Efficient Schrödinger-Poisson Solvers for Quasi 1D Systems That Utilize PETSc and SLEPc

by

Pranay Kumar Reddy Baikadi

A Thesis Presented in Partial Fulfillment
of the Requirements for the Degree
Master of Science

Approved November 2020 by the
Graduate Supervisory Committee:

Dragica Vasileska, Chair
Mykhailo Povolotskyi
Stephen Goodnick

ARIZONA STATE UNIVERSITY

December 2020

ABSTRACT

The quest to find efficient algorithms to numerically solve differential equations is ubiquitous in all branches of computational science. A natural approach to address this problem is to try all possible algorithms to solve the differential equation and choose the one that is satisfactory to one's needs. However, the vast variety of algorithms in place makes this an extremely time consuming task. Additionally, even after choosing the algorithm to be used, the style of programming is not guaranteed to result in the most efficient algorithm. This thesis attempts to address the same problem but pertinent to the field of computational nanoelectronics, by using PETSc linear solver and SLEPc eigenvalue solver packages to efficiently solve Schrödinger and Poisson equations self-consistently.

In this work, quasi 1D nanowire fabricated in the GaN material system is considered as a prototypical example. Special attention is placed on the proper description of the heterostructure device, the polarization charges and accurate treatment of the free surfaces. Simulation results are presented for the conduction band profiles, the electron density and the energy eigenvalues/eigenvectors of the occupied sub-bands for this quasi 1D nanowire. The simulation results suggest that the solver is very efficient and can be successfully used for the analysis of any device with two dimensional confinement. The tool is ported on www.nanoHUB.org and as such is freely available.

Dedicated to my parents and teachers

ACKNOWLEDGEMENTS

I would like to first express my deep gratitude to my advisor Professor Dragica Vasileska for introducing me the rich field of Computational Electronics and admitting me to her research group. She has been a constant source of guidance throughout my Masters studies here at Arizona State University. Her unparalleled patience and encouragement helped me to overcome many roadblocks and this work would not be possible without her support. I am highly indebted to Dr. Michael Povolotskyi for his invaluable help and suggestions during the meetings, which shaped the course of this project. I would like to extend my sincere thanks to Professor Stephen Goodnick for being my thesis committee member. His EEE 539 course helped me immensely in solidifying my understanding of semiconductor physics.

I would like to thank my group members, Viswanathan Naveen for all the insights and discussions during the project and Izak Baranowski, Ziyi Wang for creating a great research environment in the group. I always learnt something new from their research in the weekly group meetings.

Finally, I thank my parents B.Janardhan Reddy, K.Vinodha Kumari for their eternal support and for taking the right decisions at the right times which ultimately shaped my career.

TABLE OF CONTENTS

CHAPTER Page

LIST OF TABLES

LIST OF FIGURES

Figure Page

Chapter 1

INTRODUCTION

1.1 PETSc and SLEPc: Building Effective Computational Electronics Tools

The continuous effort to keep pace with the predictions of the Moore's Law has resulted in the computer resources becoming considerably cheaper. This has resulted in many branches of basic sciences adopting computational tools to bridge the gap between theory and experiment. This has been especially true in the semiconductor industry, where the constant device scaling has introduced new technologies accompanied by new phenomena. For example, as the transistor feature size of the current generation of transistors is well into the nanometer regime [1, 2], the de-Broglie wavelength of the electron is comparable to the device dimensions and quantum mechanical effects start to play a critical role in the operation of these devices. Fabrication of nanoscale devices is accompanied by an increasingly intricate and time-consuming manufacturing process which is extremely expensive. Computational tools, validated with experiments, offer a cost-effective alternative to tackle this problem, as they accurately capture the underlying physics of these devices. Thus they became an indispensable tool to the semiconductor industry. They also provide the ability to test and predict the behaviour of hypothetical structures, before they can be fabricated at high volume.

To be able to reliably use these tools, the numerical engines on which they are based should be efficient and robust in solving the differential equations describing the phenomena. This problem of finding the best algorithms to numerically solve differential equations at hand is prevalent in all branches of computational science. One trivial

approach would be to experiment with various algorithms and choose the one that satisfies a set of predetermined criteria, such as the error tolerance, computational time etc. However, this approach would naturally be a highly time consuming endeavour. Moreover, it also demands thorough understanding and expertise in subtle nuances of many fields tangential to one's area of research. With the vast variety of numerical methods already in place, combined with the increasing pace at which new methods are added to the existing framework, it becomes very difficult to make an accurate choice of the numerical algorithm. In this work, an attempt is made to address a problem specific to the field of computational nanoelectronics, using the PETSc [3] linear solver and SLEPc eigenvalue solver packages [4]. The Portable, Extensible Toolkit for Scientific Computation (PETSc), developed by Argonne National Laboratory, is a collection of data structures and numerical routines for solving various partial differential equations describing scientific phenomena. It provides the user with a vast collection of efficiently programmed, ready-to-use numerical methods and preconditioners, thus offering the user enormous flexibility in experimenting with various algorithms to solve a system of linear equations. Developed on similar lines by researchers from Universitat Politècnica de València, the Scalable Library for Eigenvalue Problem Computation (SLEPc), is a software package for the solution of large sparse eigenvalue problems. SLEPc is built on top of PETSc and is often considered as an extension of PETSc, using the same programming paradigm. Over the years, both PETSc and SLEPc have been successfully used in a number of scientific applications in various fields such as geological sciences, computational fluid dynamics, medical biology [5, 6, 7], etc.

1.2 Need for Efficient Two Dimensional Schrödinger-Poisson Solvers for Lateral GaN FETs

Nitride semiconductors have emerged as a strong candidate for high power, high temperature and high frequency applications in the recent years [8, 9, 10]. Even though the effective mass of gallium nitride is three times larger than gallium arsenide and therefore results in low low-field mobility of bulk gallium nitride, large band gaps, high peak velocity, large saturation velocity and high thermal stability make them ideal material for channel in microwave devices. The strongest feature of III-V materials is the heterostructure technology it can support - quantum well, modulation doped hetero interface, and heterojunction structure can all be made in this material system.

Two dimensional electron densities in the order of $10^{13}\ cm^{-3}$ or higher can be achieved in GaN HEMTs owing to its large piezoelectric polarization charge that arises due to strain of the top layers. The polarization charge in these devices is about five times larger in comparison to GaAs HEMT structures. In addition, the spontaneous polarization, which is an inherent property of the material, is very high for GaN and AlN material systems. The electric fields produced by these charges are in the order of 2-5 MV/cm. These high electric fields are responsible for the very high two-dimensional electron densities in these devices [11].

Similar to today's Si processor technology, 3D GaN FETs offer multi-gate structures that provide excellent electrostatic control over the channel and enable very low subthreshold swing values close to the theoretical limit [12]. Various concepts have been demonstrated, including both lateral and vertical devices with GaN nanowire (NW) or nanofin (NF) geometries [13]. Outstanding transport properties were achieved

with laterally contacted NWs that were grown in a bottom-up approach and transferred onto an insulating substrate [12]. For higher power application, vertical FETs based on regular arrays of GaN nanostructures are particularly promising due to their parallel integration capability and large sidewall surfaces, which can be utilized as channel area. In GaN nanowire FETs two dimensional confinement plays a significant role on the operation of the transistors fabricated in this technology. Therefore, there is a need for development of efficient two dimensional Schrödinger-Poisson solvers for these material systems. For that purpose, PETSc and SLEPc are used in this work to develop two dimensional Schrödinger-Poisson solver for lateral GaN nanowire FETs.

1.3 Outline of the Thesis

The organization of the thesis is as follows.

1. Chapter 2 discusses the PETSc linear solver package and a two dimensional Poisson solver is developed. The equilibrium potential profiles for AlGaN-GaN and AlGaN-AlN-GaN heterostructures are compared with Silvaco-ATLAS.

2. Chapter 3 discusses the SLEPc eigenvalue solver package and the results from the two dimensional Schrödinger solver are presented.

3. In Chpater 4, the Schrödinger and Poisson solver are coupled self-consistently and the energy eigenvalues/eigenvectors, electron density profile are shown.

4. Lastly, conclusions and future directions of research are presented in Chapter 5.

Chapter 2

PETSC AND THE POISSON SOLVER

The Poisson equation, given by Eq.(2.1), is a second order elliptic partial differential equation, which establishes the relationship of electric field and electric charges in the device. The quantity ρ is the total charge density, which is sum of free charge density(ρ_f) and the bound charge density(ρ_b), at a point $\mathbf{r}$ in the system.

$$\nabla.(\epsilon(\mathbf{r})\nabla V(\mathbf{r})) = \rho(\mathbf{r}) \tag{2.1}$$

Solving the Poisson equation is a crucial part of any device simulation. It provides the electrostatic potential profile in the device at a given time, which might be used later in the device simulation process. For example, for a Monte Carlo based transport simulation of a device, the Poisson solver would be called after each time step for the entire simulation time scale. An inefficient Poisson solver can, thus, lead to a severe computational bottleneck in terms of the runtime of the simulation. This chapter introduces the PETSc linear solver package, which allows the user to experiment with a vast variety of preprogrammed algorithms, thus giving the flexibility to choose an algorithm which best meets ones needs.

Since the purpose of this work is development of Schrödginer-Poisson solvers for GaN based semiconductor devices, this chapter starts with a discussion on the polarization charges in III-V Nitrides. This is next followed by a section on band parameter approach for heterostructures. Next, linearization and discretization of the Poisson equation is presented. Then the PETSc linear solver package is introduced detailing a few important subroutines useful for programming in PETSc. Finally, the results

from PETSc Poisson solver are compared with Silvaco-ATLAS, establishing the validity of the Poisson solver developed as part of this work.

2.1 Polarization in III-V Nitrides

2.1.1 The Generalized Hooke's Law

To understand the Piezoelectric polarization in a material, one has to first understand the stress-strain tensors and the generalized Hooke's law. In 3D, the state of stress at any point in a solid is given by a rank-2 tensor called the Cauchy stress tensor(σ_{ij}) shown below.

$$\sigma = \begin{bmatrix} \sigma_{xx} & \sigma_{xy} & \sigma_{xz} \\ \sigma_{yx} & \sigma_{yy} & \sigma_{yz} \\ \sigma_{zx} & \sigma_{zy} & \sigma_{zz} \end{bmatrix} \tag{2.2}$$

In general, the element σ_{ij} can be interpreted as the stress acting on the surface at a point with unit normal along the $\hat{\mathbf{e}}_i$ due to the component F_j of the force $\mathbf{F}$. Under mechanical equilibrium, the stress tensor is symmetric. Then in Eq.(2.2) $\sigma_{xy} = \sigma_{yx}$, $\sigma_{xz} = \sigma_{zx}$ and $\sigma_{yz} = \sigma_{zy}$. Thus, the state of stress at point in a solid can be represented by 6 independent components.

The strain at any point is representative of the deformation of the solid at that point and is the rate of change of displacement with distance. If the effect of rotation is neglected, the strain tensor ϵ in 3D can be represented as a symmetric part of the deformation tensor as given below.

$$\epsilon = \begin{bmatrix} \epsilon_{xx} & \epsilon_{xy} & \epsilon_{xz} \\ \epsilon_{yx} & \epsilon_{yy} & \epsilon_{yz} \\ \epsilon_{zx} & \epsilon_{zy} & \epsilon_{zz} \end{bmatrix} \tag{2.3}$$

In the above equation $\epsilon_{ij} = \frac{1}{2}(e_{ij} + e_{ji})$ where $e_{ij} = \frac{\partial u_i}{\partial x_j}$. Then in Eq.(2.3) $\epsilon_{xy} = \epsilon_{yx}$, $\epsilon_{xz} = \epsilon_{zx}$ and $\epsilon_{yz} = \epsilon_{zy}$ and thus, the state of strain at a point in a solid can also be

represented by 6 independent components. In the strain tensor, $\epsilon_{xx}, \epsilon_{yy}, \epsilon_{zz}$ represent the extensions per unit length parallel to the x,y and z axes respectively. e_{xy} is the rotation about the z axis towards the x axis of a line element parallel to the y axis. Other components can be interpreted similarly. Using Eq.(2.2) and Eq.(2.3) the generalized Hooke's law is given as follows.

$$\boxed{\sigma_{ij} = C_{ijkl}\epsilon_{kl}} \tag{2.4}$$

where i,j,k,l $\in$ {x,y,z}. The quantity C_{ijkl} is a rank-4 tensor and is called the stiffness tensor. It is clear that the stiffness tensor in general has 81 components. However, following the symmetry arguments outlined below, the number of independent components can be reduced.

1. **Symmetry of the Stress Tensor:**

   The stress tensor, as mentioned above, is symmetric under mechanical equilibrium. Thus $\sigma_{ij} = \sigma_{ji}$ and hence $C_{ijkl} = C_{jikl}$. The number of independent components in now reduced to 54(=6*3*3).

2. **Symmetry of Strain Tensor:**

   The strain tensor is also symmetric by definition. Thus $\epsilon_{ij} = \epsilon_{ji}$ and hence $C_{ijkl} = C_{ijlk}$. The number of independent components in now reduced to 36(=6*6).

3. **Equivalence of Mixed Partials:** To further reduce the number of independent components, it is useful to remind oneself that for a given value of strain, the strain energy density is given as:

   $$\Delta = \frac{1}{2} C_{ijkl}\epsilon_{ij}\epsilon_{kl}$$

   Then the stress tensor can be represented as

   $$\sigma_{ij} = \frac{\partial \Delta}{\partial \epsilon_{ij}} = C_{ijkl}\epsilon_{kl} \tag{2.5}$$

Differentiating Eq.(2.5) with respect to ϵ_{mn} gives:

$$\begin{aligned}\frac{\partial^2 \Delta}{\partial \epsilon_{ij} \partial \epsilon_{mn}} &= \frac{\partial}{\partial \epsilon_{mn}} \left(\frac{\partial \Delta}{\partial \epsilon_{ij}} \right) \\ &= \frac{\partial}{\partial \epsilon_{mn}} \left(C_{ijkl} \epsilon_{kl} \right) \\ &= C_{ijkl} \frac{\partial \epsilon_{kl}}{\partial \epsilon_{mn}} \\ &= C_{ijkl} \delta_{km} \delta_{ln} \\ &= C_{ijmn}\end{aligned} \tag{2.6}$$

If the differentiation is carried out with respect to ϵ_{mn} first, we get:

$$\begin{aligned}\frac{\partial^2 \Delta}{\partial \epsilon_{ij} \partial \epsilon_{mn}} &= \frac{\partial}{\partial \epsilon_{ij}} \left(\frac{\partial \Delta}{\partial \epsilon_{mn}} \right) \\ &= \frac{\partial}{\partial \epsilon_{ij}} \left(C_{ijkl} \epsilon_{kl} \frac{\partial \epsilon_{ij}}{\partial \epsilon_{mn}} \right) \\ &= C_{ijkl} \delta_{ki} \delta_{lj} \delta_{im} \delta_{jn} \\ &= \delta_{ki} \delta_{lj} \left(C_{ijkl} \delta_{im} \delta_{jn} \right) \\ &= \delta_{ki} \delta_{lj} C_{mnkl} \\ &= C_{mnij}\end{aligned} \tag{2.7}$$

Thus with the use of equivalence of mixed partials, it is clear that $C_{ijkl} = C_{klij}$. This further brings down the total number of independent components of C_{ijkl} to 21.

With 6 independent stress and strain tensor components, the matrix formulation of the generalized Hooke's law is represented below with the independent components

of the stiffness tensor labelled in blue.

$$\begin{bmatrix} \sigma_{xx} \\ \sigma_{yy} \\ \sigma_{zz} \\ \sigma_{yz} \\ \sigma_{xz} \\ \sigma_{xy} \end{bmatrix} = \begin{bmatrix} C_{xxxx} & C_{xxyy} & C_{xxzz} & C_{xxyz} & C_{xxxz} & C_{xxxy} \\ C_{yyxx} & C_{yyyy} & C_{yyzz} & C_{yyyz} & C_{yyxz} & C_{yyxy} \\ C_{zzxx} & C_{zzyy} & C_{zzzz} & C_{zzyz} & C_{zzxz} & C_{zzxy} \\ C_{yzxx} & C_{yzyy} & C_{yzzz} & C_{yzyz} & C_{yzxz} & C_{yzxy} \\ C_{xzxx} & C_{xzyy} & C_{xzzz} & C_{xzyz} & C_{xzxz} & C_{xzxy} \\ C_{xyxx} & C_{xyyy} & C_{xyzz} & C_{xyyz} & C_{xyxz} & C_{xyxy} \end{bmatrix} \begin{bmatrix} \epsilon_{xx} \\ \epsilon_{yy} \\ \epsilon_{zz} \\ 2\epsilon_{yz} \\ 2\epsilon_{xz} \\ 2\epsilon_{xy} \end{bmatrix} \tag{2.8}$$

An alternative matrix notation of the Hooke's law represented below, called the Voigt notation, is more frequently used in the literature.

$$\begin{bmatrix} \sigma_1 \\ \sigma_2 \\ \sigma_3 \\ \sigma_4 \\ \sigma_5 \\ \sigma_6 \end{bmatrix} = \begin{bmatrix} C_{11} & C_{12} & C_{13} & C_{14} & C_{15} & C_{16} \\ C_{21} & C_{22} & C_{23} & C_{24} & C_{25} & C_{26} \\ C_{31} & C_{32} & C_{33} & C_{34} & C_{35} & C_{36} \\ C_{41} & C_{42} & C_{43} & C_{44} & C_{45} & C_{46} \\ C_{51} & C_{52} & C_{53} & C_{54} & C_{55} & C_{56} \\ C_{61} & C_{62} & C_{63} & C_{64} & C_{65} & C_{66} \end{bmatrix} \begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ \epsilon_3 \\ 2\epsilon_4 \\ 2\epsilon_5 \\ 2\epsilon_6 \end{bmatrix} \tag{2.9}$$

The Voigt notation uses the following representation of the indices:

$$\begin{aligned} 1 &\rightarrow xx \\ 2 &\rightarrow yy \\ 3 &\rightarrow zz \\ 4 &\rightarrow yz \\ 5 &\rightarrow xz \\ 6 &\rightarrow xy \end{aligned}$$

The number of independent components of the stiffness tensor can be further reduced by utilizing the inherent symmetries of the solid. The III-V nitrides belong to the

C_{6v} crystallographic point group [14] and the corresponding stiffness tensor is given as follows:

$$C = \begin{bmatrix} C_{11} & C_{12} & C_{13} & 0 & 0 & 0 \\ C_{12} & C_{11} & C_{13} & 0 & 0 & 0 \\ C_{13} & C_{13} & C_{33} & 0 & 0 & 0 \\ 0 & 0 & 0 & C_{44} & 0 & 0 \\ 0 & 0 & 0 & 0 & C_{44} & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{1}{2}(C_{11} - C_{12}) \end{bmatrix} \tag{2.10}$$

2.1.2 Piezoelectric Polarization

If stress is applied to certain crystal, they develop a dipole moment which is proportional to the applied stress as given below:

$$P^{pz} = d\sigma \tag{2.11}$$

The constant d is called the piezoelectric modulus and is a rank-3 tensor in 3D with $d_{ijk} = d_{ikj}$. When a general stress σ_{jk} is applied to the crystal, the component P_i of the piezoelectric polarization is given as,

$$P_i^{pz} = d_{ijk}\sigma_{jk} \tag{2.12}$$

For example, if a uniaxial tensile stress is applied along the x-direction (σ_{xx}), the components of the piezoelectric polarization can be written as,

$$P_x^{pz} = d_{xxx}\sigma_{xx}, \quad P_y^{pz} = d_{yxx}\sigma_{xx}, \quad P_z^{pz} = d_{zxx}\sigma_{xx}$$

A rank-3 tensor in general will have 27 independent components. However, if the j and k symmetry of the d_{ijk} tensor is taken into account, the number of independent

components reduces to 18. Following the Voigt notation described in the previously, the matrix representation of the piezoelectric modulus tensor is given as follows:

$$d = \begin{bmatrix} d_{11} & d_{12} & d_{13} & d_{14} & d_{15} & d_{16} \\ d_{21} & d_{22} & d_{23} & d_{24} & d_{25} & d_{26} \\ d_{31} & d_{32} & d_{33} & d_{34} & d_{35} & d_{36} \end{bmatrix} \tag{2.13}$$

Similar to the stiffness tensor, internal crystal symmetries reduce the number of independent components of the piezoelectric modulus. For the C_{6v} point group, the d matrix has the following form.

