Numerical Simulation Poisson

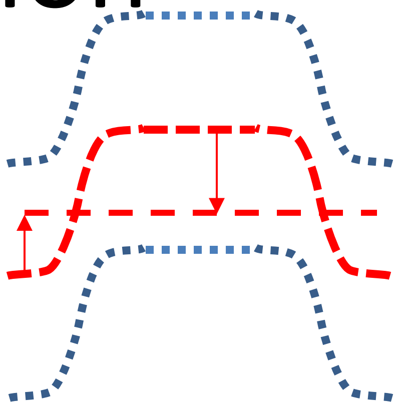
Udayan Ganguly

8/2/2024

Reference

- Computational Nanoelectronics
  - D Vasileska, S.M Goodnick, G. Klimeck
  - CRC Press
  - Chapter 4
  - link

Poisson Numerical Solution



- Linearization
- $\frac{d^2V}{dx^2} = -\frac{q\rho}{\varepsilon}$ where $\rho = N_D + p(x) - N_A - n(x)$
- $n = n_i \exp\left(\frac{E_F - E_i}{kT}\right) ➔ \frac{n}{n_i} = \exp(\varphi)$
- Where $\varphi = \frac{E_F - E_i}{kT}$ as unitless energy where $\mathrm{E_F} - E_i = q(V_F + V)$
- What is p?

V_F is constant with x

- Similarly $\frac{\mathrm{p}}{n_i} = \exp(-\varphi)$
- $\frac{d^2V}{dx^2} = -\frac{d^2E_i}{dx^2} = kT\frac{d^2\varphi}{dx^2}$
- $\frac{d^2\varphi}{dx^2} = -\frac{q^2n_i}{kT\varepsilon}(\exp(-\varphi) - \exp(\varphi) + C)$ where $C = \frac{N_D - N_A}{n_i}$
- $\frac{d^2\varphi}{dX^2} = -(\exp(-\varphi) - \exp(\varphi) + C)$ where $\frac{1}{{L_D}^2} = \frac{q^2n_i}{kT\varepsilon}$
- where X=x/L_D dimensionless distance

Where is the sign error?

Moving from known solution to new solution with small change

- $\frac{d^2\varphi}{dX^2} = -(\exp(-\varphi) - \exp(\varphi) + C)$
- $\varphi_{new} = \varphi_{old} + \ \delta$ where δ is small
- $\frac{d^2\varphi_{new}}{dX^2} = -\{\exp -(\varphi_{old} + \ \delta) - \exp(\varphi_{old} + \ \delta) + C\}$
- $\frac{d^2\varphi_{new}}{dX^2}$ $=-\{(\exp(-\varphi_{old})\ ) - \exp(\varphi_{old}) + C) - (\varphi_{new} - \varphi_{old})(\exp(-\varphi_{old})\ ) + \exp(\varphi_{old})\}$
- $\frac{d^2\varphi_{new}}{dX^2} - \varphi_{new}\ (e^{-\varphi_{old}} + e^{\varphi_{old}})$ $=-\{(e^{-\varphi_{old}} + e^{\varphi_{old}} + C) + \varphi_{old}\ (e^{-\varphi_{old}} + e^{\varphi_{old}}\}$

Separate out φ_{new} to the right and convert to p and n

$$\frac{1}{\Delta^2}\varphi_{i+1}^{n+1} - \left(\frac{2}{\Delta^2} + \boxed{p_i + n_i}\right)\varphi_i^{n+1} + \frac{1}{\Delta^2}\varphi_{i-1}^{n+1} = -[p_i - n_i + C_i] - \varphi_i^n(p_i + n_i)$$

Where is this terms come from?

Poisson

$$\nabla \cdot \varepsilon \nabla V = -\left(p - n + N_D^+ - N_A^- \right)$$

$$\frac{1}{\Delta^2}\varphi_{i+1}^{n+1} - \left(\frac{2}{\Delta^2} + p_i + n_i \right)\varphi_i^{n+1} + \frac{1}{\Delta^2}\varphi_{i-1}^{n+1} = -\left[p_i - n_i + C_i \right] - \varphi_i^n \left(p_i + n_i \right)$$

$$a^n \varphi_{i+1}^{n+1} - b^n \varphi_i^{n+1} + c^n \varphi_{i-1}^{n+1} = f^n$$

New potential (n + 1 th iteration)

Old potential

Solve numerically (example: pn junction)

1. Choose Guess Potential $V_{old}(x)$
2. Find carrier concentration $n(x), p(x)$ from V_{old} using equilibrium carrier statistics i.e. $\rho(V_{old})$
3. Using Poisson compute $V_{new}(\rho)$
4. Calculate error $E(x) = V_{new} - V_{old}$
5. If Error<min spec then stop
6. If Error > min error spec then use V_{new} as guess potential in step 1; continue until Error E negligible

Exercise: Please draw a flow chart of this strategy!

