nanoHUB.org
online simulations and more

Computational Electronics

Sharfetter-Gummel Discretization Scheme
Time-Dependent Simulations

Prepared by
Dragica Vasileska
Associate Professor
Arizona State University

nanoHUB.org
online simulations and more

Sharfetter-Gummel Discretization Scheme

- The discretization of the continuity equation in conservative form requires the knowledge of the current densities

$$J_n(x) = en(x)\mu_n E + eD_n \nabla n$$
$$J_p(x) = ep(x)\mu_p E - eD_p \nabla p$$

on the mid-points of the mesh lines connecting neighboring grid nodes. Since solutions are available only on the grid nodes, interpolation schemes are needed to determine the solutions.

- There are two schemes that one can use:
  - (a) Linearized scheme: *V, n, p*, μ and *D* vary linearly between neighboring mesh points
  - (b) Scharfetter-Gummel scheme: electron and hole densities follow exponential variation between mesh points

Network for Computational Nanotechnology

nanoHUB.org
online simulations and more

(a) Linearized Scheme

- Within the linearized scheme, one has that

$$J_{i+1/2} = -en_{i+1/2}\mu_{i+1/2}\frac{V_{i+1}-V_i}{a_i} + eD_{i+1/2}\nabla n|_{i+1/2}$$

$$\frac{n_{i+1}+n_i}{2} \qquad\qquad \frac{n_{i+1}-n_i}{a_i}$$

$$J_{i+1/2} = n_{i+1}\left[-\frac{e\mu_{i+1/2}}{2}\frac{V_{i+1}-V_i}{a_i} + \frac{eD_{i+1/2}}{a_i}\right] - n_i\left[\frac{e\mu_{i+1/2}}{2}\frac{V_{i+1}-V_i}{a_i} + \frac{eD_{i+1/2}}{a_i}\right]$$

- This scheme can lead to substantial errors in regions of high electric fields and highly doped devices.

nanoHUB.org
online simulations and more

(b) Sharfetter-Gummel Scheme

- One solves the electron current density equation

$$J_{i+1/2} = -en\mu_{i+1/2}\frac{V_{i+1}-V_i}{a_i} + eD_{i+1/2}\frac{\partial n}{\partial x}$$

$$= -en\mu_{i+1/2}\frac{V_{i+1}-V_i}{a_i} + eD_{i+1/2}\frac{\partial n}{\partial V}\frac{\partial V}{\partial x}$$

for $n(V)$, subject to the boundary conditions

$$n(V_i) = n_i \quad \text{and} \quad n(V_{i+1}) = n_{i+1}$$

- The solution of this first-order differential equation leads to

$$n(V) = n_i[1-g(V)] + n_{i+1}g(V), \quad g(V) = \frac{e^{(V-V_i)/Vt}-1}{e^{(V_{i+1}-V_i)/Vt}-1}$$

$$J_{i+1/2} = \frac{eD_{i+1/2}}{a_i}\left[n_{i+1}B\left(\frac{V_{i+1}-V_i}{Vt}\right) - n_iB\left(\frac{V_i-V_{i+1}}{Vt}\right)\right]$$

$B(x) = \dfrac{x}{e^x-1}$ is the Bernouli function

nanoHUB.org
online simulations and more

Bernouli Function Implementation. Others as well …

```
C      ======================================================================
C      ======================================================================
       FUNCTION BER1(X)
C      this is more accurate version of the Bernouli function
C      ======================================================================
C      ======================================================================
       IMPLICIT REAL*4(A-H, O-Z)
       LOGICAL FLAG_SUM

       FLAG_SUM = .FALSE.
          if(x.gt.0.01)then
             Ber1 = x*exp(-x)/(1.-exp(-x))
          elseif(x.lt.0.and.abs(x).gt.0.01)then
             Ber1 = x/(exp(x)-1.)
          elseif(x.eq.0)then
             Ber1 = 1.
          else
             temp_term = 1.
             sum = temp_term
             i = 0.
             do while(.not.flag_sum)
                i = i + 1
                temp_term = temp_term*x/dfloat(i+1)
                if(sum+temp_term.eq.sum)flag_sum = .true.
                sum = sum + temp_term
             enddo
             Ber1 = 1./sum
          endif
       RETURN
       END
```