$$d = \begin{bmatrix} 0 & 0 & 0 & 0 & d_{15} & 0 \\ 0 & 0 & 0 & d_{24}(= d_{15}) & 0 & 0 \\ d_{31} & d_{32}(= d_{31}) & d_{33} & 0 & 0 & 0 \end{bmatrix} \tag{2.14}$$

The piezoelectric polarization given in Eq.(2.12) can also be written in terms of strain as shown below.

$$\begin{aligned} P_i^{pz} &= d_{ijk}\sigma_{jk} \quad i,j,k \in \{1,2,3\} \\ &= d_{ijk}C_{jklm}\epsilon_{lm} \quad i,j,k,l,m \in \{1,2,3\} \\ &= d_{ij}C_{jk}\epsilon_k \quad i \in \{1,2,3\}; j,k \in 1,2,3,4,5,6 \\ &= e_{ik}\epsilon_k \end{aligned} \tag{2.15}$$

where $e_{ik} = d_{ij}C_{jk}$ in the Voigt notation. The constants e_{ik} are called the piezoelectric constants and more often than the stress formulation, this strain formulation of the piezoelectric polarization is used in the literature. The definition of piezoelectric

polarization in Eq.(2.12) can be written as follows.

$$\begin{aligned}
P_1^{pz} &= d_{11}\sigma_1 + d_{12}\sigma_2 + d_{13}\sigma_3 + d_{14}\sigma_4 + d_{15}\sigma_5 + d_{16}\sigma_6 \\
&= d_{15}\sigma_5 \\
P_2^{pz} &= d_{21}\sigma_1 + d_{22}\sigma_2 + d_{23}\sigma_3 + d_{24}\sigma_4 + d_{25}\sigma_5 + d_{26}\sigma_6 \\
&= d_{24}\sigma_4 \\
&= d_{15}\sigma_4 \\
P_3^{pz} &= d_{31}\sigma_1 + d_{32}\sigma_2 + d_{33}\sigma_3 + d_{34}\sigma_4 + d_{35}\sigma_5 + d_{36}\sigma_6 \\
&= d_{31}\sigma_1 + d_{32}\sigma_2 + d_{33}\sigma_3 \\
&= d_{31}(\sigma_1 + \sigma_2) + d_{33}\sigma_3
\end{aligned} \tag{2.16}$$

For the case of III-V nitrides, the growth direction is generally along [0001]. The crystal does not experience any stress in the growth direction and the shear stresses are negligible. Thus $\sigma_1 = \sigma_2$, $\sigma_3 = 0$ and $\sigma_4 = \sigma_5$. Thus the piezoelectric polarization will have a component only along the growth direction as given below.

$$\boxed{P_3^{pz} = 2d_{31}\sigma_1} \tag{2.17}$$

To calculate σ_1 in the above equation, it is useful to observe that since σ_3 is zero, we have the following result from Eq.(2.10).

$$\begin{aligned}
C_{31}\epsilon_1 + C_{32}\epsilon_2 + C_{33}\epsilon_3 + C_{34}\epsilon_4 + C_{35}\epsilon_5 + C_{36}\epsilon_6 &= 0 \\
2C_{13}\epsilon_1 + C_{33}\epsilon_3 &= 0 \\
\epsilon_3 &= -2\frac{C_{13}}{C_{33}}\epsilon_1
\end{aligned} \tag{2.18}$$

Now using Hooke's law and Eq.(2.18), σ_1 can be expanded as follows:

$$\begin{aligned}
\sigma_1 &= C_{11}\epsilon_1 + C_{12}\epsilon_2 + C_{13}\epsilon_3 + C_{14}\epsilon_4 + C_{15}\epsilon_5 + C_{16}\epsilon_6 \\
&= \epsilon_1(C_{11} + C_{12}) + C_{13}\epsilon_3 \\
&= \epsilon_1\left(C_{11} + C_{12} - 2\frac{C_{13}^2}{C_{33}}\right)
\end{aligned} \tag{2.19}$$

Similar to the piezoelectric moduli, the piezoelectric constants also have the same set of independent indices and they are related to the piezoelectric moduli using Eq.(2.15) as given below.

$$\begin{aligned}
e_{15} &= d_{11}C_{15} + d_{12}C_{25} + d_{13}C_{35} + d_{14}C_{45} + d_{15}C_{55} + d_{16}C_{65} \\
&= d_{15}C_{44} \\
e_{24} &= d_{21}C_{14} + d_{22}C_{24} + d_{23}C_{34} + d_{24}C_{44} + d_{25}C_{54} + d_{26}C_{64} \\
&= d_{24}C_{44} \\
&= d_{15}C_{44} = e_{15} \\
e_{31} &= d_{31}C_{11} + d_{32}C_{21} + d_{33}C_{31} + d_{34}C_{41} + d_{35}C_{51} + d_{36}C_{61} \\
&= d_{31}(C_{11} + C_{12}) + d_{33}C_{13} \\
e_{32} &= d_{31}C_{12} + d_{32}C_{22} + d_{33}C_{32} + d_{34}C_{42} + d_{35}C_{52} + d_{36}C_{62} \\
&= d_{31}(C_{12} + C_{11}) + d_{33}C_{13} = e_{31} \\
e_{33} &= d_{31}C_{13} + d_{32}C_{23} + d_{33}C_{33} + d_{34}C_{43} + d_{35}C_{53} + d_{36}C_{63} \\
&= 2d_{31}C_{13} + d_{33}C_{33}
\end{aligned} \tag{2.20}$$

Using the strain formulation of the Piezoelectric polarization in Eq.(2.15) and Eq.(2.18), we have:

$$\begin{aligned}
P_3^{pz} &= e_{31}\epsilon_1 + e_{32}\epsilon_2 + e_{33}\epsilon_3 + e_{34}\epsilon_4 + e_{35}\epsilon_5 + e_{36}\epsilon_6 \\
&= 2e_{31}\epsilon_1 + e_{33}\epsilon_3 \\
&= 2\epsilon_1 \left(e_{31} - e_{33}\frac{C_{13}}{C_{33}} \right)
\end{aligned} \tag{2.21}$$

The piezoelectric constants and elastic moduli for Group III-nitrides is given in the Table (2.1)[15].

Parameter	AlN	GaN	InN
P_{sp} $(\frac{C}{m^2})$	-0.081	-0.029	-0.032
e_{33} $(\frac{C}{m^2})$	1.46	0.73	0.97
e_{31} $(\frac{C}{m^2})$	-0.60	-0.49	-0.57
C_{13} (GPa)	120	70	1
C_{33} (GPa)	395	379	182

Table 2.1: Experimental Values of Piezoelectric Constants and Elastic Moduli of III-V Nitrides.

Along with the piezoelectric polarization, the III-V nitrides also exhibit spontaneous polarization, which is attributed to the fact that the geometric centers of negative and positive charges along the [0001] axis do not coincide due to the ionic nature of Ga(Al)-N bond. The positive direction of spontaneous polarization is always in the direction from Ga(Al) to N, and the negative values of the spontaneous polarizations for all III-V nitrides signifies that the dipole associated with the polarization is from N to Ga(Al) atom. The direction of piezoelectric polarization depends on whether the material under consideration is under tensile or compressive strain. As is evident from the Table (2.1), the term $(e_{31} - e_{33}\frac{C_{13}}{C_{33}})$ is always negative, the direction of piezoelectric polarization would be parallel to spontaneous polarization if the material is under tensile strain and anti-parallel to spontaneous polarization if the material is under compressive strain.

The total polarization in GaN based materials is represented in general, as a sum of the spontaneous and piezoelectric polarization charge densities. Due to the difference of the values of the spontaneous and piezoelectric polarization charges in the two materials, net sheet polarization charge density occurs at the hetero interfaces, which

has to be accounted for via the charge density term of the Poisson equation. For example, at the GaN-AlGaN hetero interface, the net polarization charge density σ is given by

$$\begin{aligned} \sigma &= \mathbf{P}^{bot}.\hat{n} - \mathbf{P}^{top}.\hat{n} \\ &= P_{sp}^{bot} + P_{pz}^{bot} - P_{sp}^{top} - P_{pz}^{top} \end{aligned} \tag{2.22}$$

If the material is relaxed, then the piezoelectric polarization charge density is zero. An example of the total polarization charge density at the hetero interface in a AlGaN-AlN-GaN material system is shown in the Figure (2.1).

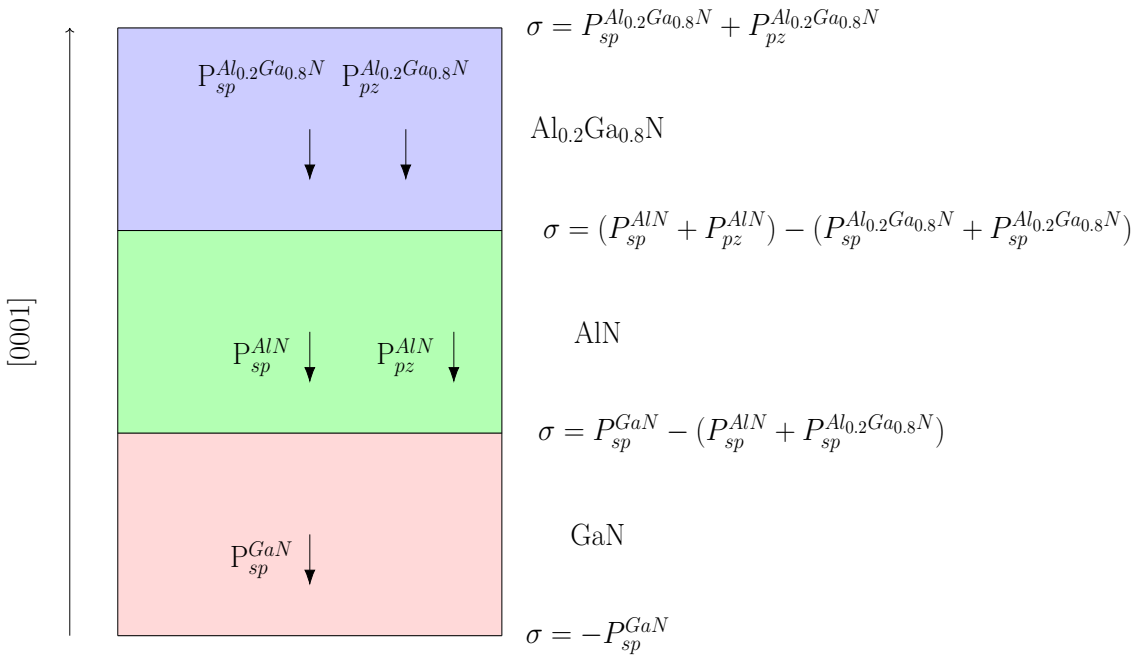

Figure 2.1: Spontaneous and Piezoelectric Polarizations in Psuedomorphically Grown Ga-face AlGaN-AlN-GaN Heterostructure System.

2.2 The Band Parameter Approach for Heterostructures

2.2.1 The Local Vacuum Level

Prior to the thorough investigations on heterostructure devices, the intrinsic level E_i was thought of as a correct measure to track the potential in the device [16]. However, this identification is incorrect and leads to erroneous conclusions when applied to heterostructures, as pointed out by Marshak [17]. Instead, an energy level called the local vacuum level E_l is constructed as shown in the Figure 2.2.

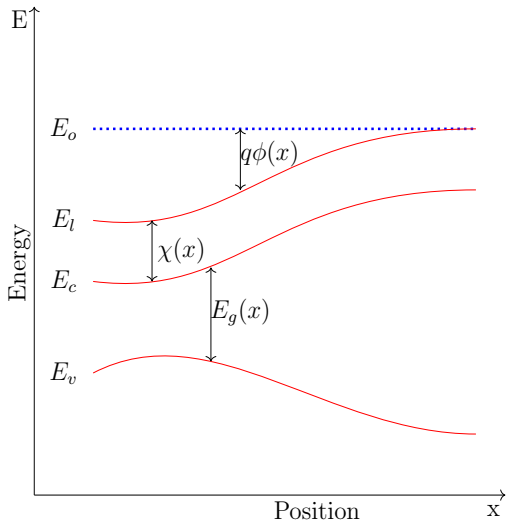

Figure 2.2: Energy Band Diagram of a Hypothetical Device with Position Dependent Parameters. E_l Denotes the Local Vacuum Level and E_c, E_v Denote the Conduction and Valence Bands Respectively.

The Local Vacuum Level E_l at a point is defined as the energy of an electron if it were at rest and free from the influence of the crystal potential. It is important to recognize that the local vacuum level E_l is different from the vacuum level at infinity, E_{vac}^{∞}, which is the reference energy of the electron at rest when situated far away from the semiconductor, so that it is completely unaware of the existence of the semicon-

ductor. In contrast, when the electron is at the local vacuum level, it is free from the influence of the microscopic crystal potential, but not from the potentials due to the electrostatic field.

The concept of local vacuum level becomes intuitive when one considers the example of electron liberation in a hypothetical P-N homojunction. Assuming that the surfaces are ideal, when the electrons are liberated from the conduction band into vacuum, there must be a difference in energies of the two electrons. This is because, the electron on the N-side of the homojunction has a energy which is lower by qV_{bi} as compared to the electron on the P-side. Hence, it is apparent that the local vacuum level must follow the electrostatic potential in the device. Thus from the Figure (2.2),

$$E_l(x) = E_o - qV(x) \tag{2.23}$$

where E_o is any reference level. The conduction band profile $E_c(x)$ and the valence band profile $E_v(x)$ are consequently given by

$$\begin{aligned} E_c(x) &= E_l(x) - \chi(x) \\ &= E_o - qV(x) - \chi(x) \\ E_v(x) &= E_c(x) - E_g(x) \\ &= E_o - qV(x) - \chi(x) - E_g(x) \end{aligned} \tag{2.24}$$

Using Eq.(2.24) and the general expression for the intrinsic level $E_i(x)$, one arrives at the following result:

$$\begin{aligned} E_i(x) &= \frac{E_c(x) + E_v(x)}{2} + \frac{k_b T}{2} ln\left(\frac{N_v(x)}{N_c(x)}\right) \\ &= E_o - \chi(x) - qV(x) - \frac{E_g(x)}{2} + \frac{k_b T}{2} ln\left(\frac{N_v(x)}{N_c(x)}\right) \end{aligned} \tag{2.25}$$

Analysing the result given in Eq.(2.25), it is evident that $E_i(x)$ is in general not parallel to $V(x)$ and hence is not an appropriate measure for the electrostatic potential

in the devices with position dependent material parameters. Except for the case of a homostructure, the intrinsic Fermi level E_i would be discontinuous at the interface between two different materials.

2.2.2 Equations for Band parameters

The appropriate theory for analysing heterostructures comes from the analysis of heavily doped semiconductors [18][19]. In a heavily doped semiconductor, the dopant atoms are close enough to one another and the wavefunctions of the neighbouring atoms overlap and the impurity levels broaden into a band that merges with the conduction or valence band [19]. This naturally alters the band structure of the semiconductor. As the semiconductor is heavily doped, Fermi-Dirac statistics must be used. The electron concentration n is then given by the following expression.

$$n = \int_{E_c}^{E_{c_{top}}} g_c(E - E_c, x) f_{FD}(E, E_{fn}) dE \tag{2.26}$$

where E_{fn} is the electron quasi Fermi level. It can further be shown that the electron concentration n at a given temperature T depends on the position x and the parameter η_c defined as

$$\eta_c = \frac{E_{fn} - E_c}{k_b T} \tag{2.27}$$

The electron concentration n can thus be written as follows:

$$n = n(x, \eta_c) \tag{2.28}$$

Using chain rule, the above equation becomes

$$\begin{aligned} \nabla n &= \nabla^{\eta_c} n + \frac{\partial n}{\partial \eta_c} \nabla \eta_c \\ &= \nabla^{\eta_c} n + \frac{\partial n}{\partial \eta_c} \nabla \left(\frac{E_{fn} - E_c}{k_b T} \right) \end{aligned} \tag{2.29}$$

where $\nabla^{\eta_c} n$ is the gradient of electron concentration with η_c held constant. From Eq.(2.29), it follows that the gradient of the electron quasi Fermi level is

$$\nabla E_{fn} = k_b T \left(\frac{\nabla n - \nabla^{\eta_c} n}{\frac{\partial n}{\partial \eta_c}} \right) + \nabla E_c \tag{2.30}$$

The gradient of the electron quasi Fermi level is related to the electron current density as,

$$\begin{aligned} J_n &= n\mu_n \nabla E_{fn} \\ &= n\mu_n k_b T \left(\frac{\nabla n}{\frac{\partial n}{\partial \eta_c}} \right) - n\mu_n k_b T \left(\frac{\nabla^{\eta_c} n}{\frac{\partial n}{\partial \eta_c}} \right) + n\mu_n \nabla E_c \\ &= qD_n \nabla n - n\mu_n k_b T \left(\frac{\nabla^{\eta_c} n}{\frac{\partial n}{\partial \eta_c}} \right) + n\mu_n \nabla E_c \end{aligned} \tag{2.31}$$

where generalized Einstein relation is used to arrive at the final result shown in Eq.(2.31). In Eq.(2.29), the first term accounts for the position dependence of density of states. Hence, a new parameter Γ_n is defined as

$$\nabla \Gamma_n = k_b T \left(\frac{\nabla^{\eta_c} n}{\frac{\partial n}{\partial \eta_c}} \right) \tag{2.32}$$

The electron current density can then be rewritten as

$$\begin{aligned} J_n &= qD_n \nabla n - n\mu_n \nabla \Gamma_n + n\mu_n \nabla E_c \\ &= n\mu_n \nabla (E_c - \Gamma_n) + qD_n \nabla n \end{aligned} \tag{2.33}$$

The first term on the RHS of Eq.(2.33) takes into account the position dependence of the density of states. The effect of Fermi-Dirac statistics is accounted for by modifying the second term of Eq.(2.31) by using Einstein relation as

$$\begin{aligned} qD_n \nabla n &= k_b T \mu_n \left(\frac{n}{\frac{\partial n}{\partial \eta_c}} - 1 + 1 \right) \nabla n \\ &= k_b T \mu_n \left(\frac{n}{\frac{\partial n}{\partial \eta_c}} - 1 \right) \nabla n + k_b T \mu_n \nabla n \end{aligned} \tag{2.34}$$

In Eq.(2.34), the first term accounts for dependence of electron concentration on Fermi-Dirac statistics. Now, a single parameter θ_n is defined which takes into account the effects of both Fermi-Dirac statistics and position dependence of density of states function.

$$\begin{aligned}\nabla\theta_n &= \nabla\Gamma_n - \frac{1}{n\mu_n}k_bT\mu_n\left(\frac{n}{\frac{\partial n}{\partial\eta_c}} - 1\right)\nabla n \\ &= \nabla\Gamma_n - \frac{1}{n\mu_n}(qD_n - k_bT\mu_n)\nabla n\end{aligned} \tag{2.35}$$

The expression for the current density equation can then be rewritten as follows.

$$\begin{aligned}J_n &= n\mu_n\nabla(E_c - \theta_n) + k_bT\mu_n\nabla n \\ &= n\mu_n\nabla(E_o - qV(x) - \chi(x) - \theta_n(x)) + k_bT\mu_n\nabla n \\ &= -qn\mu_n\nabla\left(V(x) + \frac{\chi(x) + \theta_n(x)}{q}\right) + k_bT\mu_n\nabla n\end{aligned} \tag{2.36}$$

One can arrive at the expression for $\theta_n(x)$ by substituting Eq.(2.32) into Eq.(2.35). This gives

$$\begin{aligned}\nabla\theta_n &= k_bT\left(\frac{\nabla^{\eta_c}n}{\frac{\partial n}{\partial\eta_c}}\right) - \frac{1}{n\mu_n}k_bT\mu_n\left(\frac{n}{\frac{\partial n}{\partial\eta_c}} - 1\right)\nabla n \\ &= k_bT\left(\frac{\nabla^{\eta_c}n}{\frac{\partial n}{\partial\eta_c}}\right) - \frac{k_bT}{\frac{\partial n}{\partial\eta_c}}\nabla n + k_bT\frac{\nabla n}{n} \\ &= k_bT\left(\frac{\nabla n - \frac{\partial n}{\partial\eta_c}\nabla\eta_c}{\frac{\partial n}{\partial\eta_c}}\right) - \frac{k_bT}{\frac{\partial n}{\partial\eta_c}}\nabla n + k_bT\frac{\nabla n}{n} \\ &= k_bT\frac{\nabla n}{n} - k_bT\nabla\eta_c\end{aligned} \tag{2.37}$$