Linearized Poisson Equation
$\phi \rightarrow \phi + \delta$ where $\delta = \phi^{new} - \phi^{old}$

- Finite difference discretization:
  - Potential varies linearly between mesh points
  - Electric field is constant between mesh points
- Linearization → Diagonally-dominant coefficient matrix A is obtained

$$\frac{d^2V^{new}}{dx^2} = -\frac{en_i}{\varepsilon}\left(e^{-V^{old}/V_T} - e^{V^{old}/V_T} + C/n_i\right) + \frac{en_i}{\varepsilon}\delta\left(e^{-V^{old}/V_T} + e^{V^{old}/V_T}\right)$$

$$\frac{d^2V^{new}}{dx^2} - \frac{en_i}{\varepsilon V_T}\left(e^{-V^{old}/V_T} + e^{V^{old}/V_T}\right)V^{new} = -\frac{en_i}{\varepsilon}\left(e^{-V^{old}/V_T} - e^{V^{old}/V_T} + C/n_i\right) -$$

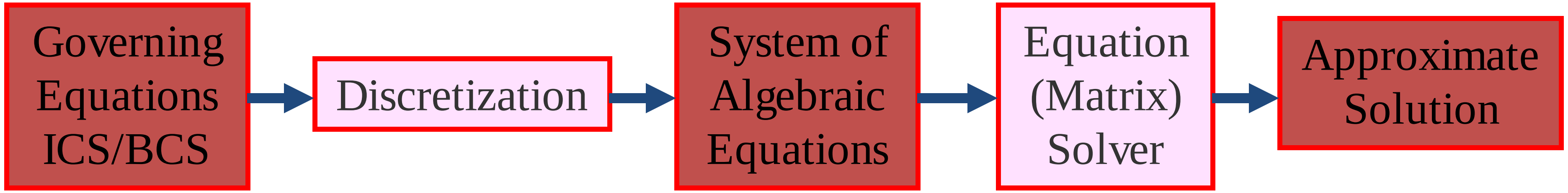
$$-\frac{en_i}{\varepsilon V_T}\left(e^{-V^{old}/V_T} + e^{V^{old}/V_T}\right)V^{old}$$

$$\delta = V^{new} - V^{old}$$

Be Unitless – Use Fundamental Scale (due to de Mari)

Variable	**Scaling Variable**	**Formula**
Space	Intrinsic Debye length (N=n_i)	$L = \sqrt{\frac{\varepsilon k_B T}{q^2 N}}$
	Extrinsic Debye length (N=N_{max})	
Potential	Thermal voltage	$V* = \frac{k_B T}{q}$
Carrier concentration	Intrinsic concentration	N=n_i
	Maximum doping concentration	N=N_{max}
Diffusion coefficient	Practical unit	$D = 1\frac{cm^2}{s}$
	Maximum diffusion coefficient	$D = D_{max}$
Mobility		$M = \frac{D}{V*}$
Generation-Recombination		$R = \frac{DN}{L^2}$
Time		$T = \frac{L^2}{D}$

Numerical Solution Details



Governing Equations ICS/BCS → Discretization → System of Algebraic Equations → Equation (Matrix) Solver → Approximate Solution

Continuous Solutions

Finite-Difference

Finite-Volume

Finite-Element

Spectral

Boundary Element

Hybrid

Discrete Nodal Values

Tridiagonal

SOR

Gauss-Seidel

Krylov

Multigrid

$\varphi_i\,(x,y,z,t)$

$p\ (x,y,z,t)$

$n\ (x,y,z,t)$

Reference (optional)
https://nanohub.org/resources/1542/download/numericalanalysis_ppt.pdf

D. Vasileska, *EEE533 Semiconductor Device and Process Simulation* Lecture Notes, Arizona State University, Tempe, AZ.

Method

1. Discretize Poisson’s Equation

   Get $[P] * [V_{new}] = f[V_{old}]$ where [P] is tri-diagonal and [Vnew] is unknown

2. To solve $[P]V = f$ use LU decomposition

   decompose [P]=LU (easy method)

where $LUV = f$

As LA=f where A=UV; solve for A (easy method)

Then knowing UV=A; solve for V (easy method)