nanoHUB.org
online simulations and more

```
 FUNCTION Ec_Ei(Rnn, Rpp, Eg)
    IMPLICIT REAL*4(A-H, O-Z)
    LOGICAL I$FLAG
    REAL*4 FX(0:200)
    REAL*4 xmid, ddx
    x   = - Eg
    ddx = + 5.D0
    fx(0) = Rnn*Fermi(-x) - Rpp*Fermi(x-Eg)
    I$FLAG = .FALSE.
    k = 0
    DO WHILE(.NOT.I$FLAG)
        k = k + 1
        if(k.gt.100)print*,'Too small array!!!'
        x = x + ddx
        fx(k) = Rnn*Fermi(-x) - Rpp*Fermi(x-Eg)
        IF(fx(k-1)*fx(k).lt.0)THEN
           x1 = x - ddx
           x2 = x
           f1 = fx(k-1)
           f2 = fx(k)
           I$FLAG = .TRUE.
        ENDIF
    ENDDO
    rtbis = x1
    I$FLAG = .FALSE.
    DO WHILE(.NOT.I$FLAG)
        ddx = 0.5D0*ddx
        xmid = rtbis + ddx
        fmid = Rnn*Fermi(-xmid) - Rpp*Fermi(xmid-Eg)
        if(fmid.gt.0.D0)rtbis = xmid

        if(abs(ddx).lt.1.D-10.or.fmid.eq.0.D0)I$FLAG = .TRUE.
    ENDDO
    Ec_Ei = xmid
 RETURN
 END
```

Network for Computational Nanotechnology

nanoHUB.org

online simulations and more

```
      FUNCTION VOLTAGE(C, Rnn, Rpp, Eg, dEc)
          IMPLICIT REAL*4(A-H, O-Z)
          LOGICAL I$FLAG
          REAL*4 FX(0:200)
          REAL*4 xmid, ddx
          REAL*4 Rnn, Rpp
          x    = - 2.*Eg
          ddx = + 5.D0
          fx(0) = Rnn*Fermi(x-dEc) -
     1            Rpp*Fermi(dEc-Eg-x) + C
          I$FLAG = .FALSE.
          k = 0
          DO WHILE(.NOT.I$FLAG)
             k = k + 1
             if(k.gt.200)print*,'Too small array!!!'
             x = x + ddx
             fx(k) = Rnn*Fermi(x-dEc) -
     1               Rpp*Fermi(dEc-Eg-x) + C
             IF(fx(k-1)*fx(k).lt.0)THEN
                x1 = x - ddx
                x2 = x
                f1 = fx(k-1)
                f2 = fx(k)
                I$FLAG = .TRUE.
             ENDIF
          ENDDO
          rtbis = x1
          I$FLAG = .FALSE.
          DO WHILE(.NOT.I$FLAG)
             ddx = 0.5D0*ddx
             xmid = rtbis + ddx
             fmid = Rnn*Fermi(xmid-dEc) -
     1               Rpp*Fermi(dEc-Eg-xmid) + C
             if(fx(0).lt.0)then
               if(fmid.lt.0.D0)rtbis = xmid
             elseif(fx(0).gt.0)then
               if(fmid.gt.0.D0)rtbis = xmid
             endif
             if(abs(ddx).lt.1.D-10.or.fmid.eq.0.D0)I$FLAG = .TRUE.
          ENDDO
          Voltage = xmid
       RETURN
       END
```