Integrating Eq.(2.37) on both sides gives,

$$\theta_n(x) - \theta_n^{ref} = k_bT\ln\left(\frac{n(x)}{n_{ref}}\right) - k_bT(\eta_c - \eta_c^{ref}) \tag{2.38}$$

where *ref* is a reference point in a non-degenerate material, which is a natural consequence of integrating Eq.(2.37). For non-degenerate material, the effect of Fermi-Dirac statistics and of non-uniform band structure is absent and hence θ_n is taken as

zero. Hence, θ_n^{ref} is also zero. In addition, $n_{ref} = N_c^{ref} \exp(\eta_c^{ref})$ since the reference material is assumed to be non-degenerate. This gives,

$$\begin{aligned}\theta_n(x) &= k_b T \ln\left(\frac{N_c(x) F_{\frac{1}{2}}(\eta_c)}{N_c^{ref} \exp(\eta_c^{ref})}\right) - k_b T(\eta_c - \eta_c^{ref}) \\ &= k_b T \ln\left(\frac{N_c(x)}{N_c^{ref}}\right) + k_b T \ln\left(\frac{F_{\frac{1}{2}}(\eta_c)}{\exp(\eta_c^{ref})}\right) - k_b T(\ln(\exp(\eta_c)) - \eta_c^{ref}) \\ &= k_b T \ln\left(\frac{N_c(x)}{N_c^{ref}}\right) + k_b T \ln\left(\frac{F_{\frac{1}{2}}(\eta_c)}{\exp(\eta_c)}\right)\end{aligned} \tag{2.39}$$

The electron affinity $\chi(x)$ in Eq.(2.36) can be combined with $\theta_n(x)$ and a new quantity $V_n(x)$ is defined as,

$$\nabla V_n(x) = \nabla(\chi(x) + \theta_n(x)) \tag{2.40}$$

Integrating Eq.(2.40) on both sides gives,

$$V_n(x) - V_n^{ref} = \chi(x) - \chi^{ref} + \theta_n(x) - \theta_n^{ref}(x) \tag{2.41}$$

Since the reference material is assumed to non-degenerate, V_n^{ref} and θ_n^{ref} are both set to zero. This gives,

$$\begin{aligned}V_n(x) &= \frac{\chi(x) - \chi^{ref}}{q} + \frac{\theta_n(x)}{q} \\ &= \frac{k_b T}{q} \ln\left(\frac{N_c(x)}{N_c^{ref}}\right) + \frac{k_b T}{q} \ln\left(\frac{F_{\frac{1}{2}}(\eta_c)}{\exp(\eta_c)}\right) + \frac{\chi(x) - \chi^{ref}}{q}\end{aligned} \tag{2.42}$$

Similar analysis done for the hole quasi Fermi energy level leads to the parameter V_p given by,

$$V_p(x) = \frac{k_b T}{q} \ln\left(\frac{N_v(x)}{N_v^{ref}}\right) + \frac{k_b T}{q} \ln\left(\frac{F_{\frac{1}{2}}(\eta_v)}{\exp(\eta_v)}\right) - \frac{\chi(x) - \chi^{ref}}{q} - \frac{E_g(x) - E_g^{ref}}{q} \tag{2.43}$$

These two quantities, V_p and V_n, are referred to as band parameters in the literature. It can be observed from Eq.(2.42) and Eq.(2.43) that $V_n(x)$ and $V_p(x)$ depend only on material parameters if the material is non-degenerate, as Boltzmann approximation

can be used to eliminate the second term of Eq.(2.42) and Eq.(2.43). When Fermi-Dirac statistics are assumed, $V_n(x)$ and $V_p(x)$ both depend on electron and hole concentrations respectively. It must be emphasized that although the above equations are derived based on the assumption that the variation in the material parameters is slow, the formulation can also be used for abrupt heterostructures and the deviations of the estimated potential from the exact results are minimal [20].

2.2.3 Equations for Electron and Hole Concentrations

Once the band parameters $V_n(x)$ and $V_p(x)$ are defined, the electron and hole concentrations are calculated from by rewriting, for example, the result given in Eq.(2.39) for the electrons as,

$$\theta_n(x) = kT \ln\left(\frac{n(x)}{N_c^{ref}}\right) - k_b T \eta_c \tag{2.44}$$

Rearranging the terms leads to:

$$\begin{aligned} n(x) &= N_c^{ref} \exp\left(\frac{\theta_n(x) + E_{fn} - E_c}{k_b T}\right) \\ &= n_i^{ref} \exp\left(\ln\left(\frac{N_c^{ref}}{n_i^{ref}}\right)\right) \exp\left(\frac{\theta_n(x) + E_{fn} - E_c}{k_b T}\right) \\ &= n_i^{ref} \exp\left(\frac{\theta_n(x) + E_{fn} - E_c + k_b T \ln\left(\frac{N_c^{ref}}{n_i^{ref}}\right)}{k_b T}\right) \end{aligned} \tag{2.45}$$

Now, Eq.(2.41) and Eq.(2.24) are used to change the band parameter variable from θ_n to V_n to get:

$$\begin{aligned} n(x) &= n_i^{ref} \exp\left(\frac{qV_n(x) - \chi(x) + \chi^{ref} + E_{fn} - E_o + qV + \chi(x) + k_bT\ln\left(\frac{N_c^{ref}}{n_i^{ref}}\right)}{k_bT}\right) \\ &= n_i^{ref} \exp\left(\frac{qV_n(x) + \chi^{ref} + E_F - q\phi_n - E_o + qV + k_bT\ln\left(\frac{N_c^{ref}}{n_i^{ref}}\right)}{k_bT}\right) \\ &= n_i^{ref} \exp\left(\frac{qV + qV_n(x) - q\phi_n}{k_bT}\right) \exp\left(\frac{\chi^{ref} + E_F + kT\ln\left(\frac{N_c^{ref}}{n_i^{ref}}\right) - E_o}{k_bT}\right) \end{aligned} \tag{2.46}$$

Since E_o is a reference energy, it can be chosen such that the second exponential of the result given in Eq.(2.46) is one. This results in

$$E_o = \chi^{ref} + E_F + k_bT\ln\left(\frac{N_c^{ref}}{n_i^{ref}}\right) \tag{2.47}$$

The physical meaning of the choice of E_o becomes apparent when one considers the case of the reference material being intrinsic. Then,

$$E_c^{ref} - E_F = k_bT\ln\left(\frac{N_c^{ref}}{n_i^{ref}}\right) \tag{2.48}$$

Substituting Eq.(2.48) into Eq.(2.47) gives

$$\begin{aligned} E_o &= \chi^{ref} + E_F + E_c^{ref} - E_F \\ &= \chi^{ref} + E_o - qV - \chi^{ref} \end{aligned} \tag{2.49}$$

Hence, the choice of reference potential energy E_o is such that it makes $V = 0$ in the reference material if it is intrinsic. Thus the electron concentration *n(x)* is given by

$$n(x) = n_i^{ref} \exp\left(\frac{qV(x) + qV_n(x) - q\phi_n(x)}{k_bT}\right) \tag{2.50}$$

Similar analysis for the holes gives the hole concentration *p(x)* as

$$p(x) = n_i^{ref} \exp\left(\frac{-qV(x) + qV_p(x) + q\phi_p(x)}{k_bT}\right) \tag{2.51}$$

It is important to emphasize the necessary assumptions and conditions under which the equations derived above are valid.

1. The equation (2.37) for $\nabla\theta_n$ is valid in general for any device with position dependent density of states.

2. The equation (2.42), which forms the basis for defining band parameter $V_n(x)$ is valid when the reference material is non-degenerate.

2.2.4 Boundary Conditions for Poisson Equation

A) Schottky Boundary Condition:

At the interface, the intrinsic level is:

$$\begin{aligned} E_I(x) &= E_o - qV(x) - \chi(x) - \frac{E_g(x)}{2} + \frac{k_bT}{2}\ln\left(\frac{N_v(x)}{N_c(x)}\right) \\ &= E_F + (\chi^{ref} - \chi(x)) - \frac{E_g(x)}{2} + k_bT\ln\left(\frac{N_c^{ref}}{n_i^{ref}}\right) \\ &\quad + \frac{k_bT}{2}\ln\left(\frac{N_v(x)}{N_c(x)}\right) - qV(x) \end{aligned} \tag{2.52}$$

$E_F - E_I(x)$ at the interface can also be written as:

$$\begin{aligned} E_F - E_I(x) &= E_F - E_c(x) + E_c - E_I(x) \\ &= -\phi_{bo} + \frac{E_g(x)}{2} - \frac{k_bT}{2}\ln\left(\frac{N_v(x)}{N_c(x)}\right) \end{aligned} \tag{2.53}$$

Substituting Eq.(2.52) into Eq.(2.53) we have:

$$\begin{aligned} -\frac{E_g(x)}{2} + \frac{k_bT}{2}\ln\left(\frac{N_v(x)}{N_c(x)}\right) + \phi_{bo} &= (\chi^{ref} - \chi(x)) - \frac{E_g(x)}{2} + k_bT\ln\left(\frac{N_c^{ref}}{n_i^{ref}}\right) + \\ &\quad \frac{k_bT}{2}\ln\left(\frac{N_v(x)}{N_c(x)}\right) - qV(x) \end{aligned} \tag{2.54}$$

$$qV(x) = -\phi_{bo} + (\chi^{ref} - \chi(x)) + k_bT\ln\left(\frac{N_c^{ref}}{n_i^{ref}}\right) \tag{2.55}$$

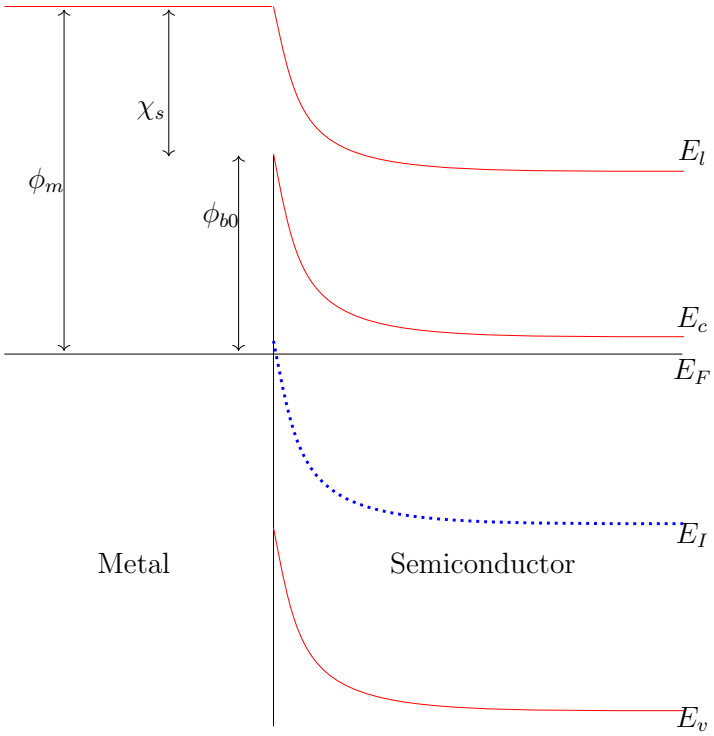

Figure 2.3: Metal-Semiconductor Contact at Equilibrium

i.e.:

$$\begin{aligned} qV(x) &= -\phi_{bo} + (\chi^{ref} - \chi(x)) + k_b T \ln(N_c^{ref}) - \frac{k_b T}{2} \ln(N_c^{ref} N_v^{ref}) + \frac{E_g^{ref}}{2} \\ &= -\phi_{bo} + (\chi^{ref} - \chi(x)) + \frac{E_g^{ref}}{2} + \frac{k_b T}{2} \ln\left(\frac{N_c^{ref}}{N_v^{ref}}\right) \end{aligned} \tag{2.56}$$

B) Ohmic Boundary Condition: To evaluate the ohmic boundary conditions, space-charge neutrality is invoked and the contact is assumed to be at equilibrium. Using Eq.(2.50) and Eq.(2.51) in the charge neutrality equation gives,

$$p - n + (N_d - N_a) = 0$$

$$n_i^{ref} \exp\left(\frac{-qV(x) + qV_p(x)}{k_b T}\right) - n_i^{ref} \exp\left(\frac{qV(x) + qV_n(x)}{k_b T}\right) + (N_d - N_a) = 0$$

It is clear that the above equation is a quadratic in $exp(\frac{qV(x)}{kT})$. Solving the equation

gives the ohmic boundary condition for V(x), of the form:

$$V(x) = \frac{V_p(x) - V_n(x)}{2} + sgn(D)\frac{k_b T}{q} \\ ln\left(\left|\frac{D}{2}\right| e^{\left(\frac{-q(V_p(x)+V_n(x)}{2k_b T}\right)} + \sqrt{\frac{1}{4}D^2 e^{\left(\frac{-q(V_p(x)+V_n(x)}{2kT}\right)} + 1}\right) \tag{2.57}$$

where sgn(D) is the sign of the doping and $D = \frac{N_d - N_a}{n_{ir}}$.

2.3 Linearization and Discretization of the Poisson Equation

To solve any differential equation numerically, first the equation has to be linearized. Linearization converts a given differential equation into a finite set of linear equations which can then be solved using numerical algorithms. To linearize the Poisson equation, the Eq.(2.1) is rewritten as follows.

$$\nabla \cdot (\epsilon \nabla V^{k+1}) = \frac{-q}{\epsilon_o}\left(p^{k+1} - n^{k+1} + DOP\right) \\ = f(V^{k+1}) \tag{2.58}$$

In the above equation, $k+1$ is the iteration number, n^{k+1} and p^{k+1} are the electron and hole concentrations in the $(k+1)^{th}$ iteration and DOP is the net doping density at any point. Using $V^{k+1} = V^k + \delta^{k+1}$, the above equation can be written as:

$$\nabla \cdot (\epsilon \nabla \delta^{k+1}) = f(V^{k+1}) - \nabla \cdot (\epsilon \nabla V^k) \tag{2.59}$$

The first term on the right hand side of Eq.(2.59), $f(V^{k+1})$, can be approximated using Taylor Series expansion around V^k. This gives,

$$f(V^{k+1}) = f(V^k) + \left.\frac{\partial f(V^{k+1})}{\partial V^{k+1}}\right|_{V^k} (\delta^{k+1}) \tag{2.60}$$

Using Eq.(2.50) and Eq.(2.51) under equilibrium conditions, $\left.\frac{\partial f(V^{k+1})}{\partial V^{k+1}}\right|_{V^k}$ can be written as,

$$\left.\frac{\partial f}{\partial V^{k+1}}\right|_{V^k} = \frac{q}{\epsilon_0 K_B T}\left(n^k + p^k\right) \tag{2.61}$$

The final differential equation to solve then becomes

$$\boxed{\nabla \cdot (\epsilon \nabla \delta^{k+1}) - \frac{q\left(n^k + p^k\right)}{\epsilon_0 K_B T} \delta^{k+1} = f(V^k) - \nabla \cdot (\epsilon \nabla V^k)} \tag{2.62}$$

The above equation can now be discretized using any of the standard discretization schemes. For this work, a five point stencil version of the finite volume discretization is used. The resultant coefficients are listed below:

$$\boxed{\begin{aligned} E_{i,j} &= \frac{\epsilon_{i-1,j} + \epsilon_{i,j}}{(X_{i,j+1} - X_{i,j})(X_{i,j+1} - X_{i,j-1})} \\ W_{i,j} &= \frac{\epsilon_{i-1,j-1} + \epsilon_{i,j-1}}{(X_{i,j} - X_{i,j-1})(X_{i,j+1} - X_{i,j-1})} \\ N_{i,j} &= \frac{\epsilon_{i-1,j-1} + \epsilon_{i-1,j}}{(Y_{i,j} - Y_{i-1,j})(Y_{i+1,j} - Y_{i-1,j})} \\ S_{i,j} &= \frac{\epsilon_{i,j-1} + \epsilon_{i,j}}{(Y_{i+1,j} - Y_{i,j})(Y_{i+1,j} - Y_{i-1,j})} \\ C1_{i,j} &= -(E_{i,j} + W_{i,j} + N_{i,j} + S_{i,j}) \\ C^k_{i,j} &= C1_{i,j} - \frac{q(e^k_{i,j} + h^k_{i,j})}{\epsilon_0 V_t} \end{aligned}} \tag{2.63}$$

The forcing function given by the following equation.

$$\boxed{\begin{aligned} F_{i,j} =& \frac{-q(e^k_{i,j} + h^k_{i,j} + DOP_{i,j})}{\epsilon_0 V_t} - (N_{i,j} V^k_{i-1,j} + W_{i,j} V^k_{i,j-1} \\ &+ C1_{i,j} V^k_{i,j} + E_{i,j} V^k_{i,j+1} + S_{i,j} V^k_{i+1,j}) \end{aligned}} \tag{2.64}$$

2.4 Portable, Extensible Toolkit for Scientific Computation (PETSc)

The problem of choosing the most efficient algorithm to numerically solve the differential equation at hand prevails in all branches of computational science. The most natural approach would be to try all possible algorithms and choose the one that is satisfactory to one's needs. However, the vast variety of numerical algorithms already in place, together with their ever expanding variations makes it an extremely time consuming endeavour. Moreover, even after the choice of an algorithm is made,

the style of programming is not guaranteed to result in an efficient algorithm.
The Portable, Extensible Toolkit for Scientific Computation(PETSc)[3] addresses this problem by providing the user with a vast collection of efficiently programmed algorithms, enabling enormous flexibility to the user. Along with a rich collection of linear and nonlinear, PETSc also provides algorithms for many preconditioners, allowing for easy comparison and use of different algorithms. In this chapter, a brief discussion on the Krylov subspace methods and Newton methods which form the core of the PETSc linear and nonlinear solvers respectively is provided. This is followed by a section discussing some important PETSc subroutines employed in almost all PETSc programs.

2.4.1 Preconditioning in PETSc

Consider a linear system defined by the relation $Ax = b$, where A is a non-singular matrix. Let a small perturbation be applied to the system, as is done by any iterative method, such that A matrix changes to $A + \epsilon E$, b vector changes to $b + \epsilon e$ and the solution of the new linear system is $x(\epsilon)$. The perturbed system can be written as

$$(A + \epsilon E)x(\epsilon) = b + \epsilon e \tag{2.65}$$

Expanding $x(\epsilon)$ using the Taylor series gives,

$$\delta(\epsilon) = x(\epsilon) - x + \mathcal{O}(\epsilon^2) \tag{2.66}$$

The above equation, to the first order, can be rewritten as,

$$\begin{aligned}
\delta(\epsilon) &= x(\epsilon) - x \\
(A + \epsilon E)\delta(\epsilon) &= (A + \epsilon E)x(\epsilon) - (A + \epsilon E)x \\
&= (b + \epsilon e) - (A + \epsilon E)x \\
&= \epsilon(e - Ex) \\
\delta(\epsilon) &= \epsilon(A + \epsilon E)^{-1}(e - Ex)
\end{aligned} \tag{2.67}$$

The derivative of $x(\epsilon)$ is indicative of the sensitivity of the solution to a perturbation. This is given as

$$\begin{aligned}
\left.\frac{\partial x(\epsilon)}{\partial \epsilon}\right|_{\epsilon=0} &= \lim_{\epsilon \to 0} \frac{\delta(\epsilon)}{\epsilon} \\
&= A^{-1}(e - Ex)
\end{aligned} \tag{2.68}$$

Using Eq.(2.68), the relative variation in the solution can be written as

$$\begin{aligned}
\frac{||x(\epsilon) - x||}{||x||} &= \frac{||x\prime(0)\epsilon||}{||x||} + \mathcal{O}(\epsilon^2) \\
&\leq \epsilon \frac{||A^{-1}|| \quad ||e - Ex||}{||x||} + \mathcal{O}(\epsilon^2) \\
&\leq \epsilon ||A^{-1}|| \left(\frac{||e|| + ||Ex||}{||x||} \right) + \mathcal{O}(\epsilon^2) \\
&\leq \epsilon ||A^{-1}|| \left(\frac{||e||}{||x||} + ||E|| \right) + \mathcal{O}(\epsilon^2) \\
&\leq \epsilon ||A^{-1}||||A|| \left(\frac{||e||}{||b||} + \frac{||E||}{||A||} \right) + \mathcal{O}(\epsilon^2)
\end{aligned} \tag{2.69}$$

From Eq.(2.69), it is evident that the for a given ϵ, the sensitivity of the solution to perturbation, and thus the rate of convergence, are dependent on the factor $||A^{-1}||||A||$. This parameter is called the condition number, $\kappa(A)$, of the linear system and is relative to a norm. For the case of 2-norm, the condition number is given as

$$\kappa(A) = \frac{\sigma^{max}(A)}{\sigma^{min}(A)} \tag{2.70}$$

where $\sigma^{max}(A)$ and $\sigma^{min}(A)$ are the highest and the lowest eigenvalues of the A matrix. If condition number is small, then the eigenvalues are closely spaced and the system is said to be well conditioned. On the other hand, if condition number is large, the eigenvalues of the system would be spread out and the system is said to be ill-conditioned. The purpose of preconditioning a system is to convert an ill-conditioned system to a well-conditioned system. This is accomplished by defining a preconditioning matrix M and rewriting the linear system as,

$$(M^{-1}A)x = (M^{-1}b) \tag{2.71}$$

Thus the linear system to solve now is given by Eq.(2.71), which has better spectrum than the unconditioned linear system. The problem of solving the linear system $Ax = b$ is now shifted to finding an appropriate preconditioner such that Eq.(2.71) can be solved in fewer iterations.