Alan Turing in 1948

LU Decomposition

- If LU decomposition exists, then for a tri-diagonal matrix **A**, resulting from the finite-difference discretization of the 1D Poisson equation, one can write

$$\underbrace{\begin{pmatrix} a_1 & c_1 & & & \\ b_2 & a_2 & c_2 & & \\ & \cdots & \cdots & \cdots & \\ & & b_{n-1} & a_{n-1} & c_{n-1} \\ & & & b_n & a_n \end{pmatrix}}_{\text{Tri diagonal}} = \underbrace{\begin{pmatrix} 1 & & & & \\ \beta_2 & 1 & & & \\ & \beta_3 & \cdots & & \\ & & \cdots & \cdots & \\ & & & \beta_n & 1 \end{pmatrix}}_{L} \underbrace{\begin{pmatrix} \alpha_1 & c_1 & & & \\ & \alpha_2 & c_2 & & \\ & & \cdots & \cdots & \\ & & & \cdots & c_{n-1} \\ & & & & \alpha_n \end{pmatrix}}_{U}$$

where $\boxed{\alpha_1 = a_1,}\quad \beta_k = \dfrac{b_k}{\alpha_{k-1}}, \quad \alpha_k = a_k - \beta_k c_{k-1}, \quad k = 2,3,\ldots,n$

LU Decomposition

- If LU decomposition exists, then for a tridiagonal matrix **A**, resulting from the finite-difference discretization of the 1D Poisson equation, one can write

X

$$\begin{pmatrix} a_1 & c_1 & & & \\ b_2 & a_2 & c_2 & & \\ & \cdots & \cdots & \cdots & \\ & & b_{n-1} & a_{n-1} & c_{n-1} \\ & & & b_n & a_n \end{pmatrix} = \begin{pmatrix} 1 & & & & \\ \beta_2 & 1 & & & \\ & \beta_3 & \cdots & & \\ & & \cdots & \cdots & \\ & & & \beta_n & 1 \end{pmatrix} \begin{pmatrix} \alpha_1 & c_1 & & & \\ & \alpha_2 & c_2 & & \\ & & \cdots & \cdots & \\ & & & \cdots & c_{n-1} \\ & & & & \alpha_n \end{pmatrix}$$

where $\alpha_1 = a_1\,,\quad \boxed{\beta_k = \dfrac{b_k}{\alpha_{k-1}}\,,}\quad \alpha_k = a_k - \beta_k c_{k-1}\,,\qquad k = 2,3,\ldots,n$

Then, the solution is found by forward and back substitution:

LU Decomposition

- If LU decomposition exists, then for a tri-diagonal matrix **A**, resulting from the finite-difference discretization of the 1D Poisson equation, one can write

$$\begin{pmatrix} a_1 & c_1 & & & \\ b_2 & a_2 & c_2 & & \\ & \cdots & \cdots & \cdots & \\ & & b_{n-1} & a_{n-1} & c_{n-1} \\ & & & b_n & a_n \end{pmatrix} = \begin{pmatrix} 1 & & & & \\ \beta_2 & 1 & & & \\ & \beta_3 & \cdots & & \\ & & \cdots & \cdots & \\ & & & \beta_n & 1 \end{pmatrix} \begin{pmatrix} \alpha_1 & c_1 & & & \\ & \alpha_2 & c_2 & & \\ & & \cdots & \cdots & \\ & & & \cdots & c_{n-1} \\ & & & & \alpha_n \end{pmatrix}$$

where $\alpha_1 = a_1\,, \quad \beta_k = \dfrac{b_k}{\alpha_{k-1}}\,, \quad \boxed{\alpha_k = a_k - \beta_k c_{k-1}\,,} \quad k = 2,3,\ldots,n$

Then, the solution is found by forward and back substitution:

LU Decomposition

- If LU decomposition exists, then for a tridiagonal matrix **A**, resulting from the finite-difference discretization of the 1D Poisson equation, one can write

$$\begin{pmatrix} a_1 & c_1 & & & \\ b_2 & a_2 & c_2 & & \\ & \cdots & \cdots & \cdots & \\ & & b_{n-1} & a_{n-1} & c_{n-1} \\ & & & b_n & a_n \end{pmatrix} = \begin{pmatrix} 1 & & & & \\ \beta_2 & 1 & & & \\ & \beta_3 & \cdots & & \\ & & \cdots & \cdots & \\ & & & \beta_n & 1 \end{pmatrix} \begin{pmatrix} \alpha_1 & c_1 & & & \\ & \alpha_2 & c_2 & & \\ & & \cdots & \cdots & \\ & & & \cdots & c_{n-1} \\ & & & & \alpha_n \end{pmatrix}$$

where $\alpha_1 = a_1\,,\quad \beta_k = \dfrac{b_k}{\alpha_{k-1}}\,,\quad \boxed{\alpha_k = a_k - \beta_k c_{k-1}\,,}\quad k = 2,3,\ldots,n$