nanoHUB.org

online simulations and more

```
      FUNCTION FERMI(X)
         IMPLICIT REAL*4(A-H, O-Z)
         PI = 4.D0*DATAN(1.D0)
         rnu  = x*x*x*x + 50. +
     1          33.6*x*(1.-0.68*exp(-0.17*(x+1)*(x+1)))
         zeta = 3.*sqrt(PI)/(4.*rnu**0.375)
         if(x.gt.0)then
            Fermi = 1./(exp(-x) + zeta)
         elseif(x.lt.0.and.abs(x).lt.6)then
            Fermi = exp(x)/(1. + zeta*exp(x))
         else
            Fermi = exp(x)
         endif
      RETURN
      END
```

```
      FUNCTION R_COEFF(XO, N, XT)
      IMPLICIT REAL*4(A-H, O-Z)
      LOGICAL FLAG$CONV
      flag$conv = .false.
      r = 3.
      do while(.not.flag$conv)
          term1 = 1./(r-1.D0)
          term2 = r**float(N)-1.D0
          rnum  = xo*term1*term2 - xt
          term3 = float(N)*r**float(N-1)
          denom = xo*(term1*term3-term1*term1*term2)
          r_new = r - rnum/denom
          error = abs(r-r_new)/abs(r_new)
          if(error.le.1.e-10)then
             flag$conv = .true.
          else
             r = r_new
          endif
      enddo
      r_coeff = r
      RETURN
      END
```

Network for Computational Nanotechnology

nanoHUB.org
online simulations and more

Validity of the Drift-Diffusion Model

(a) Approximations made in its derivation

- Temporal variations occur in a time-scale much longer than the momentum relaxation time.
- The drift component of the kinetic energy was neglected, thus removing all thermal effects.
- Thermoelectric effects associated with the temperature gradients in the device are neglected, i.e.

$$\mathbf{J}_n(x) = en(x)\mu_n\mathbf{E} + eD_n\nabla n$$
$$\mathbf{J}_p(x) = ep(x)\mu_p\mathbf{E} - eD_p\nabla p$$

- The spatial variation of the external forces is neglected, which implies slowly varying fields.
- Parabolic energy band model was assumed, i.e. degenerate materials can not be treated properly.

nanoHUB.org
online simulations and more

(b) Extension of the capabilities of the DD model

- Introduce field-dependent mobility μ(**E**) and diffusion coefficient D(**E**) to empirically extend the range of validity of the DD Model.
- An extension to the model, to take into account the overshoot effect, has been accomplished in 1D by adding an extra term that depends on the spatial derivative of the electric field

$$J_n(x) = en(x)\left[\mu_n(E) + \mu_n(E)L(E)\frac{\partial E}{\partial x}\right] + eD_n(E)\frac{\partial n}{\partial x}$$

1. K.K. Thornber, IEEE Electron Device Lett., Vol. 3 p. 69, 1982.
2. E.C. Kan, U. Ravaioli, and T. Kerkhoven, Solid-State Electron., Vol. 34, 995 (1991).

Network for Computational Nanotechnology

nanoHUB.org
online simulations and more

Time-Dependent Simulations

- The time-dependent form of the drift-diffusion equations can be used both for
  - steady-state, and
  - transient calculations.
- Steady-state analysis is accomplished by starting from an initial guess, and letting the numerical system evolve until a stationary solution is reached, within set tolerance limits.

nanoHUB.org
online simulations and more

- This approach is seldom used in practice, since now robust steady-state simulators are widely available.
  - It is nonetheless an appealing technique for beginners since a relatively small effort is necessary for simple applications and elementary discretization approaches.
  - If an explicit scheme is selected, no matrix solutions are necessary, but it is normally the case that stability is possible only for extremely small time-steps.
- The simulation of transients requires the knowledge of a physically meaningful initial condition.
- The same time-dependent numerical approaches used for steady-state simulation are suitable, but there must be more care for the boundary conditions, because of the presence of **displacement current during transients**.