Although there are a vast variety of preconditioners available, the choice of a good preconditioner for a problem is not always intuitive and depends on the problem at hand. PETSc offers the user with more than a dozen preconditioners, a few of which are listed in the Table (2.2). The user can experiment with different preconditioners and choose the one that works best for the problem at hand.

Method	PCType	Options Database Name
Jacobi	PCJACOBI	jacobi
Incomplete LU	PCILU	ilu
Algebraic Multigrid	PCGAMG	gamg
SOR	PCSOR	sor
No preconditioning	PCNONE	none
Additive Schwarz	PCASM	asm
Cholesky	PCCHOLESKY	cholesky

Table 2.2: Partial List of Preconditioners in PETSc.

For the current work, Incomplete LU (ILU) factorization method, specifically ILU(0), is used as a preconditioner. The ILU method involves performing Gaussian elimination and dropping some elements in predetermined nondiagonal positions. This can be done by defining a static zero pattern set P such that $P \subset \{(i,j)|i \neq j; 1 \leq i,j \leq n\}$. The algorithm of generalized ILU factorization is as follows [21].

1) $For \quad i = 2, ..., n \quad Do:$

2) $For \quad k = 1, ..., i-1 and \quad if \quad (i,k) \notin P \quad Do:$

3) $a_{ik} = a_{ik}/a_{kk}$

4) $For \quad j = k+1, ..., n and \quad if \quad (i,j) \notin P \quad Do:$

5) $a_{ij} = a_{ij} - a_{ik}a_{kj}$

If the zero pattern P is taken to be precisely the zero pattern of A, one arrives at the Incomplete LU factorization technique with no fill-in, denoted by ILU(0).

2.4.2 PETSc Linear Solvers

For a given system of linear equations $Ax = b$, direct methods always give the unique solution of the system. However, as the size of the A matrix increases, performing n^3 calculations to arrive at the solution becomes computationally expensive. In contrast to the direct methods, the aim of the iterative methods is to progress closer towards the true solution of a system of linear equations in each iteration. The governing equation of a basic iterative method is given as:

$$\mathbf{x}^{k+1} = \mathbf{x}^k + \alpha M^{-1}\mathbf{r}^k \tag{2.72}$$

where α is a relaxation parameter, $\mathbf{r}^k$ is the residual after the k^{th} iteration. If the matrix M is the identity matrix I, one arrives at the Successive Over Relaxation(SOR) method. Without loss of generality, if $\alpha = 1$ and $M = I$, then the first few iterations of the basic iterative methods are given by:

$$\begin{aligned}
\mathbf{x}^0 &= \mathbf{0} \\
\mathbf{x}^1 &= \mathbf{x}^0 + \mathbf{r}^0 = \mathbf{r}^0 \\
\mathbf{x}^2 &= \mathbf{x}^1 + \mathbf{r}^1 = 2\mathbf{r}^0 - A\mathbf{r}^0 \\
\mathbf{x}^3 &= (3\mathbf{r}^0 - 3A\mathbf{r}^0 + A^2\mathbf{r}^0
\end{aligned} \tag{2.73}$$

It is evident from the first three iterations that the solution vector $\mathbf{x}^k$ after k iterations is such that $\mathbf{x}^k \epsilon$ span$\{\mathbf{r}^0, A\mathbf{r}^0, A^2\mathbf{r}^0, ..., A^{k-1}\mathbf{r}^0\}$. This space spanned by the vectors $\{\mathbf{r}^0, A\mathbf{r}^0, A^2\mathbf{r}^0, ..., A^{k-1}\mathbf{r}^0\}$ is called the Krylov subspace, denoted by $K^k(\mathbf{r}^0, A)$. In each iteration, the linear solvers based on Krylov subspace find the optimal solution $\mathbf{x}^k$ in this space where the optimality is such that A-norm, $||\mathbf{x} - \mathbf{x}^k||_A$, is minimal for a certain α^k.

2.4.3 PETSc Nonlinear Solvers

Along with the linear solvers, PETSc also includes many algorithms to solve a system of nonlinear equations of the form

$$\mathbf{F}(\mathbf{x}) = 0 \tag{2.74}$$

where $\mathbf{F} : R^n \rightarrow R^n$. For the Poisson equation without the defect correction formulation, the variable $\mathbf{x}$ in Eq.(2.74) is the potential in the system and Eq.(2.74) can be written as

$$\mathbf{F}(V^{k+1}) = \nabla \cdot (\epsilon \nabla V^{k+1}) - f(V^{k+1}) \tag{2.75}$$

where $f(V^{k+1}) = \frac{-q}{\epsilon_o}\left(p^{k+1} - n^{k+1} + DOP\right)$. After discretization, the above equation reduces to a set of linear equations:

$$F_{ij}^k = N_{ij}V_{i-1,j}^k + W_{ij}V_{i,j-1}^k + C_{ij}V_{i,j}^k + E_{ij}V_{i,j+1}^k + S_{ij}V_{i+1,j}^k - f(V_{i,j}^k) \tag{2.76}$$

The nonlinear solvers in PETSc, for solving Eq.(2.74), are based on the Newton method. The general form of a n-dimensional Newton's method is given as,

$$\mathbf{x}^{k+1} = \mathbf{x}^k - \mathbf{J}(\mathbf{x}^k)^{-1}\mathbf{F}(\mathbf{x}^k), k = 0, 1, 2... \tag{2.77}$$

In Eq.(2.77), $\mathbf{J}(\mathbf{x}^k)$ is the Jacobian of the system evaluated at $\mathbf{x}^k$. Using Eq.(2.76), the Jacobian of the Poisson system can be written as,

$$J(V^k) = \begin{bmatrix} \frac{\partial F^k_{1,1}}{\partial V^k_{1,1}} & \frac{\partial F^k_{1,1}}{\partial V^k_{1,2}} & \cdots & \frac{\partial F^k_{1,1}}{\partial V^k_{N,N-1}} & \frac{\partial F^k_{1,1}}{\partial V^k_{N,N}} \\ \frac{\partial F^k_{1,2}}{\partial V^k_{1,1}} & \frac{\partial F^k_{1,2}}{\partial V^k_{1,2}} & \cdots & \frac{\partial F^k_{1,2}}{\partial V^k_{N,N-1}} & \frac{\partial F^k_{1,2}}{\partial V^k_{N,N}} \\ \vdots & \vdots & \cdots & \vdots & \vdots \\ \frac{\partial F^k_{i,j}}{\partial V^k_{1,1}} & \frac{\partial F^k_{i,j}}{\partial V^k_{1,2}} & \cdots & \frac{\partial F^k_{i,j}}{\partial V^k_{N,N-1}} & \frac{\partial F^k_{i,j}}{\partial V^k_{N,N}} \\ \vdots & \vdots & \cdots & \vdots & \vdots \\ \frac{\partial F^k_{N,N-1}}{\partial V^k_{1,1}} & \frac{\partial F^k_{N,N-1}}{\partial V^k_{1,2}} & \cdots & \frac{\partial F^k_{N,N-1}}{\partial V^k_{N,N-1}} & \frac{\partial F^k_{N,N-1}}{\partial V^k_{N,N}} \\ \frac{\partial F^k_{N,N}}{\partial V^k_{1,1}} & \frac{\partial F^k_{N,N}}{\partial V^k_{1,2}} & \cdots & \frac{\partial F^k_{N,N}}{\partial V^k_{N,N-1}} & \frac{\partial F^k_{N,N}}{\partial V^k_{N,N}} \end{bmatrix} \tag{2.78}$$

To form the Jacobian, one has to assume that the potentials $V^k_{i,j}$ are continuous and thus $\frac{\partial F^k_{i,j}}{\partial V^k_{i,j}}$ exists. It is also evident from Eq.(2.76) that the Jacobian would be a sparse matrix and the nonzero entries are given as,

$$\boxed{\begin{aligned} \frac{\partial F^k_{ij}}{\partial V^k_{i-1,j}} &= N_{i,j} \\ \frac{\partial F^k_{ij}}{\partial V^k_{i,j-1}} &= W_{i,j} \\ \frac{\partial F^k_{ij}}{\partial V^k_{i,j+1}} &= E_{i,j} \\ \frac{\partial F^k_{ij}}{\partial V^k_{i+1,j}} &= S_{i,j} \\ \frac{\partial F^k_{ij}}{\partial V^k_{i,j}} &= C_{i,j} - \frac{\partial f(V_{i,j})}{\partial V_{i,j}} \end{aligned}} \tag{2.79}$$

In practice, the Newton method is implemented in the following two steps:

1) Solve $\mathbf{J}(\mathbf{x}^k)\Delta\mathbf{x}^k = -\mathbf{F}(\mathbf{x}^k)$

2) Update $\mathbf{x}^{k+1} = \mathbf{x}^k + \Delta\mathbf{x}^k$

The necessary routines to be used and the procedure for implementation of a nonlinear solver is discussed in the next section.

2.4.4 Important PETSc Routines for Linear and Nonlinear Solvers

The program flow of a typical PETSc linear and nonlinear solvers are depicted in Figure (2.4) and Figure (2.5) respectively. The various subroutines that comprise each block are discussed below. All the PETSc routines return an integer error code, denoted by the variable ierr, which is set to be nonzero if an error is detected. CHKERRQ(ierr) is a PETSc macro that checks the value of ierr and calls the PETSc error handler if ierr is nonzero. CHKERRQ(ierr) should be used after every PETSc routine to ensure that successful execution of a routine.

Adding PETSc Header Files

To successfully compile any PETSc program, one has to first include all the necessary header files required by the program. For the case of linear solvers in PETSc, the header file to be included is petscksp.h. This defines the interface functions for the Krylov subspace accelerators. If the programming language is Fortran, then appropriate module should also be added at the beginning of the program. These are achieved using the following statements.

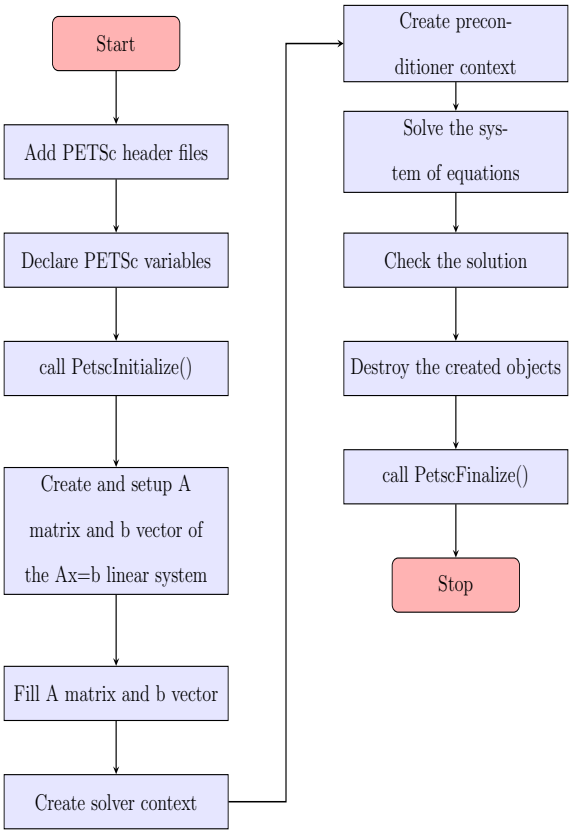

Figure 2.4: Flow of PETSc Program Based on Krylov Solvers

```
#include ⟨petsc/finclude/petscksp.h⟩
use petscksp
```

PetscInitialize

All the PETSc programs must call PetscInitialize() at the beginning of the program using the following statement.

```
PetscInitialize(PETSC_NULL_CHARACTER,ierr)
```

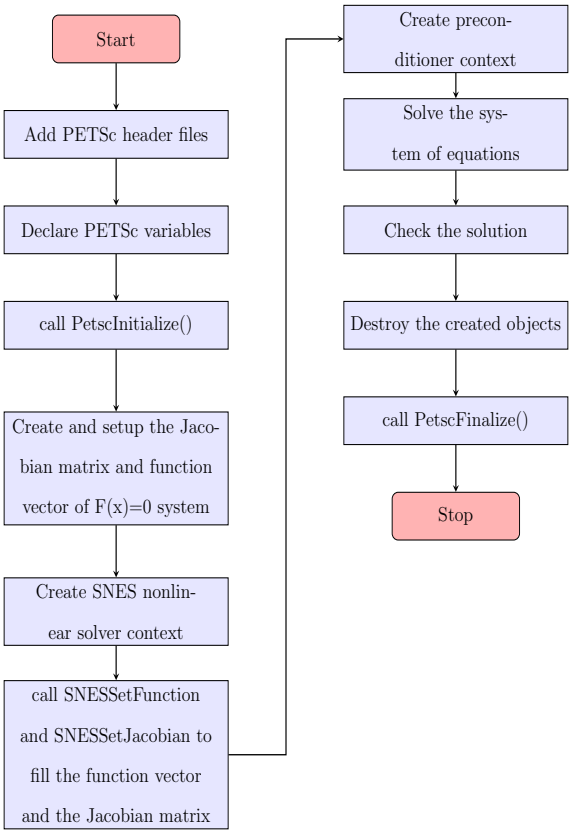

Figure 2.5: Flow of PETSc Program Based on Newton Method

Creating and Setting up Matrix and Vector Objects

After PETSc is initialized, the next step is to create the matrix A and the vector b of the equation $Ax = b$. To create a matrix, one has to call the following routines.

```
MatCreate(PETSC_COMM_WORLD,A,ierr)
```

After the matrix is created, the next step is to set the size and type of the matrix by calling the following routines.

```
MatSetSizes(A,PETSC_DECIDE,PETSC_DECIDE,NUM,NUM,ierr)
MatSetType(A,MATSEQAIJ,ierr)
```

By default, the MatCreate routine creates a sequential sparse matrix based on compressed sparse row format. PETSc has more than 50 different types of matrices and the MatSetType() routine allows the user to build a matrix of a certain type. The next step is to preallocate the memory needed for the sparse matrix. In general, allocating memory dynamically is extremely expensive and in order to obtain a good performance, one has to always preallocate memory for the sparse matrix. This is achieved using the following command.

```
MatSeqAIJSetPreallocation(A,PETSC_DEFAULT_INTEGER,nnz,ierr)
```

The variable `nnz` in the above command is an integer array which indicates the number of nonzeros in each row of the A matrix. If the number of nonzeros is roughly the same in each row, one has to use a scalar `nz` to specify the number of nonzeros. If `nnz` is specified then `nz` is ignored. This is followed by setting up the matrix A using the MatSetUp() routine. This sets up the internal matrix data structures for later use.

```
MatSetUp(A,ierr)
```

Once the matrix is setup, the next step is to add values into the matrix. PETSc allows the user to set single entry or a block of values to the matrix using the following routines.

```
MatSetValues0(A,m,idxm[],n,idxn[],v[],InsertMode,ierr)
```

The variable `m` is the number of rows to be filled, `idxm[]` is the index of rows to be filled, `n` is the number of columns to be filled, `idxn[]` is the index of the columns to be filled and `v[]` is a logical 2D array of size `mXn` containing the values to be entered in

the matrix. The `InsertMode` argument allows the user to either overwrite the existing values in the matrix or to put values into a location that previously has no value. To set single entry instead of an array of values, one can call the MatSetValue() routine as follows.

```
MatSetValue(A,m,n,v,InsertMode,ierr)
```

The final step before the matrix is ready to use is to assemble the matrix. This is accomplished using the routines mentioned below.

```
MatAssemblyBegin(A,MAT_FINAL_ASSEMBLY,ierr)
MatAssemblyEnd(A,MAT_FINAL_ASSEMBLY,ierr)
```

Similar routines are available to create, setup and assemble a vector object. These are listed below.

```
VecCreate(PETSC_COMM_SELF,x,ierr)
VecSetType(x,VECSEQ,ierr)
VecSetValues(x,n,ix[],y[],InsertMode,ierr)
VecAssemblyBegin(x,ierr)
VecAssemblyEnd(x,ierr)
```

If the nonlinear solvers are to be used, then the user has to define two subroutines, one to calculate the Jacobian matrix and another to calculate the forcing function, which are defined as follows.

```
FormJacobian(snes,x,jac,prec,ctx,ierr)
FormFunction(snes,x,f,ctx,ierr)
```

In the first routine, the variables snes, x and ctx are the input variables and jac, prec, ierr are the output variables. x is the solution vector at the k^{th} iteration, jac is jacobian matrix, prec is the preconditioning matrix and ctx is an optional user defined context. In the second routine snes, x and ctx are the input variables and f is the output variable that stores the value of the function. These subroutines are then given as inputs to two PETSc routines which enable the formation of jacobian matrix and evaluate function **F**. These two routines are defined as follows.

```
SNESSetFunction(snes,f,FormFunction(),ctx,ierr)
SNESSetJacobian(snes,Amat,Pmat,FormJacobian(),ctx,ierr)
```

Creating Solver and Preconditioner Contexts

Once the A matrix and the b vector are formed, the next step is to create the solver context and set the type of Krylov method or the Newton method using the following routines.

```
KSPCreate(PETSC_COMM_WORLD,ksp,ierr)
KSPSetType(ksp,KSPBCGS,ierr)
```

```
SNESCreate(PETSC_COMM_WORLD,snes,ierr)
SNESSetType(snes,SNESNEWTONLS,ierr)
```

The second argument of the routine KSPSetType and SNESSetType is the type of the solver, where the user can choose from among more than thirty Krylov methods and dozen Newton methods, partial list of which is shown in Table 2.3 and Table 2.4 respectively. PETSc also allows user to set the tolerance in the form of a real number or in terms of maximum number of iterations to use with the following routine.

Method	KSPType	Options Database Name
Richardson	KSPRICHARDSON	richardson
BiCGSTAB	KSPBCGS	bcgs
Generalized Minimal Residual	KSPGMRES	gmres
Conjugate Gradient	KSPCG	cg
Chebyshev	KSPCHEBYSHEV	chebyshev

Table 2.3: Partial List of Available Krylov Methods.

Method	SNESType	Options Database Name
Line Search Newton	SNESNEWTONLS	newtonls
Nonlinear GMRES	SNESNGMRES	ngmres
Nonlinear Gauss-Seidel	SNESNGS	ngs
Anderson Mixing	SNESANDERSON	anderson

Table 2.4: Partial List of Available Newton Methods.

```
KSPSetTolerances(ksp,rtol,abstol,dtol,maxits,ierr)
```

After setting the tolerances, the user needs to specify the matrix associated with the linear system to SLEPc, by calling the following routine.

```
KSPSetOperators(ksp,Amat,Pmat,ierr)
```

The argument Pmat in the above routine is a matrix that is used in constructing the preconditioner. This is usually same as Amat. To experiment with other preconditioners available in PETSc, one has to first get the preconditioner context followed by setting the preconditioner type. This is achieved using the following routines.

```
KSPGetPC(ksp,pc,ierr)
PCSetType(pc,PCJACOBI,ierr)
```

The second argument of the PCSetType() routine allows the user to choose from among more than dozen preconditioners offered by PETSc. Once the solver and preconditioner contexts are set, the next step is to solve the system of equations. This is achieved in a single command given below depending the solver context .

```
KSPSolve(ksp,b,x,ierr)
```

```
SNESSolve(snes,b,x,ierr)
```

The argument b in the above routine is the right hand side vector in the equation $Ax = b$ and x is the solution vector.