Then, the solution is found by forward and back substitution:

Backward substitution

$$\begin{pmatrix} 1 &&&& \\ \beta_2 & 1 &&& \\ & \beta_3 & \cdots && \\ && \cdots & \cdots & \\ &&& \beta_n & 1 \end{pmatrix}\begin{pmatrix} \alpha_1 & c_1 &&& \\ & \alpha_2 & c_2 && \\ && \cdots & \cdots & \\ &&& \cdots & c_{n-1} \\ &&&& \alpha_n \end{pmatrix}\begin{bmatrix} V^1 \\ V^2 \\ .. \\ V^{n-1} \\ V^n \end{bmatrix}=\begin{bmatrix} f^1 \\ f^2 \\ .. \\ f^{n-1} \\ f^n \end{bmatrix}$$

LU V=f

L*A*=f where A=U*V*

Solve for A

$$\begin{pmatrix} 1 &&&& \\ \beta_2 & 1 &&& \\ & \beta_3 & \cdots && \\ && \cdots & \cdots & \\ &&& \beta_n & 1 \end{pmatrix}\begin{bmatrix} A^1 \\ A^2 \\ .. \\ A^{n-1} \\ A^n \end{bmatrix}=\begin{bmatrix} f^1 \\ f^2 \\ .. \\ f^{n-1} \\ f^n \end{bmatrix}$$

A1=f1

A1*β2+A2=f2

A2*β3+A3=f3

Solve for UV=A

$$\begin{pmatrix} \alpha_1 & c_1 &&& \\ & \alpha_2 & c_2 && \\ && \cdots & \cdots & \\ &&& \cdots & c_{n-1} \\ &&&& \alpha_n \end{pmatrix}\begin{bmatrix} V^1 \\ V^2 \\ .. \\ V^{n-1} \\ V^n \end{bmatrix}=\begin{bmatrix} A^1 \\ A^2 \\ .. \\ A^{n-1} \\ A^n \end{bmatrix}$$

α1 V1+c1V2=A1

α2 V2+c2V3=A2

…

αn Vn=An

↑ Solve up

Numerical Solution Details

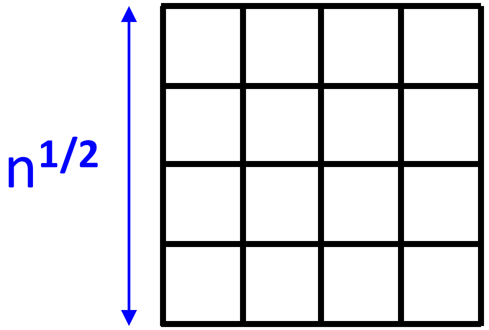
- Poisson solvers:
  - Direct
    - Gaussian Eliminatioln
    - LU decomposition
  - Iterative
    - Mesh Relaxation Methods
      - Jacobi, Gauss-Seidel, Successive over-Relaxation
    - Advanced Iterative Solvers
      - ILU, Stone’s strongly implicit method, Conjugate gradient methods and Multigrid methods

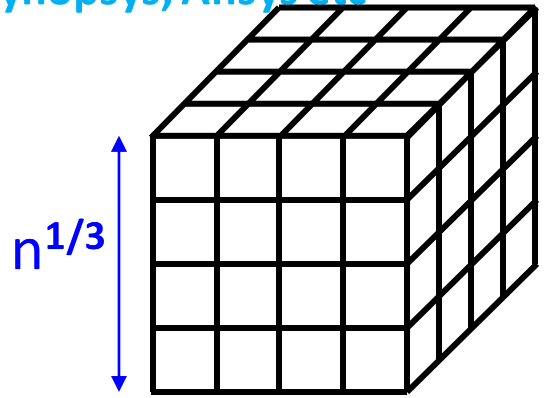
G. Speyer, D. Vasileska and S. M. Goodnick, "Efficient Poisson solver for semiconductor device modeling using the multi-grid preconditioned BICGSTAB method", *Journal of Computational Electronics*, Vol. 1, pp. 359-363 (2002).

