60] J.M. ORTEGA and W.C. RHEINBOLDT. *Iterative Solution of Nonlinear Equations in Several Variables.* Academic Press, New York, 1970.

[61] C.P. PLEASE. An analysis of semiconductor P-N junctions. *IMA J. App. Math.*, 28:301–318, 1982.

[62] S.J. POLAK, C. DEN HEIJER, W.H.A. SCHILDERS, and P. MARKOWICH. Semiconductor device modelling from the numerical point of view. *Internat. J. Numer. Meth. Engrg.*, 24:763–838, 1987.

[63] C.A. RINGHOFER and C. SCHMEISER. An approximate Newton method for the solution of the basic semiconductor device equations. *SIAM J. Numer. Anal.*, 1989.

[64] W. SCHILDERS. Initial guess strategies and extrapolation techniques for use in the Gummel method. *Preprint, Philips Research Laboratories, Eindhoven, Netherlands*, 1991.

[65] B.F. SMITH. A domain decomposition algorithm for elliptic problems in three dimensions. *Numer. Math.*, 60:219–234, 1991.

[66] B.F. SMITH. An optimal domain decomposition preconditioner for the finite element solution of linear elasticity problems. *Siam. J. Sci. Stat. Comput.*, 13, no.1:364–378, 1992.

[67] B.F. SMITH and O.B. WIDLUND. A domain decomposition algorithm using a hierarchical basis. *Siam. J. Sci. Stat. Comput.*, 11, no.6:1212–1220, 1990.

[68] J. J. SPARKES. *Semiconductor Devices.* Van Nostrand Reinhold (International), London, 1987.

[69] G. STRANG. Approximation in the finite element method. *Numer. Math.*, 19:81–98, 1972.

[70] S. M. SZE. *Physics of Semiconductor Devices, 2nd ed.* Wiley, New York, 1981.

[71] R. S. VARGA. *Matrix Iterative Analysis.* Prentice Hall, Englewood Cliffs, 1962.

[72] J. XU. Iterative methods by space decomposition and subspace correction. *SIAM Review*, 34:581–613, 1992.

[73] E. S. YANG. *Microelectronic Devices.* McGraw–Hill, New York, London, 1988.

[74] H. YSERENTANT. On the multi-level splitting of finite element spaces. *Numer. Math.*, 49:379–412, 1986.