Checking the Solution and Cleanup

To check if the solver converged, one has to call the following routine.

```
KSPGetConvergedReason(ksp,reason,ierr)
```

In the above routine, `reason` is of integer type varying from -11 to 8, where negative values indicate that the solver has not converged. Once the solver is converged, one must destroy all the objects that are created such as the matrices, vector, the solver and the preconditioner objects, followed by a call to PetscFinalize() as mentioned below.

```
VecDestroy(x,ierr)
MatDestroy(A,ierr)
KSPDestroy(ksp,ierr)
PCDestroy(ksp,ierr)
PetscFinalize(ierr)
```

2.5 Comparison of Results Between Silvaco-ATLAS and Petsc Poisson Solver

To validate the Poisson solver built using PETSc, the results of the PETSc solver and Silvaco-ATLAS are compared for two device structures shown in Figure (2.6).

In Silvaco-ATLAS [22], the **interface** statement is used to add polarization charges at the $Al_{0.2}Ga_{0.8}N$-GaN interface and at the region between the gates on the surface. For the **interface** statement at the $Al_{0.2}Ga_{0.8}N$-GaN interface, one has to add the **S.S** option to specify the application of interface models pertinent to the semiconductor-semiconductor interface. And for the **interface** statement at the top surface, one has to add the **S.X** option to specify the application of interface models pertinent to the interfaces with the outside domain. The results of the comparison are shown in the figures below.

As seen in conduction band profile of Figure (2.7), 2DEG is formed at the $Al_{0.2}Ga_{0.8}N$-GaN interface. The Silvaco results match perfectly with PETSc Poisson solver. The

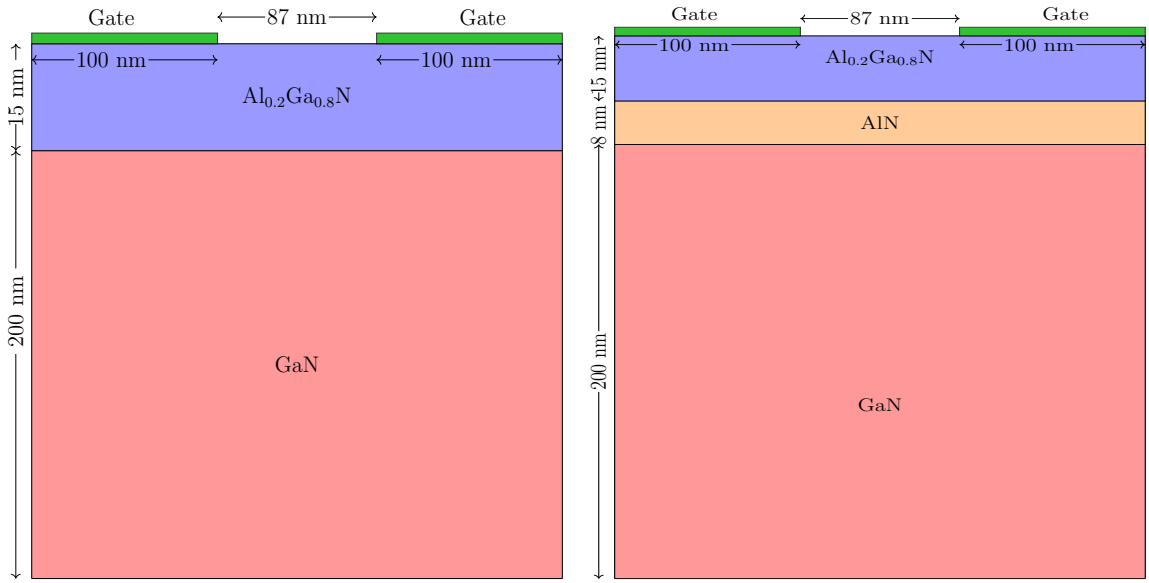

Figure 2.6: Device Structures Used to Establish the Validity of the Poisson Solver.

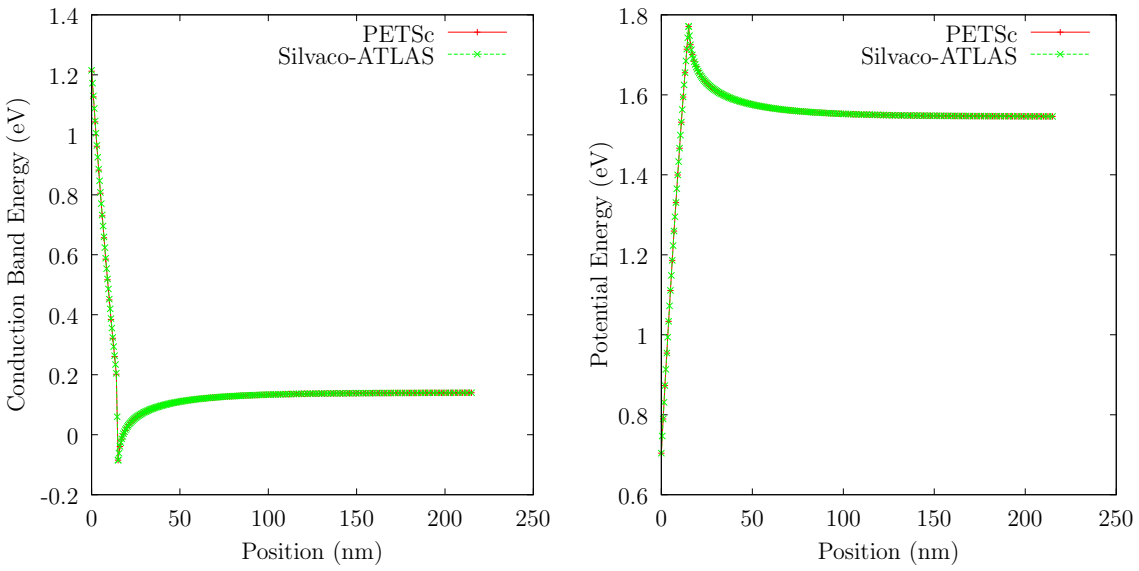

Figure 2.7: Comparison of Conduction Band Profile and Potential Profile along the Y-cutline Between the Gates for the Device Without AlN Layer.

results for the device with an AlN layer is shown in Figure (2.8). The 2DEG is form at the AlN-GaN interface and the results here match perfectly with the Silvaco-ATLAS. Lastly, the convergence profile of the Poisson solver is shown in Figure (2.9) .

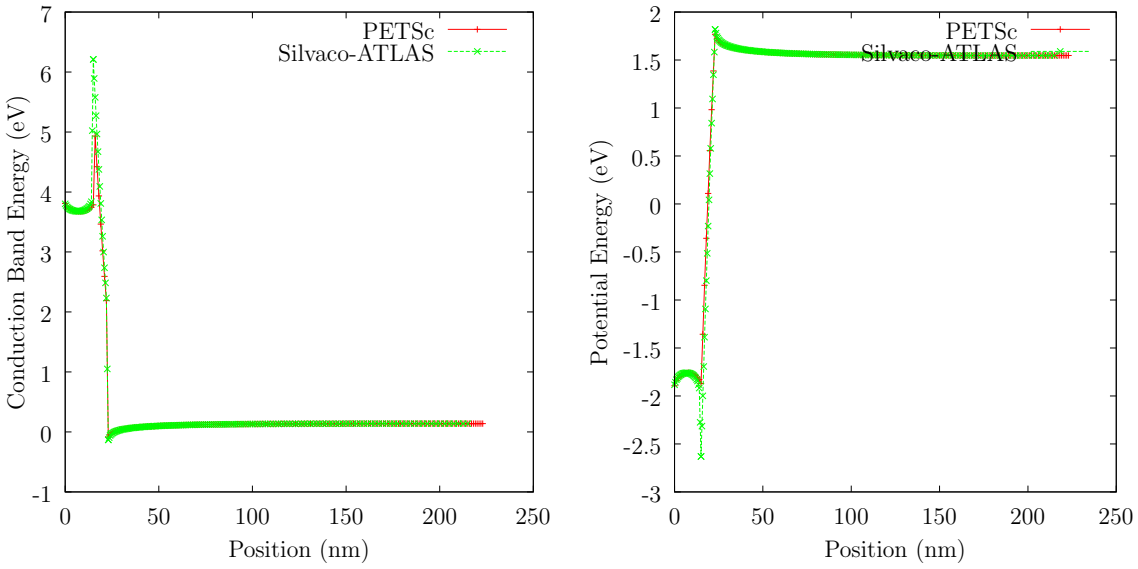

Figure 2.8: Comparison of Conduction Band Profile and Potential Profile along the Y-cutline Between the Gates for the Device with AlN Layer.

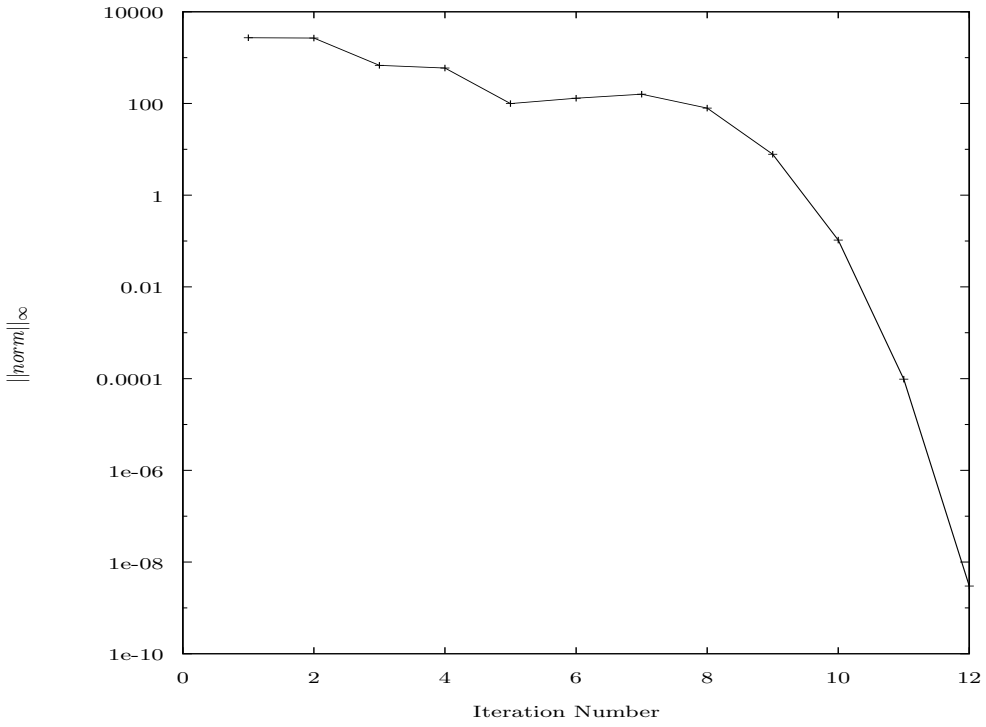

Figure 2.9: Convergence of the PETSc based Poisson Solver.

Chapter 3

SLEPC AND THE SCHRÖDINGER SOLVER

3.1 Finite Volume Discretization of a Linear PDE

The aim of any discretization scheme is to convert given partial differential equation into a system of linear equations which can then be solved using standard numerical algorithms. Various discretization schemes are available, the commonly used ones being the Finite Difference Method(FDM), the Finite Element Method(FEM) and the Finite Volume Method(FVM). This section discusses the FVM using the method of octants, as described in [23]. The general form of a linear differential operator of order n is

$$L^n(\vec{r}) = a_n(x)\hat{D}^n(\vec{r}) + \ldots + a_1(x)\hat{D}^1(\vec{r}) + a_0(x)\hat{D}^0(\vec{r}) \tag{3.1}$$

where $a_i(x), i = 0, \ldots, n$ represents an arbitrary differentiable function and $\hat{D}^i(\vec{r}), i = 0, \ldots, n$ represents the i^{th} order differential operator. For the case of the Schrödinger equation and the linearized Poisson equation, i takes the values of 0 and 2. Hence this work focuses on the discretizing the operators $\hat{D}^2(\vec{r})$ and $\hat{D}^0(\vec{r})$.

The FV discretization scheme starts with generation of a mesh over the entire simulation domain. Then a control volume of size Ω, shown by dotted lines in Figure (3.1), is constructed by connecting the midpoints of the neighbouring boxes. The control volume is further divided into N octants, where $N = 2^d$, d being the dimensionality of the simulation domain.

A general second order differential operator in 2D is of the form

$$\hat{D}^{(2)}\phi(\vec{r}) = \sum_{i,j\epsilon\{x,y\}} \partial_i \alpha_{ij}(\vec{r})\partial_j \phi(\vec{r}) \tag{3.2}$$

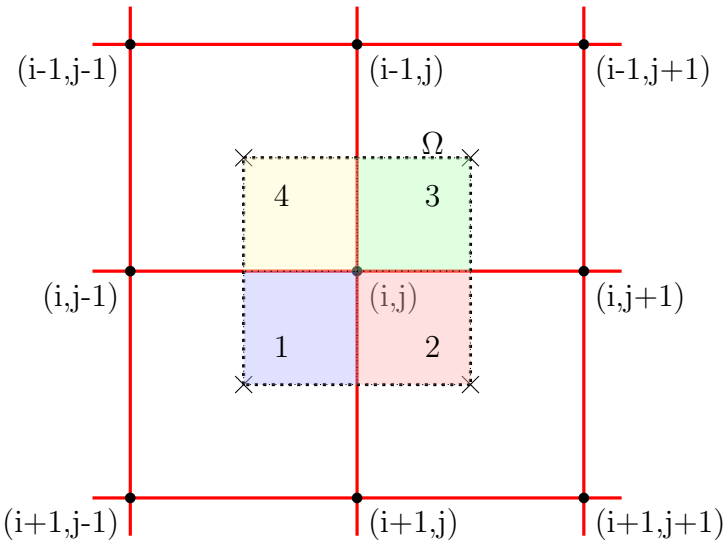

Figure 3.1: Representation of Control Volume and Octants about a Mesh Point

where $\alpha_{ij}(\vec{r})$ is a rank-2 tensor as shown below.

$$\alpha = \begin{bmatrix} \alpha_{xx} & \alpha_{xy} & \alpha_{xz} \\ \alpha_{yx} & \alpha_{yy} & \alpha_{yz} \\ \alpha_{zx} & \alpha_{zy} & \alpha_{zz} \end{bmatrix} \tag{3.3}$$

The differential form is converted to the integral form at each grid point by integrating over the control volume Ω. This gives,

$$I_2 = \int_\Omega \sum_{i,j\epsilon\{x,y\}} \partial_i \alpha_{ij}(\vec{r}) \partial_j \phi(\vec{r}) dV \tag{3.4}$$

$$= \int_\Omega \sum_{i\epsilon\{x,y\}} \partial_i \left(\sum_{j\epsilon\{x,y\}} \alpha_{ij}(\vec{r}) \partial_j \phi(\vec{r}) \right) dV \tag{3.5}$$

It is easy to recognize that the integrand is of the form $\nabla.\mathbf{F}$ where the flux $\mathbf{F} = \left(\sum_{i,j\epsilon\{x,y\}} \alpha_{ij}(\vec{r}) \partial_j \phi(\vec{r}) \hat{\mathbf{e}}_i^0 \right)$. Thus, Eq.(3.4) can be written as

$$I_2 = \int_\Omega (\nabla.\mathbf{F}) dV \tag{3.6}$$

Applying the divergence theorem, I_2 can be converted from a volume integral to a

surface integral as

$$I_2 = \int_S \mathbf{F}.d\mathbf{s} \tag{3.7}$$

$$= \int_S \sum_{i,j\epsilon\{x,y\}} \alpha_{ij}(\vec{r})\partial_j\phi(\vec{r})\hat{\mathbf{e}}_i^0.d\mathbf{s} \tag{3.8}$$

This surface integral can be written as the summation of N surface integrals where $N = 2^d$ is the number of octants.

$$I_2 = \sum_{n=1}^{2^d} \int_{S_n} \sum_{i,j\epsilon\{x,y\}} \alpha_{ij}^{(n)}(\vec{r})[\partial_j\phi(\vec{r})]^n\hat{\mathbf{e}}_i^0.d\mathbf{s}_n \tag{3.9}$$

Furthermore, the surface integral over each octant can be written as a summation of surface integrals over each surface of an octant. The number of surfaces for an octant is $p = d$, where d is the dimensionality of the system under consideration. Thus, I_2 can be written as,

$$I_2 = \sum_{n=1}^{2^d}\sum_{p=1}^{d} \int_{S_p^n} \sum_{i,j\epsilon\{x,y\}} \alpha_{ij}^{(n)}(\vec{r})[\partial_j\phi(\vec{r})]_p^n(\hat{\mathbf{e}}_i^0.\hat{\mathbf{e}}_p^n)ds_p^n \tag{3.10}$$

where $\hat{\mathbf{e}}_p^n$ is the unit normal vector of the the p^{th} surface of the n^{th} octant. Using basis orthonormality,

$$\hat{\mathbf{e}}_i^0.\hat{\mathbf{e}}_p^n = \delta_{ip}\hat{\mathbf{e}}_p^n(p) \tag{3.11}$$

where $\hat{\mathbf{e}}_p^n(p)$ is the p^{th} element of the $\hat{\mathbf{e}}_p^n$ column vector, and substituting this back gives

$$I_2 = \sum_{n=1}^{2^d}\sum_{p=1}^{d} \int_{S_p^n} \sum_{i,j\epsilon\{x,y\}} \alpha_{ij}^{(n)}(\vec{r})[\partial_j\phi(\vec{r})]_p^n\delta_{ip}\hat{\mathbf{e}}_p^n(p)ds_p^n \tag{3.12}$$

The Kronecker delta allows one to eliminate the summation on p which leads to

$$\begin{aligned} I_2 &= \sum_{n=1}^{2^d} \sum_{i,j\epsilon\{x,y\}} \int_{S_i^n} \alpha_{ij}^{(n)}(\vec{r})[\partial_j\phi(\vec{r})]_i^n e_i^n(i)ds_i^n \\ &= \sum_{n=1}^{2^d} \sum_{i,j\epsilon\{x,y\}} \alpha_{ij}^{(n)}(\vec{r})[\partial_j\phi(\vec{r})]_i^n e_i^n(i)S_i^n \end{aligned} \tag{3.13}$$

where $[\partial_j \phi(\vec{r})]_i^n$ is the discrete approximation of differential change in ϕ in the j^{th} direction on the i^{th} surface of the n^{th} octant of the control volume around the point at $\vec{r}$. The term $[\partial_j \phi(\vec{r})]_i^n$ is discretized as

$$[\partial_j \phi(\vec{r})]_i^n = \begin{cases} \frac{\phi(\Delta_j^n \hat{e}_j^n) - \phi(\vec{r})}{\Delta_j^n} e_j^n(j), & i = j \\ \frac{1}{2}\left(\frac{\phi(\Delta_j^n \hat{e}_j^n) - \phi(\vec{r})}{\Delta_j^n} - \frac{\phi(\Delta_j^n \hat{e}_j^n + \Delta_i^n \hat{e}_i^n) - \phi(\Delta_i^n \hat{e}_i^n)}{\Delta_j^n}\right) e_j^n(j), & i \neq j \end{cases} \tag{3.14}$$

where Δ_i^n is the mesh spacing along i^{th} direction. If the parameter α. which is $\frac{1}{m^*}$ for the case of Schrödinger equation, is assumed to be isotropic, then $i = j$. Thus,

$$\begin{aligned} I_2 &= \sum_{n=1}^{2^d} \sum_{i,j \epsilon \{x,y\}} \alpha_{ij}^{(n)}(\vec{r}) [\partial_i \phi(\vec{r})]_i^n S_i^n \\ &= \sum_{n=1}^{2^d} \sum_{i,j \epsilon \{x,y\}} \alpha_{ij}^{(n)}(\vec{r}) \left(\frac{\phi(\Delta_i^n \hat{e}_i^n) - \phi(\vec{r})}{\Delta_i^n} \right) S_i^n \end{aligned} \tag{3.15}$$

From (3.15) it is clear that one can associate five coefficients to any node as shown in Figure (3.2).

$$\begin{aligned} I_2 = \quad & \alpha^{(1)}(\vec{r}) \left(\frac{(\phi(w) - \phi(c))}{\Delta_1^1} S_1^1 + \frac{(\phi(s) - \phi(c))}{\Delta_2^1} S_2^1 \right) \\ + \quad & \alpha^{(2)}(\vec{r}) \left(\frac{(\phi(e) - \phi(c))}{\Delta_1^2} S_1^2 + \frac{(\phi(s) - \phi(c))}{\Delta_2^2} S_2^2 \right) \\ + \quad & \alpha^{(3)}(\vec{r}) \left(\frac{(\phi(e) - \phi(c))}{\Delta_1^3} S_1^3 + \frac{(\phi(n) - \phi(c))}{\Delta_2^3} S_2^3 \right) \\ + \quad & \alpha^{(4)}(\vec{r}) \left(\frac{(\phi(w) - \phi(c))}{\Delta_1^4} S_1^4 + \frac{(\phi(n) - \phi(c))}{\Delta_2^4} S_2^4 \right) \end{aligned} \tag{3.16}$$