Complexity of linear solvers:

CAD companies make their money on better solvers! Synopsys, Ansys etc

Time to solve model problem (Poisson's equation) on regular mesh

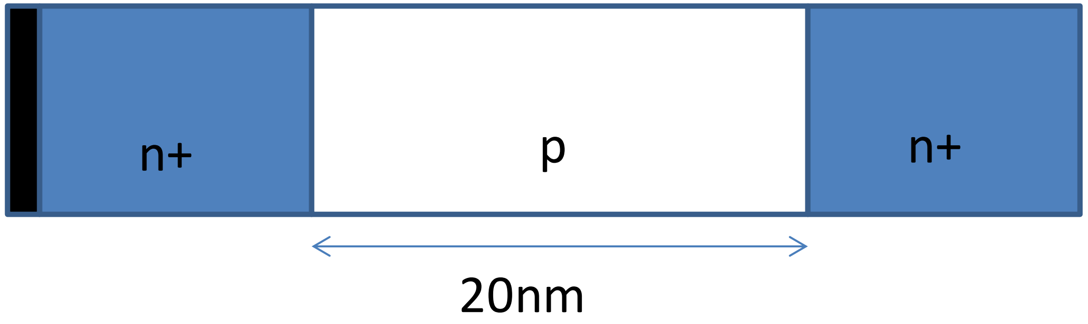




	2D	3D
Sparse Cholesky:	$O(n^{1.5})$	$O(n^2)$
CG, exact arithmetic:	$O(n^2)$	$O(n^2)$
CG, no precond:	$O(n^{1.5})$	$O(n^{1.33})$
CG, modified IC:	$O(n^{1.25})$	$O(n^{1.17})$
CG, support trees:	$O(n^{1.20})$ -> $O(n^{1+})$	$O(n^{1.75})$ -> $O(n^{1.31})$
Multigrid:	$O(n)$	$O(n)$

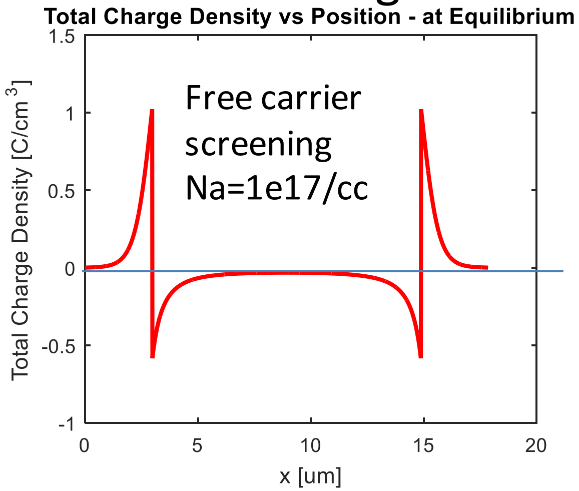
Example

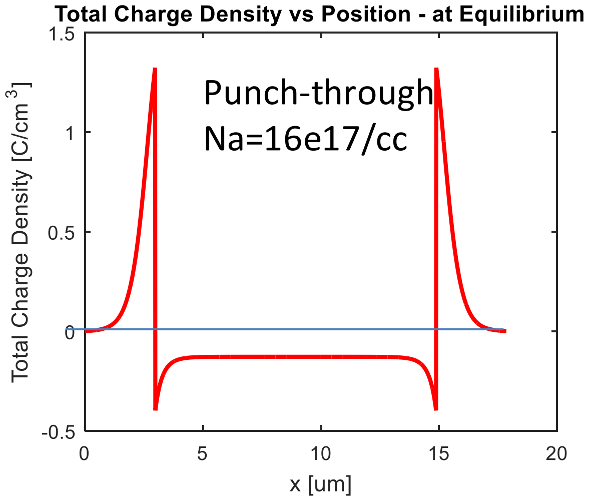
Given a n+/p/n+ device (consider it a FinFET without a gate electrode), how much is the barrier between S/D;

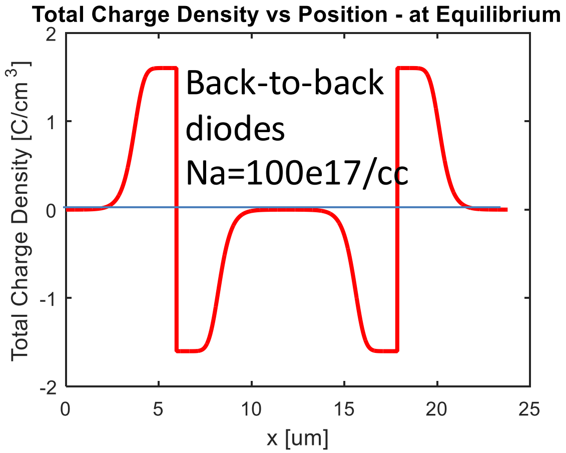


n+ doping is $10^{19}/cm^3$
p: vary from [1, 16, 100] *e17

Write down the steps including assumptions to draw charge profile and the band diagram. Show that as p<1e17 and p → intrinsic, the band diagram / does not change