Network for Computational Nanotechnology

nanoHUB.org
online simulations and more

Displacement Current

- In a true transient regime, the presence of displacement currents manifests itself as a potential variation at the contacts, superimposed to the bias, which depends on the external circuit in communication with the contacts.
  - Neglect of the displacement current in a transient is equivalent to the application of bias voltages using ideal voltage generators, with zero internal impedance.
  - In this arrangement, one observes the shortest possible switching time attainable with the structure considered.
  - Hence, a simulation neglecting displacement current effects may be useful to assess the ultimate speed limits of a device structure.

nanoHUB.org
online simulations and more

Displacement Current, …

- When a realistic situation is considered, it is necessary to include a displacement term in the current equations. It is particularly simple to deal with a 1-D situation. Consider a 1-D device with length *W* and a cross-sectional area *A*. The total current flowing in the device is

$$I_D(t) = I_n(x,t) + cA\frac{\partial E(x,t)}{\partial t} \qquad (1)$$

- The displacement term makes the total current constant at each position *x*. This property can be exploited to perform an integration along the device

$$I_D(t) = \frac{1}{W}\int_0^W I_n(x,t)dx + \frac{cA}{W}\frac{\partial V^*}{\partial t} \qquad (2)$$

- The 1-D device, therefore, can be studied as the parallel of a current generator and of the cold capacitance which is in parallel with the (linear) load circuit.
- At every time step, Eq. (1) has to be updated, since it depends on the charge stored by the capacitors.

nanoHUB.org
online simulations and more

Example: Gunn Diode

- Consider a simple Gunn diode in parallel with an *RLC* resonant load containing the bias source.

$$I(t) = C_o \frac{\partial V^*(t)}{\partial t} + I_o(t) \tag{3}$$

where $I_0(t)$ is the particle current given by the first term on the right hand side of Eq. (1), calculated at the given time step with drift-diffusion. It is also

$$I(t) = -\frac{V^*(t)}{R} - \int \frac{V^*(t) - V_b}{L} dt \tag{4}$$

- Upon time differencing this last equation, with the use of finite differences, we obtain

$$V^*(t+\Delta t) = V^*(t) + \left[I(t) - I_o(t)\right]\frac{\Delta t}{C_o}$$

$$I(t+\Delta t) = I(t) - \frac{V^*(t+\Delta t) - V^*(t)}{R} - \left[V^*(t) - V_b\right]\frac{\Delta t}{L} \tag{5}$$

nanoHUB.org
online simulations and more

Some General Comments

- A robust approach for transient simulation should be based on the same numerical apparatus established for purely steady-state models.
- It is usually preferred to use fully implicit schemes, which require a matrix solution at each iteration, because the choice of the time-step is more likely to be limited by the physical time constants of the problem rather than by stability of the numerical scheme.
- Typical calculated values for the time step are not too far from typical values of the dielectric relaxation time in practical semiconductor structures.

Unipolar Devices Problem Specification

- When dealing with unipolar devices, as often used in many microwave applications, time-dependent drift-diffusion models that can be solved with straightforward finite difference techniques are suitable for small student projects.

- If we can neglect the generation-recombination effects, the 1-D unipolar drift-diffusion model is reduced to the following system of equations

$$\frac{\delta n}{\delta t} = -\frac{d}{dx}\left[n v_d(E)\right] + \frac{d}{dx}\left[D(E)\frac{d}{dx}n\right]$$

$$\frac{d^2V}{dx^2} = \frac{q(n - N_D)}{\varepsilon}$$

nanoHUB.org
online simulations and more

A basic simple algorithm could consist of the following steps:

1. Guess the carrier distribution $n(x)$.
2. Solve Poisson's equation to obtain the field distribution.
3. Compute one iteration of the discretized continuity equation with time step δt. $v(E)$ and $D(E)$ are updated according to the local field value.
4. Check for convergence. If convergence is obtained, stop. Otherwise, go back to step (2) updating the charge distribution.

Network for Computational Nanotechnology