The discretization of zeroth order operator $\hat{D}^0(\vec{r})$ is trivial. The integral form of the operator is written as,

$$\begin{aligned} I_0 &= \int_\Omega \phi(\vec{r}) \gamma(\vec{r}) \\ &= \phi(\vec{r}) \sum_{n=1}^{4} \gamma^n vol^n \end{aligned} \tag{3.17}$$

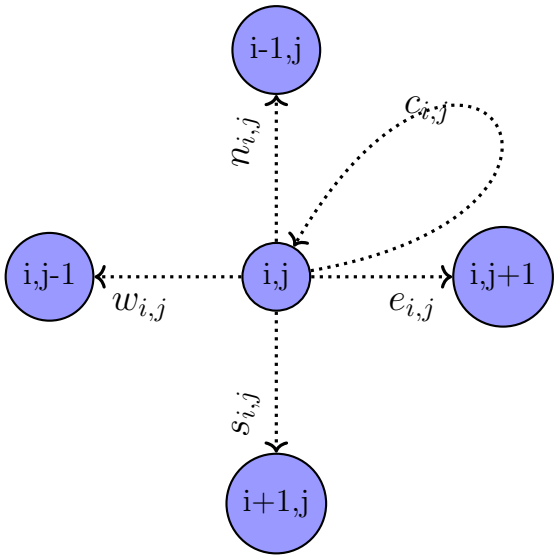

Figure 3.2: The Five Point Stencil of the Finite Volume Method

where vol^n is the volume of the n^{th} octant. Using equations (3.16) and (3.17), the north, east, west and south coefficients are given as :

$$\begin{aligned}
e_{i,j} &= \frac{\alpha^{(2)}(Y_{i+1,j} - Y_{i,j})}{2(X_{i,j+1} - X_{i,j})} + \frac{\alpha^{(3)}(Y_{i,j} - Y_{i-1,j})}{2(X_{i,j+1} - X_{i,j})} \\
w_{i,j} &= \frac{\alpha^{(1)}(Y_{i+1,j} - Y_{i,j})}{2(X_{i,j} - X_{i,j-1})} + \frac{\alpha^{(4)}(Y_{i,j} - Y_{i-1,j})}{2(X_{i,j} - X_{i,j-1})} \\
n_{i,j} &= \frac{\alpha^{(4)}(X_{i,j} - X_{i,j-1})}{2(Y_{i,j} - Y_{i-1,j})} + \frac{\alpha^{(3)}(X_{i,j+1} - X_{i,j})}{2(Y_{i,j} - Y_{i-1,j})} \\
s_{i,j} &= \frac{\alpha^{(1)}(X_{i,j} - X_{i,j-1})}{2(Y_{i+1,j} - Y_{i,j})} + \frac{\alpha^{(2)}(X_{i,j+1} - X_{i,j})}{2(Y_{i+1,j} - Y_{i,j})}
\end{aligned} \tag{3.18}$$

The central coefficient is given as :

$$\begin{aligned} c_{i,j} = &- \left(e_{i,j} + w_{i,j} + n_{i,j} + s_{i,j}\right) \\ &+ \gamma^{(1)}(\vec{r}) \frac{(X_{i,j} - X_{i,j-1})}{2} \frac{(Y_{i+1,j} - Y_{i,j})}{2} \\ &+ \gamma^{(2)}(\vec{r}) \frac{(X_{i,j+1} - X_{i,j})}{2} \frac{(Y_{i+1,j} - Y_{i,j})}{2} \\ &+ \gamma^{(3)}(\vec{r}) \frac{(X_{i,j} - X_{i,j-1})}{2} \frac{(Y_{i,j} - Y_{i-1,j})}{2} \\ &+ \gamma^{(4)}(\vec{r}) \frac{(X_{i,j+1} - X_{i,j})}{2} \frac{(Y_{i,j} - Y_{i-1,j})}{2} \end{aligned} \tag{3.19}$$

This discretization scheme leads to an eigenvalue problem of the form

$$H\Psi = ES\Psi \tag{3.20}$$

where $H_{N\times N}$ is the Hamiltonian matrix, $\Psi_{N\times N}$ is the collection of wavefunctions, E is the diagonal matrix of energy eigenvalues and S is the diagonal matrix containing the volume of each control volume. For a non-uniform mesh, the Hamiltonian matrix is not Hermitian. To convert H to a Hermitian matrix, the steps mentioned below are followed:

$$\begin{aligned} H\Psi &= ES\Psi \\ S^{-0.5}H\Psi &= S^{-0.5}ES\Psi \\ &= S^{0.5}E\Psi \\ S^{-0.5}HS^{-0.5}S^{0.5}\Psi &= S^{0.5}ES^{-0.5}S^{0.5}\Psi \\ &= ES^{0.5}\Psi \\ H_{new}\Psi_{new} &= E\Psi_{new} \end{aligned} \tag{3.21}$$

where $H_{new} = S^{-0.5}HS^{-0.5}$ and $\Psi_{new} = S^{0.5}\Psi$. Now, this new system of linear equations is solved with the SLEPc eigenvalue package mentioned in the next section. It can be observed from Eq.(3.21) that although the eigenvalues of the new system remain the same, Ψ_{new}, however, should be pre-multiplied by $S^{-0.5}$ to get the true wavefunction.

3.2 The SLEPc Eigenvalue Solver Package

Similar to the PETSc linear solver package described in Chapter 2, the Scalable Library for Eigenvalue Problem Computations (SLEPc) is a software package for the solution of large sparse eigenproblems [4]. A variety of problem classes such as the linear eigenvalue problem, polynomial eigenvalue problem, nonlinear eigenvalue problem, etc., are supported by SLEPc. SLEPc is built on top of PETSc and is considered as an extension of PETSc, providing the functionality for solution of eigenvalue problems.

There is a significant overlap between the routines used for solving an eigenvalue problem and a linear system. For example, both systems require forming the A matrix and, hence, the routines needed would be exactly the same. Since these routines are already discussed in Chapter 2, this section discusses only the routines which are exclusive to the SLEPc package. The implementation flow of a typical SLEPc program is shown in Figure (3.3).

Similar to setting linear/nonlinear solver contexts in PETSc, one has to set eigenvalue solver context and describe the problem type, the eigenpairs to be computed. This is achieved using the following routines described with the example of a Hermitian eigenvalue problem type and calculating the smallest eigenpairs.

```
EPSCreate(PETSC_COMM_WORLD,eps,ierr)
EPSSetOperators(eps,ham,PETSC_NULL_MAT,ierr)
EPSSetProblemType(eps,EPS_HEP,ierr)
EPSSetWhichEigenpairs(eps,EPS_SMALLEST_REAL,ierr)
```

After the problem type is set, one can specifiy the number of eigenvalues to be computed using the following routine.

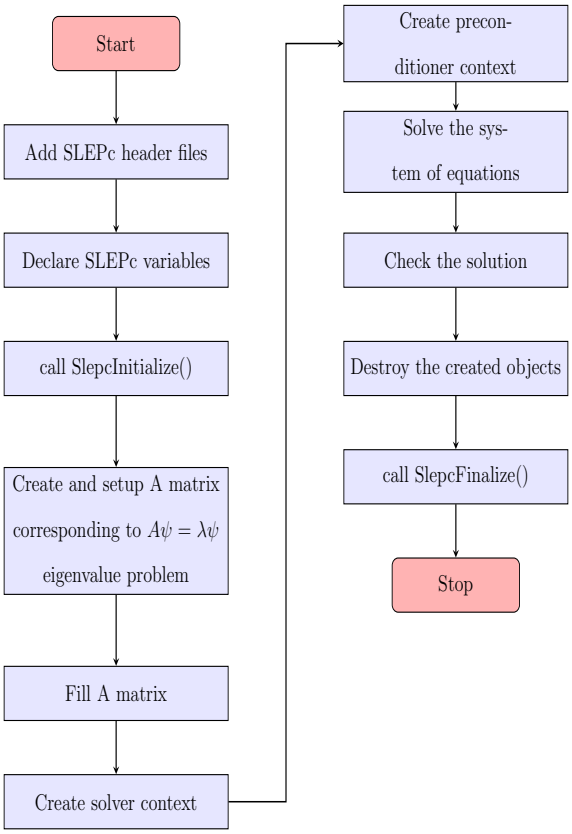

Figure 3.3: Flow of a Typical SLEPc Program

```
EPSSetDimensions(eps,nev,ncv,mpd,ierr)
```

where nev is the number of eigenvalues to be computed. The next step is to solve the eigenvalue problem to get the eigenvalues and eigenvectors using the following routine.

```
EPSSolve(eps,ierr)
```

One can check the reason for convergence of the SLEPc solver using the routine mentioned below.

```
EPSConvergedReason(eps,reason,ierr)
```

As in the case of PETSc, one has to destroy all the objects created and the program must end with a call to the SlepcFinalize() routine.

3.3 Results from the Schrödinger Solver

In this work, the SLEPc eigenvalue package was used to develop both 1D and 2D Schrödinger solvers. Both 1D and 2D Schrödinger solvers are designed in such a way as to accept any arbitrary potential profile as input and evaluate the eigen energies and wavefunctions for the system. For the case of 1D Schrödinger solver, a front end graphical user interface is built using Rapid APPlication infrastrucTURE (Rappture) toolkit [24] and the 1D Schrödinger solver is hosted as a tool on nanoHUB, titled "Bound States Calculation Lab" [25]. The tool accepts four different confining potentials: square, parabolic, triangular and v-shaped. The results from the tool are shown below.

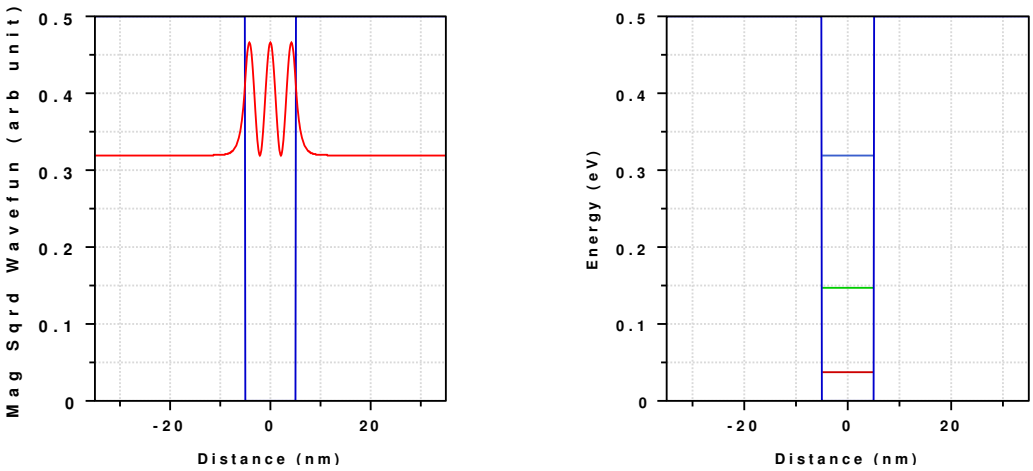

Figure 3.4: Probability Density $|\psi(x)|^2$ of the Third Eigenstate and the Eigen Energy Levels for a Square Potential Profile.

The results for a square well potential profile with a depth of 0.5eV, well width of

10nm and electron effective mass of 0.067 are shown in Figure 3.4.

Figure 3.5 shows the results for a parabolic potential profile with oscillator energy of 0.03eV and electron effective mass of 0.067. The energy levels are equidistant as excepted for a parabolic potential profile.

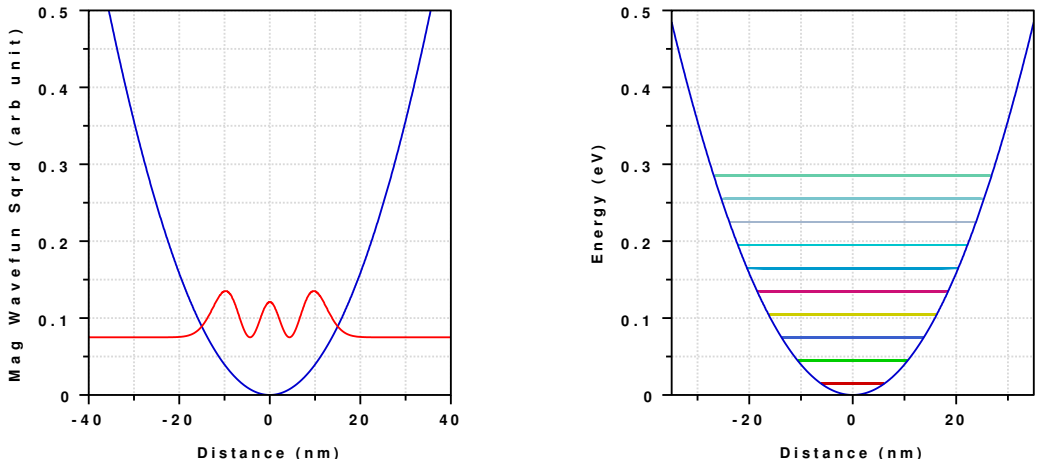

Figure 3.5: Probability Density $|\psi(x)|^2$ of the Third Eigenstate and the First Ten Eigen Energy Levels for a Parabolic Potential Profile.

Figure 3.6 shows the results for a triangular potential profile with an electric field of 10^6 V/m and electron effective mass of 0.067. The separation between the energy levels can be seen to be decreasing for higher eigenstates as expected.

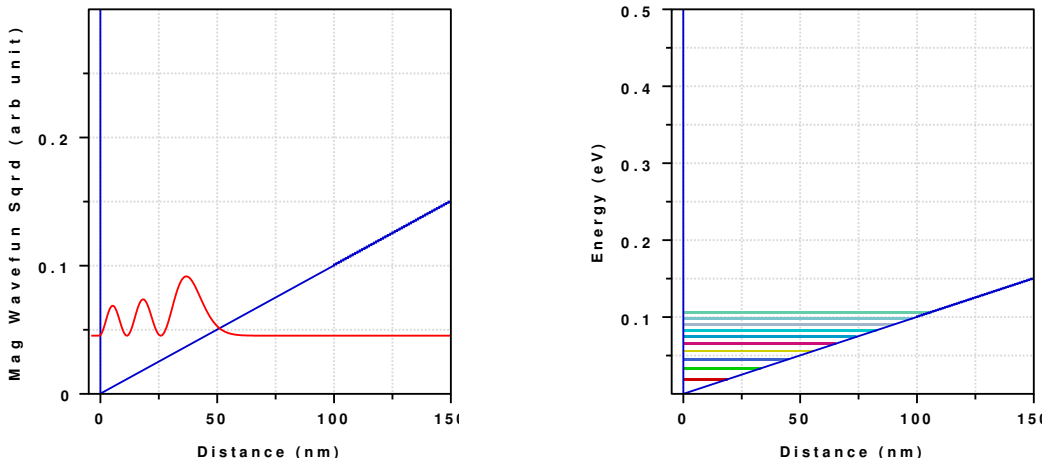

Figure 3.6: Probability Density $|\psi(x)|^2$ of the Third Eigenstate and the First Ten Eigen Energy Levels for a Triangular Potential Profile.

Figure 3.7 shows the results for a v-shaped potential profile with an electric field of 10^6 V/m and electron effective mass of 0.067. The separation between the energy levels can be seen to be decreasing for higher eigenstates as expected.

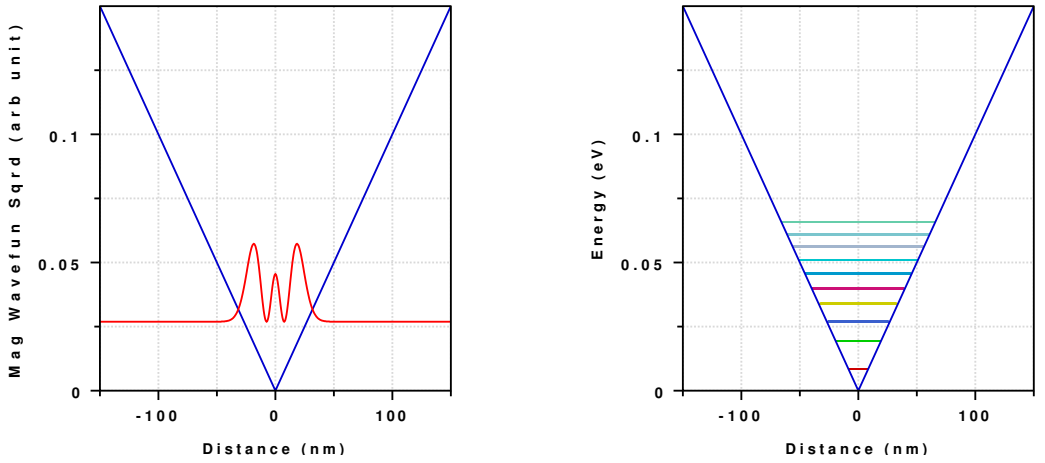

Figure 3.7: Probability Density $|\psi(x)|^2$ of the Third Eigenstate and the First Ten Eigen Energy Levels for a V-shaped Potential Profile.

The 1D Schrödinger solver is extended to 2D and a square well potential profile shown in Figure 3.8 is simulated. The electron effective mass in the barrier is taken

as 0.24 and as 0.067 in the well and the well depth is taken as 0.5 eV . The length and the width of the square well are 3 nm each, with an offset of 1 nm on all four directions.

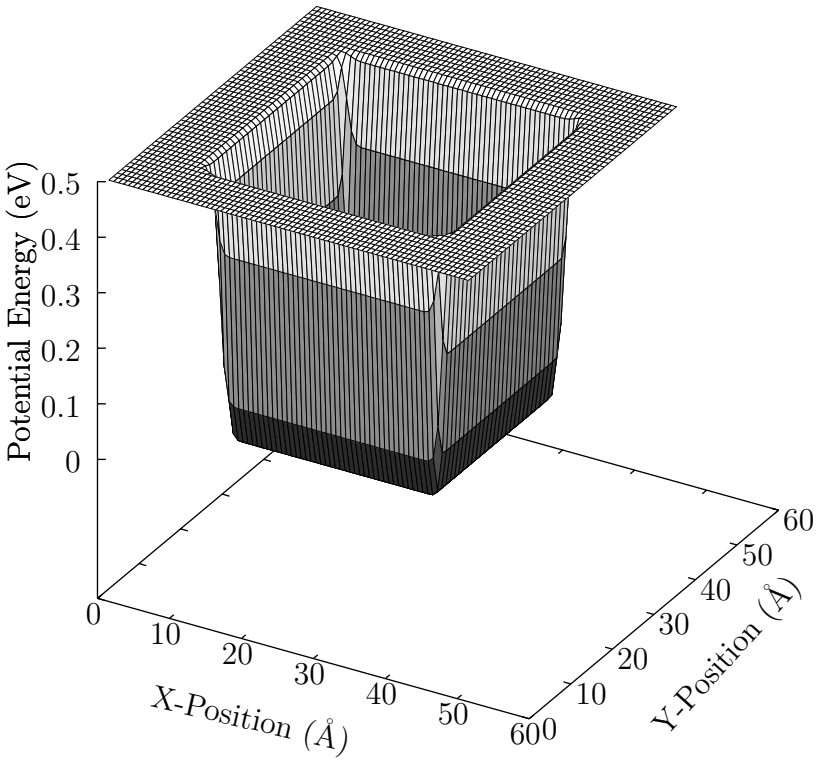

Figure 3.8: Potential Well Used for the 2D Schrödinger Solver.

The results from the 2D Schrödinger solver are given in Figure 3.9 where $|\psi(x, y)|^2$ of a few arbitrary eigen states are shown.