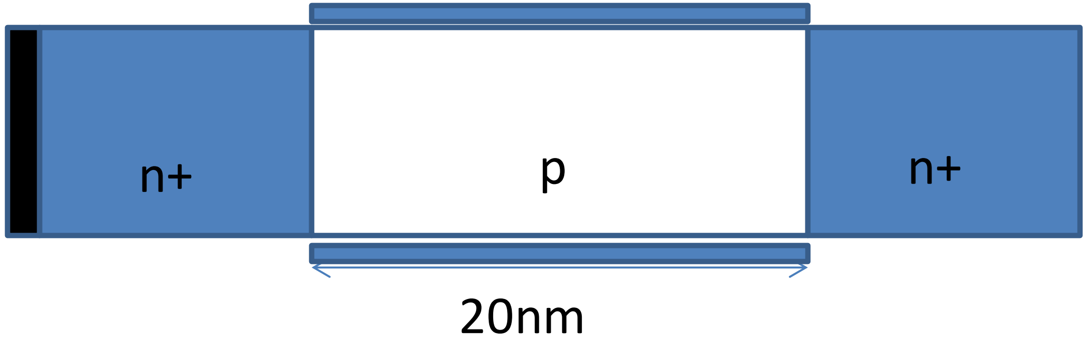




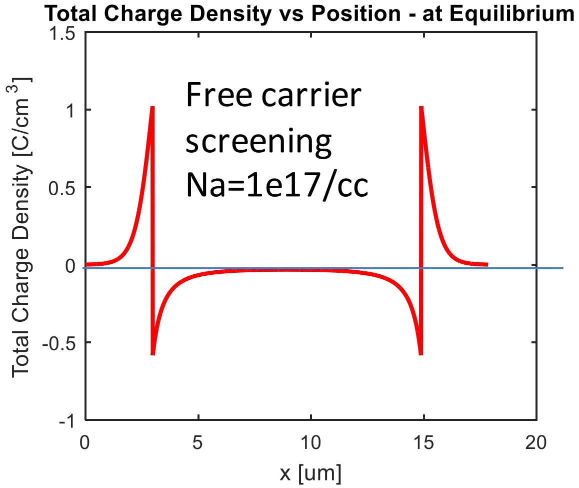


Example

Given a FinFET (consider it a n+/p/n+ device with a gate electrode),



n+ doping is 10^{19}/cm^3
p: vary from [1, 16, 100] *e17



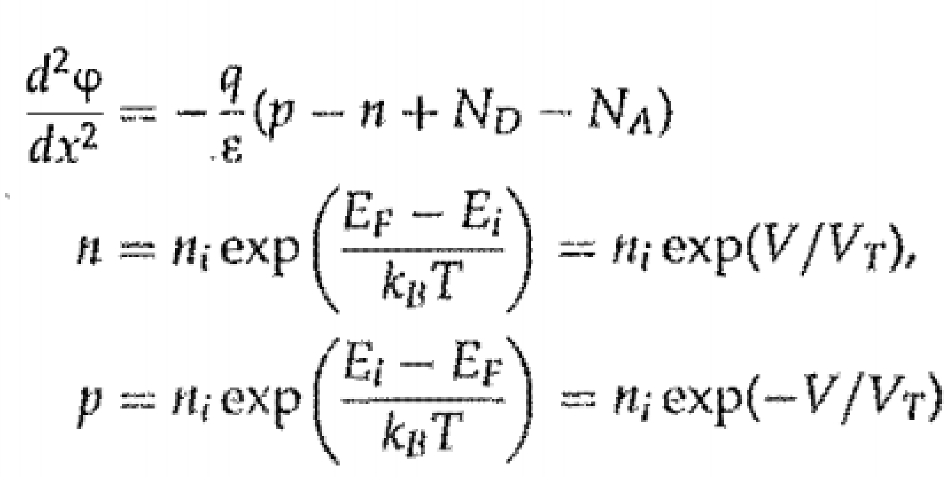
Free carrier screening Na=1e17/cc

Q: Can we increase the barrier to kill the free carrier screening?

A: Yes… by adding gate bias to push out the free carriers.

Back up

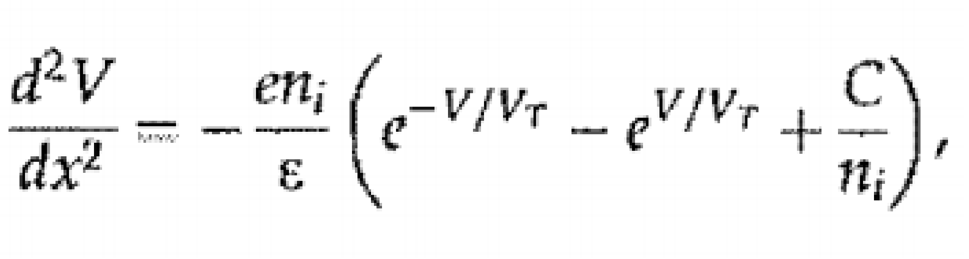
Numerical Solution



$$\frac{d^2\varphi}{dx^2} = -\frac{q}{\varepsilon}(p - n + N_D - N_A)$$

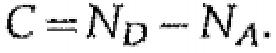
$$n = n_i \exp\left(\frac{E_F - E_i}{k_B T}\right) = n_i \exp(V/V_T),$$

$$p = n_i \exp\left(\frac{E_i - E_F}{k_B T}\right) = n_i \exp(-V/V_T)$$



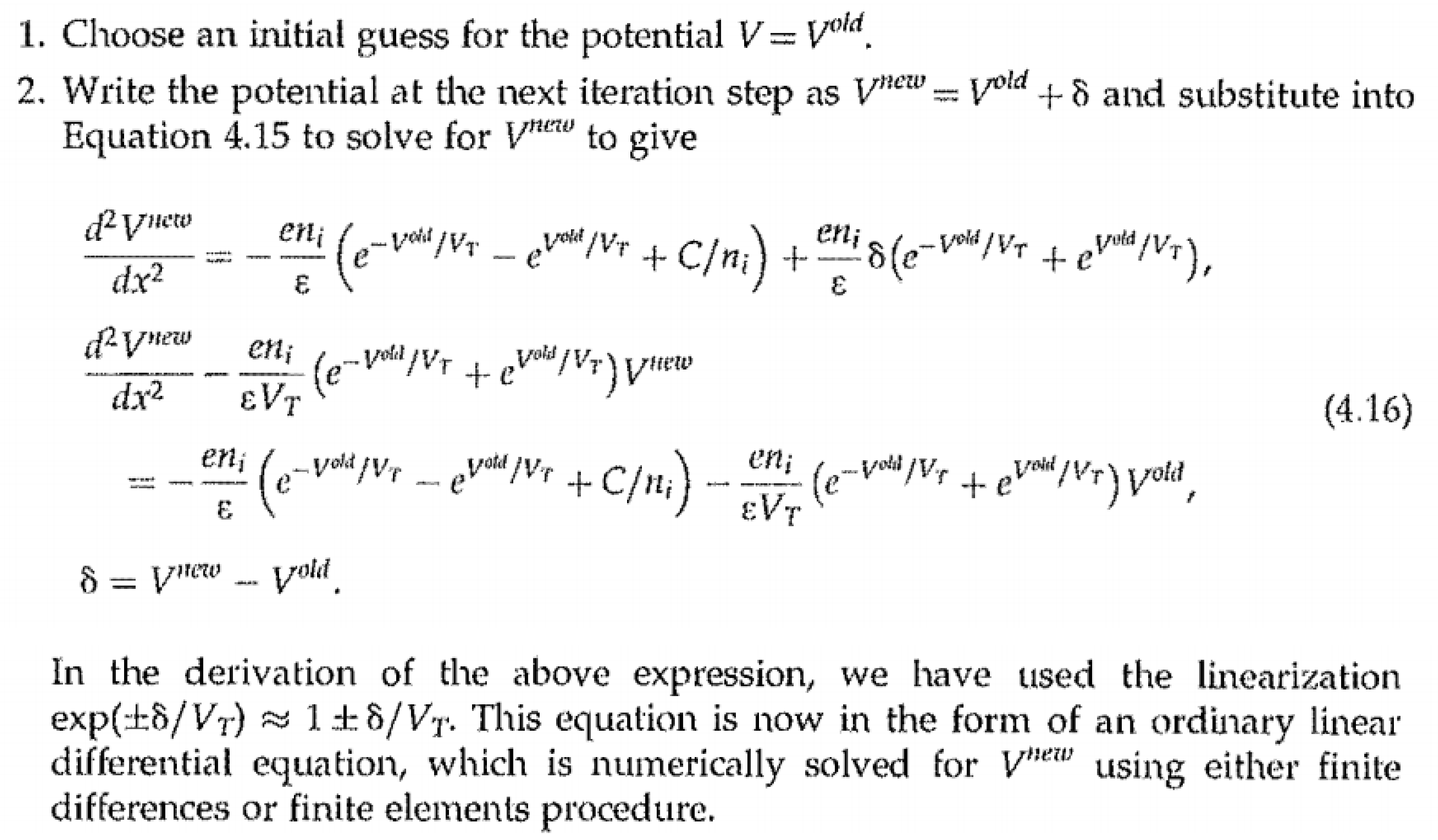
$$\frac{d^2V}{dx^2} = -\frac{en_i}{\varepsilon}\left(e^{-V/V_T} - e^{V/V_T} + \frac{C}{n_i}\right),$$