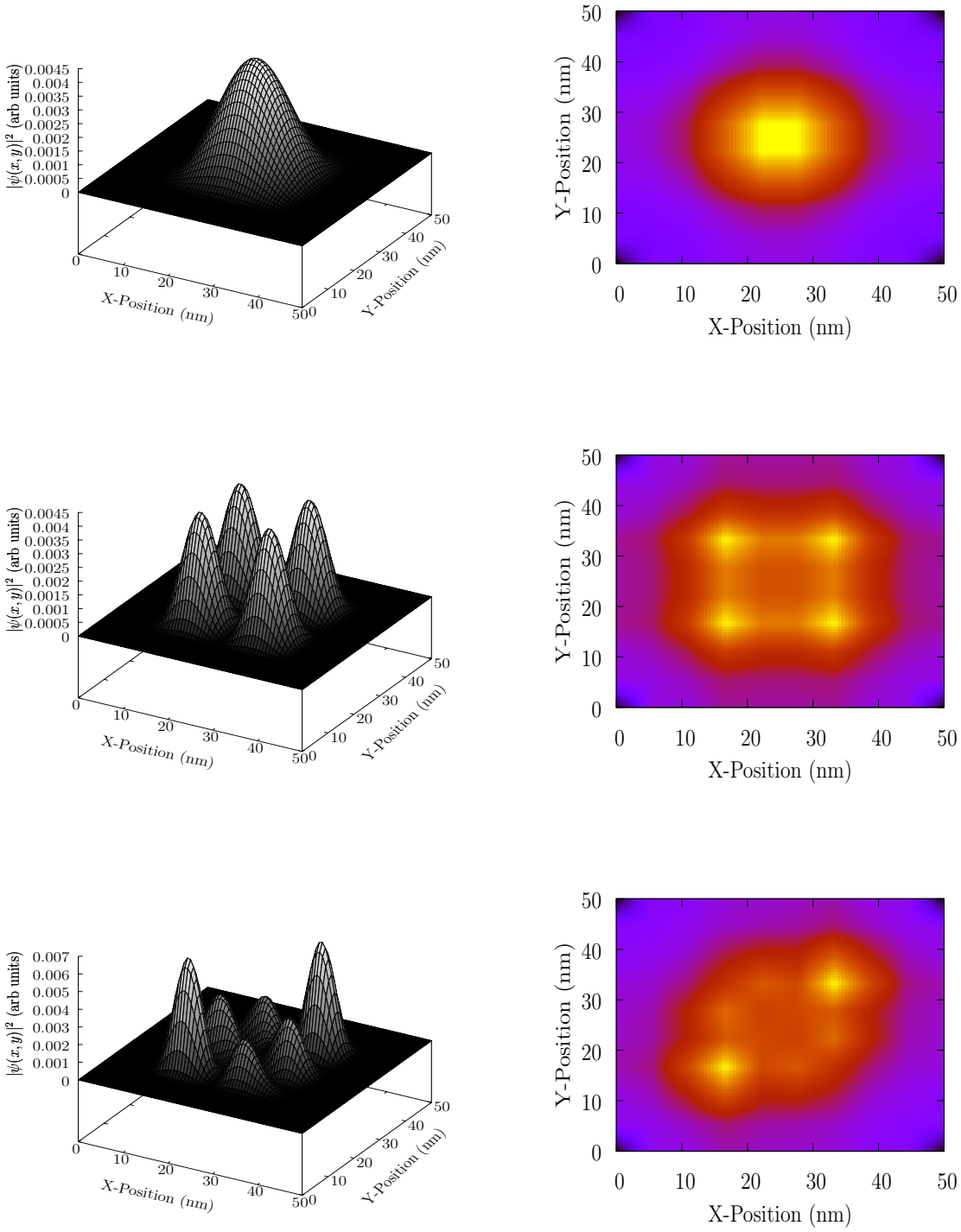

Figure 3.9: Surface Plot and Heat Map of $|\psi(x,y)|^2$ of First, Fourth and Seventh Eigen States.

Chapter 4

THE SCHRÖDINGER-POISSON SOLVER

Having developed Schrödinger and Poisson equation solvers individually, the next and final step is to couple the two solvers self-consistently. The conduction band profile from the Poisson solver is given as an input to the Schrödinger solver which then calculates the electron wavefunctions and energies. The wavefunctions are next used to evaluate the electron densities at each mesh point. The calculated electron densities are used back in the Poisson equation solver to calculate the updated conduction band profile, and this process is repeated until convergence is achieved.

This chapter starts with the description of the calculation of the electron density in a quasi-1D system quantum-mechanically. Next, the Jacobian linearization of the Quantum Poisson equation is discussed. This is followed by a section discussing the implementation flow of the Schrödinger-Poisson solver. At the end we show representative simulation results for a GaN nanowire.

4.1 Electron Density in a Quasi 1D System

For the device structure shown in of Chapter 2, which is shown here again in Figure 4.1 for completeness, the electrons are confined along the y direction due to the triangular like confinement at the $Al_{0.2}Ga_{0.8}N$-GaN interface and the confinement in the x direction is due to the negative bias applied to the gates on the surface of the device. The electrons are, thus, free to move only in z direction, leading to a quasi 1D system.

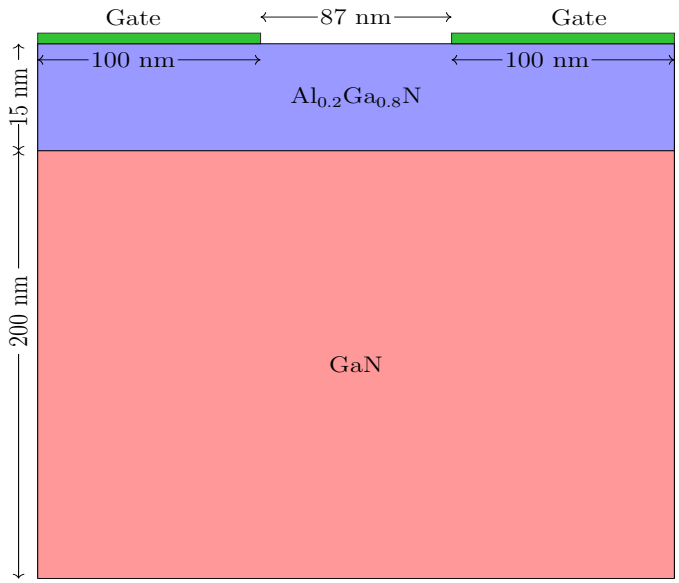

Figure 4.1: Device Structure Used for the Schrödinger-Poisson Solver.

For a 1D system, the density of states per unit volume is given by the following equation.

$$g(E) = \frac{\sqrt{2m^*}}{\pi\hbar} \frac{1}{\sqrt{E - E_c^{min}}} \tag{4.1}$$

For a quasi 1D system, the spatial confinement in the x and y direction leads to the formation of sub-bands in the conduction band. Eq.(4.1) is then modified as follows:

$$g(E) = \frac{\sqrt{2m^*}}{\pi\hbar} \frac{1}{\sqrt{E - E_c^p}} \mathcal{H}(E - E_c^p) \tag{4.2}$$

where $\mathcal{H}$ is the Heaviside step function and E_c^i is the energy of the p^{th} sub-band in the conduction band. Using the Fermi-Dirac statistics, the electron density in the p^{th} sub-band can be calculated as,

$$\begin{aligned} n^p &= \frac{\sqrt{2m^*}}{\pi\hbar} \int_{E_c^p}^{\infty} \frac{1}{\sqrt{E - E_c^p}} \frac{1}{1 + e^{\frac{E - E_F}{K_b T}}} dE \\ &= \frac{\sqrt{2m^* K_b T}}{\pi\hbar} \int_0^{\infty} \frac{\epsilon^{\frac{-1}{2}}}{1 + e^{\epsilon - \eta_p}} dE \\ &= \frac{\sqrt{2m^* K_b T}}{\pi\hbar} F_{-1/2}(\eta_p) \end{aligned} \tag{4.3}$$

where $\eta_p = \frac{E_F - E_c^p}{K_b T}$. The total electron density at a mesh point can be written as,

$$n(i,j) = \sum_{p=1}^{Max} n^p |\psi^p(i,j)|^2 \tag{4.4}$$

In the above equation, Max is a sub-band number chosen in a manner such that the probability of finding the electron with energy greater than E_c^{Max} can be taken to be zero.

4.2 Jacobian Linearization of the Quantum Poisson Solver

The Poisson equation is discussed in Chapter 2 and is given by:

$$\begin{aligned} \nabla \cdot (\epsilon \nabla V^{k+1}) &= \frac{-q}{\epsilon_o} \left(p^{k+1} - n^{k+1} + DOP \right) \\ &= f(V^{k+1}) \end{aligned} \tag{4.5}$$

In the classical case, the electron density n is calculated either from the Boltzmann approximation of the Fermi-Dirac statistics or by using Fermi-Dirac statistics directly. For the case of Quantum Poisson solver, the electron density n is calculated as shown in Section 4.1. Following a similar procedure to Chapter 2, Eq.(4.5) can be written in terms of the update δ as

$$\nabla \cdot (\epsilon \nabla \delta^{k+1}) = f(V^{k+1}) - \nabla \cdot (\epsilon \nabla V^k) \tag{4.6}$$

Using the Taylor series expansion around V^k, $f(V^{k+1})$ can be approximated to first order as,

$$f(V^{k+1}) = f(V^k) + \left. \frac{\partial f(V^{k+1})}{\partial V^{k+1}} \right|_{V^k} (\delta^{k+1}) \tag{4.7}$$

Using Eq.(4.5), $\left. \frac{\partial f(V^{k+1})}{\partial V^{k+1}} \right|_{V^k}$ can be written as,

$$\left. \frac{\partial f(V^{k+1})}{\partial V^{k+1}} \right|_{V^k} = -\frac{q}{\epsilon_o} \left(-\frac{p^k}{K_B T} - \left. \frac{\partial n^{k+1}}{\partial V^{k+1}} \right|_{V^k} \right) \tag{4.8}$$

To evaluate Eq.(4.8), one has to calculate $\left.\frac{\partial n^{k+1}}{\partial V^{k+1}}\right|_{V^k}$. This can be rewritten as,

$$\left.\frac{\partial n^{k+1}}{\partial V^{k+1}}\right|_{V^k} = \left(\frac{\partial n^{k+1}}{\partial \eta_p^{k+1}}\right)\Bigg|_{V^k} \left(\frac{\partial \eta_p^{k+1}}{\partial V^{k+1}}\right)\Bigg|_{V^k} \tag{4.9}$$

To evaluate $\left.\frac{\partial n^{k+1}}{\partial \eta_p^{k+1}}\right|_{V^k}$, one assumes that the wavefunction has no explicit dependence on the potential V. This gives,

$$\left.\frac{\partial n^{k+1}}{\partial \eta_p^{k+1}}\right|_{V^k} = \frac{\sqrt{2m^* K_B T}}{\pi\hbar} \sum_{p=1}^{Max} \left.\frac{\partial F_{-\frac{1}{2}}(\eta_p^{k+1})}{\partial \eta_p^{k+1}}\right|_{V^k} |\psi_m^{k+1}|^2 \tag{4.10}$$

Using the standard properties of Fermi-Dirac integrals, Eq.(4.10) can be rewritten as,

$$\left.\frac{\partial n^{k+1}}{\partial \eta_p^{k+1}}\right|_{V^k} = \frac{\sqrt{2m^* K_B T}}{\pi\hbar} \sum_{p=1}^{Max} \frac{\Gamma(1/2)}{\Gamma(-1/2)} F_{-\frac{3}{2}}(\eta_p^k) |\psi_p^{k+1}|^2 \tag{4.11}$$

Hence, in order to calculate $\left.\frac{\partial n^{k+1}}{\partial \eta_p^{k+1}}\right|_{V^k}$, one has to compute $F_{-\frac{3}{2}}(\eta_p^{k+1})$. Note that analytical approximations are not available for the evaluation of the Fermi-Dirac integrals of order $\frac{-3}{2}$. Hence, one needs to evaluate the integral numerically, but this would add a significant computational overhead to the solver. This problem is overcome by using standard definition of a derivative as shown below:

$$\left.\frac{\partial F_{-\frac{1}{2}}(\eta_p^{k+1})}{\partial \eta_p^{k+1}}\right|_{V^k} \approx \frac{F_{-\frac{1}{2}}(\eta_p^k + h) - F_{-\frac{1}{2}}(\eta_p^k)}{h}, \tag{4.12}$$

where h is taken to be $\frac{\eta_p^i}{100}$. Thus, Eq.(4.10) can be rewritten as,

$$\boxed{\left.\frac{\partial n^{k+1}}{\partial \eta_p^{k+1}}\right|_{V^k} = \frac{\sqrt{2m^* K_B T}}{\pi\hbar} \sum_{p=1}^{Max} \frac{F_{-\frac{1}{2}}(\eta_p^k + h) - F_{-\frac{1}{2}}(\eta_p^k)}{h} |\psi_p^{k+1}|^2} \tag{4.13}$$

To compute the second term of Eq.(4.9), an assumption is made that the eigenenergies of the $(k+1)^{th}$ iteration follow the potential update δ^{k+1} as,

$$E_p^{k+1} = E_p^k - \delta^{k+1}. \tag{4.14}$$

The above equation can be rewritten in terms of η_p as,

$$\eta_p^{k+1} = \eta_p^k + \frac{V^{k+1} - V^k}{K_B T} \tag{4.15}$$

Then, $\left.\frac{\partial \eta_p^{k+1}}{\partial V^{k+1}}\right|_{V^k}$ is given by

$$\boxed{\left.\frac{\partial \eta_p^{k+1}}{\partial V^{k+1}}\right|_{V^k} = \frac{1}{K_B T}} \tag{4.16}$$

Using Eqs. (4.9), (4.10) and (4.16), Eq.(4.8) reduces to,

$$\left.\frac{\partial f(V^{k+1})}{\partial V^{k+1}}\right|_{V^k} = \frac{q}{K_b T}(p^k + n_Q^k) \tag{4.17}$$

where $n_Q^k = \frac{\sqrt{2m^* K_b T}}{\pi\hbar} \sum_{p=1}^{Max} \left(\frac{F_{-\frac{1}{2}}(\eta_p^k + h) - F_{-\frac{1}{2}}(\eta_p^k)}{h} \right) |\psi_p^k|^2$. Substituting Eq.(4.17) into Eq.(4.7), the differential equation to solve becomes:

$$\boxed{\nabla \cdot (\epsilon \nabla \delta^{k+1}) - \frac{q}{K_b T}\left(p^k + n_Q^k\right)\delta^{k+1} = f(V^k) - \nabla \cdot (\epsilon \nabla V^k)} \tag{4.18}$$

4.3 Implementation of the Schrödinger-Poisson Solver

To solve the differential equation given in Eq.(4.18), first a section of the simulation space is identified as the Schrödinger domain. This is shown in Figure 4.2, where the dotted region is the Schrödinger domain. The Schrödinger equation is solved for

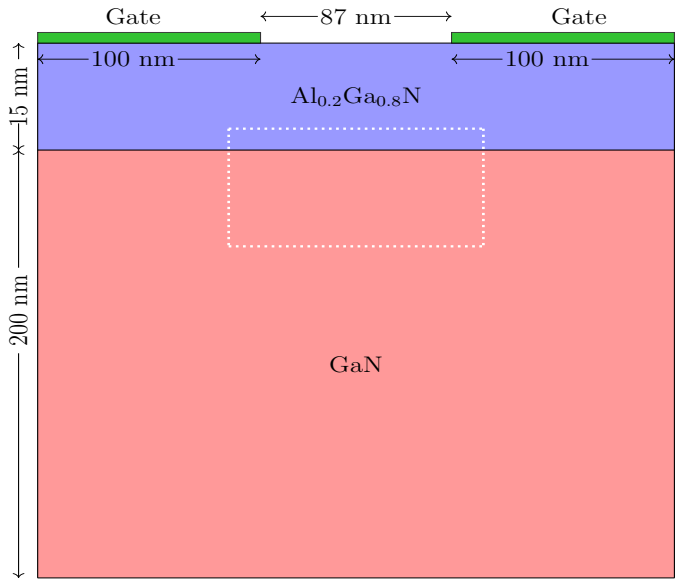

Figure 4.2: Simulation Space Indicating the Schrödinger Domain

the system in the highlighted domain and the wavefunctions generated are used to evaluate the electron density in this domain. To evaluate the electron density, it is evident from Eq.(4.13) that one has to evaluate Fermi-Dirac integral of order -1/2, which is accomplished using standard analytical approximations [26]. The calculated electron density is used back in the Poisson solver to generate updated conduction band profiles. This process is repeated until convergence is achieved. The complete implementation of the Schrödinger-Poisson solver is outlined in Figure 4.3.

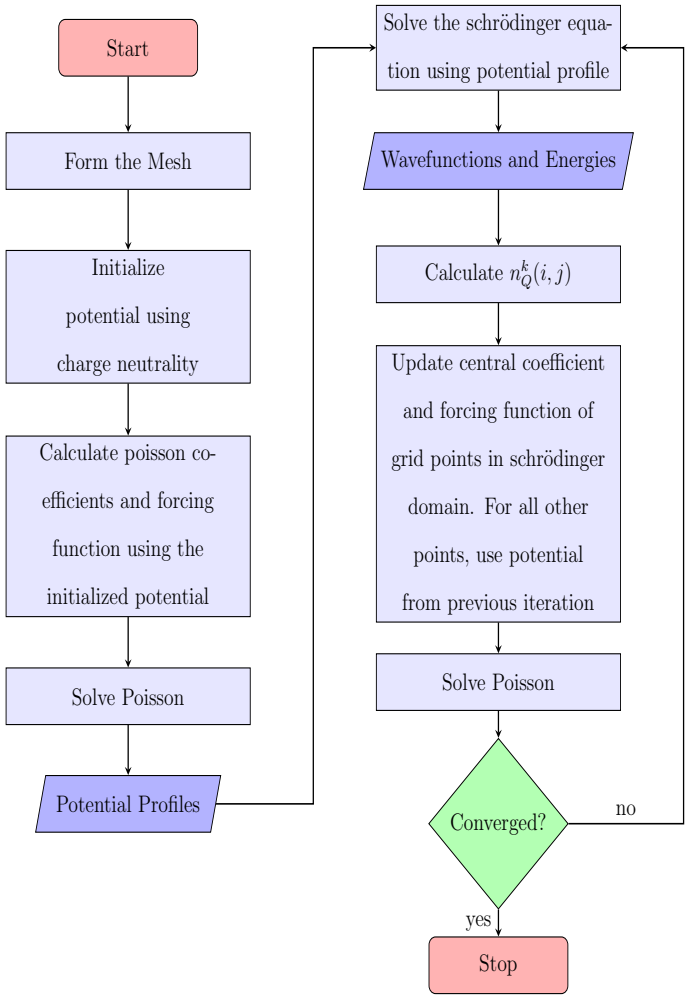

Figure 4.3: Flow Chart of the Schrödinger-Poisson Solver

4.4 Results from the Schrödinger-Poisson Solver

In this section, the results from the Schrödinger-Poisson solver are presented for the device structure shown in Figure 4.2. The self-consistent potential profile, electron concentration and the eigen wavefunctions in the device are shown, along with the convergence properties of the solver.

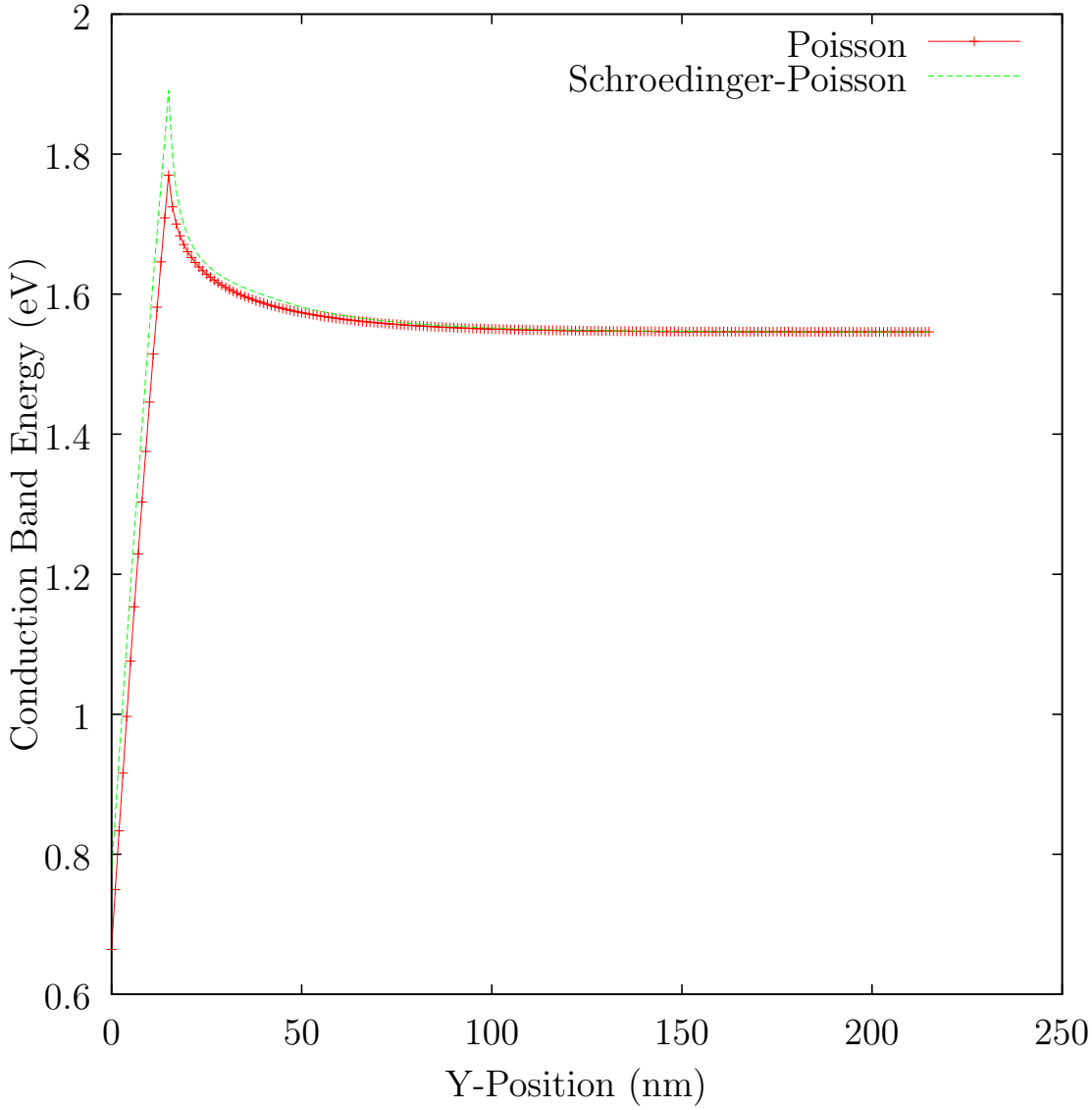

Figure 4.4: Comparison of Potential Profiles Between Self-Consistent Schrödinger-Poisson Solver and Standalone Poisson Solver Along the Y-direction. for the Structure Shown In Figure (4.2).

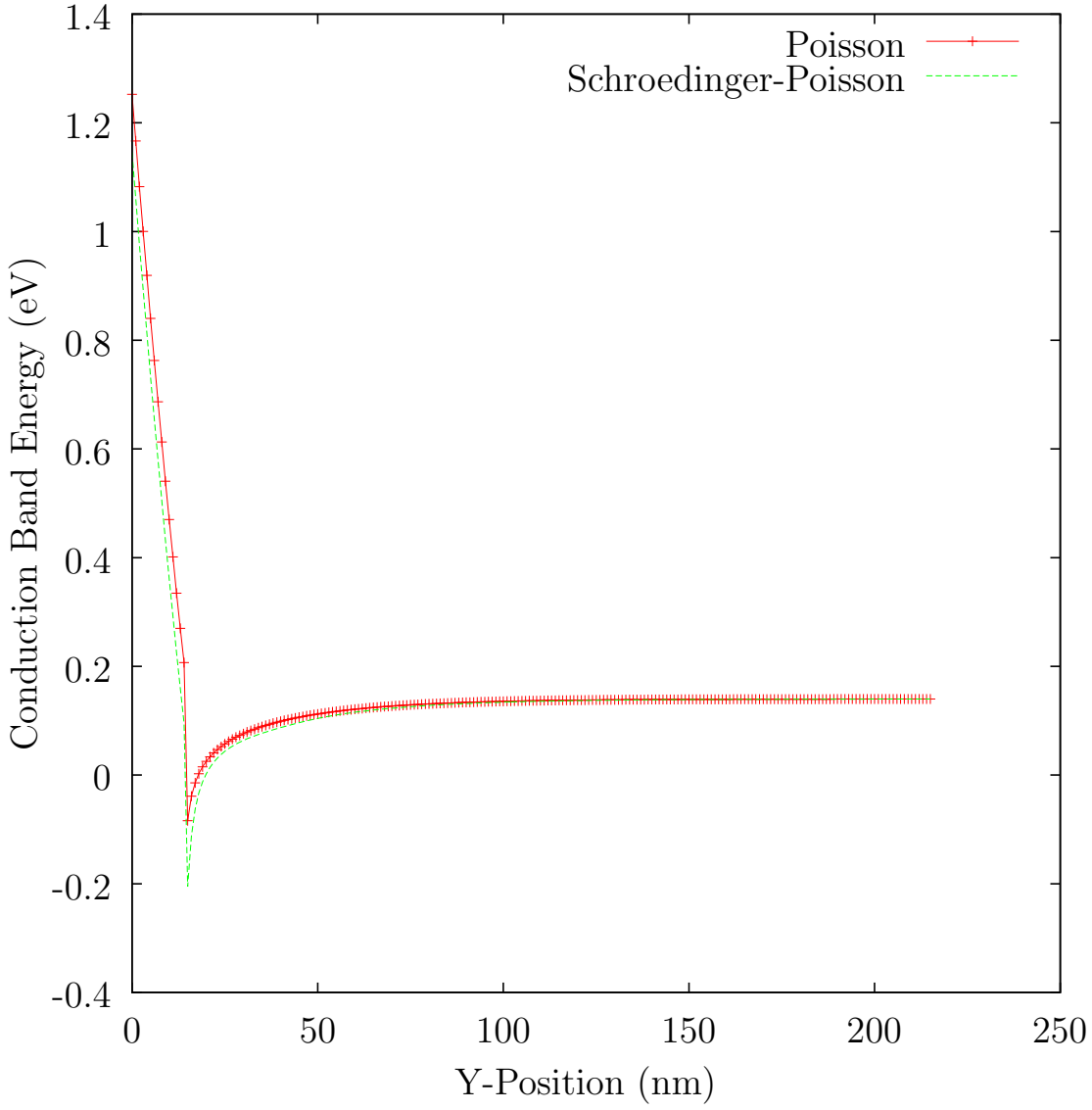

Figure 4.5: Comparison of Conduction Band Profiles Between Self-Consistent Schrödinger-Poisson Solver and Standalone Poisson Solver Along the Y-direction.

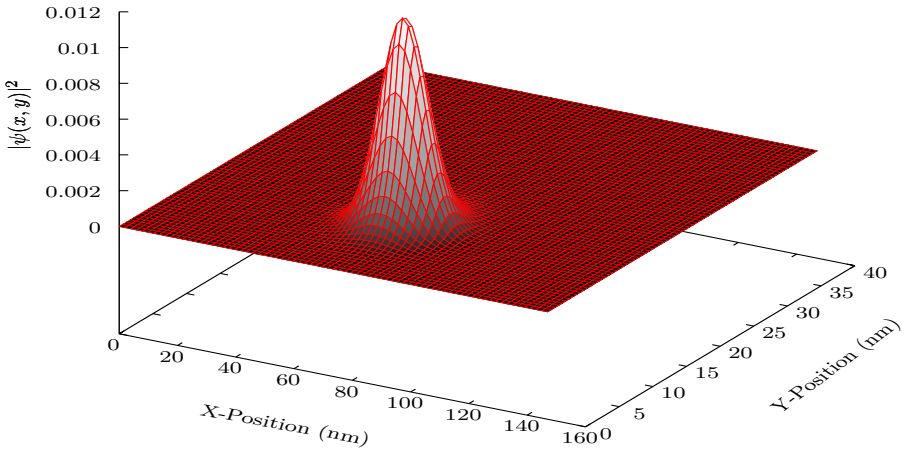

Figure 4.6: Probability Density $|\psi(x, y)|^2$ for the First Energy Eigenstate.

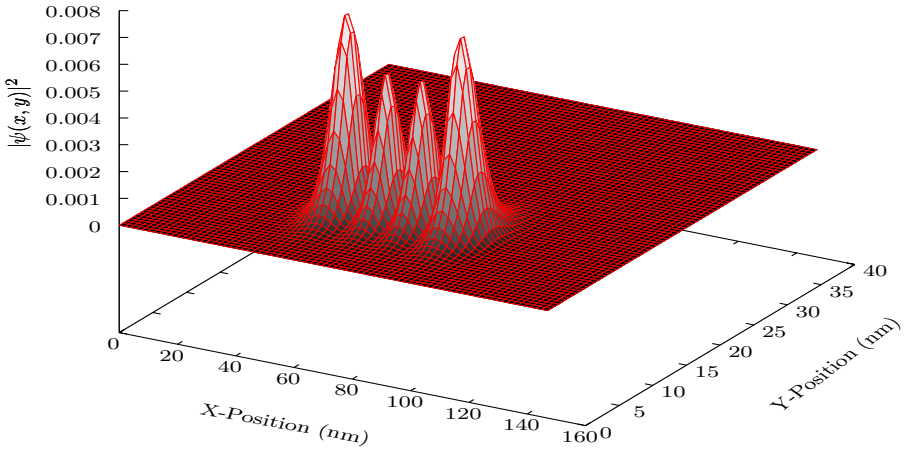

Figure 4.7: Probability Density $|\psi(x, y)|^2$ for the Fourth Energy Eigenstate.

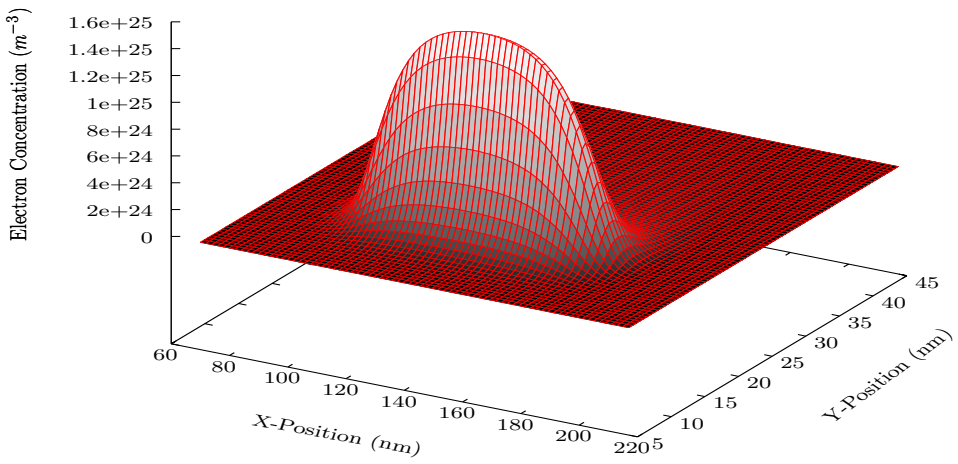

Figure 4.8: Self-Consistent Quantum Mechanical Electron Density In the Nanowire Region.

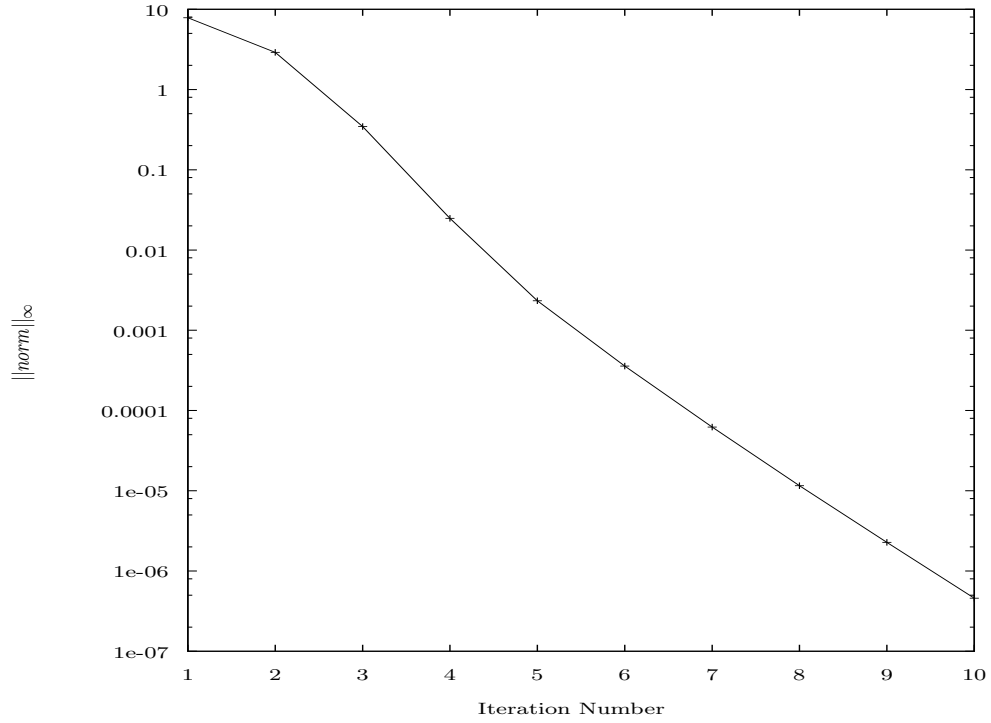

Figure 4.9: Convergence of the Schrödinger-Poisson Solver.

Chapter 5

CONCLUSIONS AND FUTURE WORKS

As the search for robust and efficient algorithms to numerically solve the differential equation at hand continues, software tools such as PETSc and SLEPc offer the user with a host of efficiently programmed, ready-to-use numerical methods which can be applied to all facets of computational science. This drastically reduces the amount of time spent on programming and debugging numerical methods, simultaneously providing enormous flexibility to the user in terms of switching between or combining these methods.

In this work, this opportunity is exploited by using PETSc and SLEPc to develop a two dimensional self-consistent Schrödinger-Poisson solver. Although the results presented in this work are focussed on GaN based heterostructure systems, the Schrödinger-Poisson solver developed here is equally applicable to other semiconductor devices. The flowcharts of the PETSc and SLEPc solvers presented in this work can serve as a blueprint if one decides to incorporate these software tools into their research.

It is imperative to remind, however, that the functionality of PETSc and SLEPc presented in this work is not exhaustive. Both PETSc and SLPEc offer many routines for the implementation of large-scale application codes on parallel computers, intensive error checking and new functionality is constantly added to the existing base.

From the view point of the Schrödinger-Poisson solver developed as a part of this work, parabolic dispersion is assumed at the Γ point. To accurately use the solver for other materials, one has to couple a band structure program, such as the Tight Binding model or the Empirical Pseudopotential model, with the Schrödinger-Poisson

solver and solve the system self-consistently.

REFERENCES

[1] Teja Singh, Sundar Rangarajan, Deepesh John, Carson Henrion, Shane Southard, Hugh McIntyre, Amy Novak, Stephen Kosonocky, Ravi Jotwani, Alex Schaefer, et al. 3.2 zen: A next-generation high-performance× 86 core. In *2017 IEEE International Solid-State Circuits Conference (ISSCC)*, pages 52–53. IEEE, 2017.

[2] Eyal Fayneh, Marcelo Yuffe, Ernest Knoll, Michael Zelikson, Muhammad Abozaed, Yair Talker, Ziv Shmuely, and Saher Abu Rahme. 4.1 14nm 6th-generation core processor soc with low power consumption and improved performance. In *2016 IEEE International Solid-State Circuits Conference (ISSCC)*, pages 72–73. IEEE, 2016.

[3] Satish Balay, Shrirang Abhyankar, Mark F. Adams, Jed Brown, Peter Brune, Kris Buschelman, Lisandro Dalcin, Alp Dener, Victor Eijkhout, William D. Gropp, Dmitry Karpeyev, Dinesh Kaushik, Matthew G. Knepley, Dave A. May, Lois Curfman McInnes, Richard Tran Mills, Todd Munson, Karl Rupp, Patrick Sanan, Barry F. Smith, Stefano Zampini, Hong Zhang, and Hong Zhang. PETSc users manual. Technical Report ANL-95/11 - Revision 3.14, Argonne National Laboratory, 2020.

[4] Jose E Roman, Carmen Campos, Eloy Romero, and Andrés Tomás. Slepc users manual. *D. Sistemes Informatics i Computació Universitat Politecnica de Valencia, Valencia, Spain, Report No. DSIC-II/24/02*, 2015.

[5] Manuel Valera, Mary P. Thomas, Mariangel Garcia, and Jose E. Castillo. Parallel implementation of a PETSc-Based framework for the general curvilinear coastal ocean model. *Journal of Marine Science and Engineering*, 7(6), 2019.

[6] B. H. Dennis, R. C. Eberhart, G. S. Dulikravich, and S.W. Radons. Finite-element simulation of cooling of realistic 3-d human head and neck. *J Biomech Eng*, 125(6):832–840, 2003.

[7] Belkis Erzincanli and Mehmet Sahin. The numerical simulation of the wing kinematics effects on near wake topology and aerodynamic performance in hovering drosophila flight. *Computers & Fluids*, 122:90–110, 2015.

[8] IP Smorchkova, L Chen, T Mates, L Shen, S Heikman, B Moran, S Keller, SP DenBaars, JS Speck, and UK Mishra. Aln/gan and (al, ga) n/aln/gan two-dimensional electron gas structures grown by plasma-assisted molecular-beam epitaxy. *Journal of Applied Physics*, 90(10):5196–5201, 2001.

[9] T Paul Chow and Ritu Tyagi. Wide bandgap compound semiconductors for superior high-voltage power devices. In *[1993] Proceedings of the 5th International Symposium on Power Semiconductor Devices and ICs*, pages 84–88. IEEE, 1993.

[10] Yi-Feng Wu, David Kapolnek, James P Ibbetson, Primit Parikh, Bernd P Keller, and Umesh K Mishra. Very-high power density algan/gan hemts. *IEEE Transactions on Electron Devices*, 48(3):586–590, 2001.

[11] O Ambacher, J Smart, JR Shealy, NG Weimann, K Chu, M Murphy, WJ Schaff, LF Eastman, R Dimitrov, L Wittmer, et al. Two-dimensional electron gases induced by spontaneous and piezoelectric polarization charges in n-and ga-face algan/gan heterostructures. *Journal of applied physics*, 85(6):3222–3233, 1999.

[12] Bin Lu, Elison Matioli, and Tomas Palacios. Tri-gate normally-off gan power misfet. *IEEE Electron Device Letters*, 33(3):360–362, 2012.

[13] Muhammad Fahlesa Fatahilah, Klaas Strempel, Feng Yu, Sindhuri Vodapally, Andreas Waag, and Hutomo Suryo Wasisto. 3d gan nanoarchitecture for field-effect transistors. *Micro and Nano Engineering*, 3:59–81, 2019.

[14] John Frederick Nye et al. *Physical properties of crystals: their representation by tensors and matrices*. Oxford University Press, 1985.

[15] Oliver Ambacher. Growth and applications of group iii-nitrides. *Journal of physics D: Applied physics*, 31(20):2653, 1998.

[16] William Shockley. Electrons and holes in semiconductors: with applications to transistor electronics. 1953.

[17] AH Marshak. On the inappropriate use of the intrinsic level as a measure of the electrostatic potential in semiconductor devices. *IEEE Electron Device Letters*, 6(3):128–129, 1985.

[18] Alan H Marshak. Transport equations for highly doped devices and heterostructures. *Solid-State Electronics*, 30(11):1089–1093, 1987.

[19] MS Lundstrom, RJ Schwartz, and JL Gray. Transport equations for the analysis of heavily doped semiconductor devices. *Solid-State Electronics*, 24(3):195–202, 1981.

[20] Mark S Lundstrom and Robert J Schuelke. Numerical analysis of heterostructure semiconductor devices. *IEEE Transactions on Electron Devices*, 30(9):1151–1159, 1983.

[21] Yousef Saad. *Iterative methods for sparse linear systems*. SIAM, 2003.

[22] ATLAS User Manual ATLAS. Silvaco international. *Santa Clara, CA*, 2015.

[23] Mykhailo Povolotskyi. *Theoretical Study of Electronic and Optical Properties of Low-Dimensional Semiconductor Nanostructures*. PhD thesis, Università di Roma "Tor Vergata", 2004.

[24] M McLennan. The rappture toolkit. *Article (CrossRef Link)*, 2004.

[25] Pranay Kumar Reddy Baikadi, Michael Povolotskyi, Viswanathan Naveen Kumar Nolastname, Dragica Vasileska, Xufeng Wang, and Gerhard Klimeck. Bound states calculation lab, April 2020.

[26] X Aymerich-Humet, F Serra-Mestres, and J Millan. A generalized approximation of the fermi–dirac integrals. *Journal of applied physics*, 54(5):2850–2851, 1983.