where



$$C = N_D - N_A.$$

Step by Step algorithm



1. Choose an initial guess for the potential $V = V^{old}$.
2. Write the potential at the next iteration step as $V^{new} = V^{old} + \delta$ and substitute into Equation 4.15 to solve for V^{new} to give

$$
\begin{aligned}
\frac{d^2V^{new}}{dx^2} &= -\frac{en_i}{\varepsilon}\left(e^{-V^{old}/V_T} - e^{V^{old}/V_T} + C/n_i\right) + \frac{en_i}{\varepsilon}\delta\left(e^{-V^{old}/V_T} + e^{V^{old}/V_T}\right), \\
\frac{d^2V^{new}}{dx^2} &- \frac{en_i}{\varepsilon V_T}\left(e^{-V^{old}/V_T} + e^{V^{old}/V_T}\right)V^{new} \\
&= -\frac{en_i}{\varepsilon}\left(e^{-V^{old}/V_T} - e^{V^{old}/V_T} + C/n_i\right) - \frac{en_i}{\varepsilon V_T}\left(e^{-V^{old}/V_T} + e^{V^{old}/V_T}\right)V^{old}, \\
\delta &= V^{new} - V^{old}.
\end{aligned}
\tag{4.16}
$$

In the derivation of the above expression, we have used the linearization $\exp(\pm\delta/V_T) \approx 1 \pm \delta/V_T$. This equation is now in the form of an ordinary linear differential equation, which is numerically solved for V^{new} using either finite differences or finite elements procedure.

3. Next, we normalize the variables. The potential V is normalized with the thermal voltage V_T, the carrier concentration is normalized with the intrinsic carrier concentration, and the mesh is normalized with the intrinsic Debye length.
4. Now using for simplicity finite difference discretization and a uniform mesh, we can write the above equation in a matrix form $AV^{new} = f$. Expanding the second derivative appearing in Equation 4.19 using a central difference scheme [7], the finite-difference form of the linearized 1D Poisson equation (Equation 4.16) is

$$\left.\frac{\partial^2 V}{\partial x^2}\right|_i \rightarrow \frac{V_{i+1} - 2V_i + V_{i-1}}{\Delta^2} \rightarrow a_i V_{i-1} + b_i V_i + c_i V_{i+1} = f_i, \tag{4.17}$$

where

i labels the mesh element

Δ is the mesh size

f_i is the forcing function and we have omitted the superscript *new* on the normalized potential V

By comparison with the terms on the left and right sides of Equation 4.17, we see that $a_i = c_i = 1/\Delta^2$ and $b_i = -[2/\Delta^2 + (e^{V_i^{old}} + e^{-V_i^{old}})]$. Since $|b_i| > |a_i + c_i|$, the matrix **A** after the linearization procedure becomes diagonally dominant, which, in turn, leads to stable convergence. The residual of Equation 4.17 is calculated and convergence is achieved if the norm of the residual is smaller than a preset tolerance. In practice, one might simply check that the maximum absolute update of the potential anywhere on the mesh is smaller than some preset tolerance.

$$\frac{1}{\Delta^2}\varphi_{i+1}^{n+1} - \left(\frac{2}{\Delta^2} + p_i + n_i\right)\varphi_i^{n+1} + \frac{1}{\Delta^2}\varphi_{i-1}^{n+1} = -\left[p_i - n_i + C_i\right] - \varphi_i^n\left(p_i + n_i\right)$$

At Equilibrium : $n_i = e^{\phi}$ and $p_i = e^{-\phi}$

$$a^n\varphi_{i+1}^{n+1} - b^n\varphi_i^{n+1} + c^n\varphi_{i-1}^{n+1} = f^n$$

All coefficients for (n+1)th φs can be determined from nth iteration
Knowing φ_i^n we can find φ_i^{n+1} where n implies the iteration number
So need an initial guess, then we can iterate.

Convergence criterion $\varphi_{i+1} - \varphi_i$< error for all n points.
We will learn to solve this is next lecture.