```fortran
if (seec == 1) then     !************* Vaughan model **************!

  Emax = sey%Emax     ! Emax(delta=max,theta=0) in eV
  deltamax = sey%deltamax      ! Maximum secondary electron yield (at Emax) for normal incidence (theta=0)
  ReRr = sey%ReRr
  delta_b = sey%delta_b
  E_0 = sey%E_0
  E_0p = sey%E_0p

  if (penf1 <= E_0p) then

    delta = delta_b

  elseif (penf1 > E_0p .AND. penf1 < E_0) then

    delta = -delta_b*(penf1 - E_0)

  else

    ! Unpack user inputs regarding particle-wall interaction and SEE
    kse = sey%kse           ! Roughness factor for energy, =0 for rough, =1 for dull, =2 for smooth and anything in between
    ks = sey%ks             ! Roughness factor for angle, =0 for rough, =1 for dull, =2 for smooth and anything in between

    ! Determine secondary electron yields (Vaughan's theory (1993))
    Emax = Emax * (1 + kse*theta*theta/(2*pi))
    deltamax = deltamax * (1 + ks*theta*theta/(2*pi))

    xi = real((penf1 - E_0))/(Emax - E_0)

    if (xi <= 3.6) then

      if (xi <= 1) then
        ke = 0.56
      elseif (xi > 1 .OR. xi <= 3.6) then
        ke = 0.25
      endif

        delta = deltamax * ( (xi * exp(1-xi))**ke )

    elseif (xi > 3.6) then

        delta = deltamax * 1.125 / (xi**0.35)

    endif

  endif
!DEBUG
  if (allocated(SEYvsE)) then
    call add23Darray(SEYvsE,(/0,0,1/))
    k = size(SEYvsE,3)
    SEYvsE(4,1,k) = penf1
    SEYvsE(4,2,k) = delta
  else
    allocate (SEYvsE(4,2,1))
    SEYvsE(4,1,1) = penf1
    SEYvsE(4,2,1) = delta
  endif

  if (delta > 10) then
    write(*,*) 'high delta'
```

```fortran
      call exit
    endif

    !if (first) then
    !  s = random_Poisson(real(delta,4),.TRUE.)        ! Number of secondary electrons produced
    !  first = .FALSE.
    !else
    !  s = random_Poisson(real(delta,4),.FALSE.)
    !endif

    if (isNaN(real(delta,8))) then
      write(*,*) 'theta = ',theta
      write(*,*) 'penf1 = ',penf1
      write(*,*) 'E_0 = ',E_0
      write(*,*) 'E_0p = ',E_0p
      write(*,*) 'delta = ',delta
      write(*,*) 'E_0p = ',E_0p
      write(*,*) 'E_0 = ',E_0
    endif

    if (atype == 9) then
      read(17,'(F17.15,1X,I1)') delta0, s
      if (delta0-delta>1.5e-3) then
        write(*,*) 'discrepancy in delta'
        write(*,*) 'penf1 = ',penf1
        write(*,*) 'theta = ',theta
        write(*,*) 'delta0 = ',delta0
        write(*,*) 'delta = ',delta
        write(*,*) 'diff = ',delta0-delta
        call exit
      endif
    else
      s = poisdev(real(delta,8),atype)
    endif

    if (s == 1) then

      allocate (pvf(1,3)) ! Only one electron comes off the wall

      if (ReRr == 1) then
        ! Determine Re and Rr from de Lara's (2006) empiricial fit
        a_lara = sey%a_lara
        z_lara = sey%z_lara
        Eb = 300 + 175*z_lara
        Rr = a_lara * (1 - (3e-5)*penf1) * (penf1**0.56) * exp(-((penf1/Eb)**0.70))

        Ee1 = 50./sqrt(z_lara)
        Ee2 = 0.25 * z_lara * z_lara
        Re = 0.93/(1+penf1/Ee1) + 0.07/(1+penf1/Ee2)

        ! Take into account angle dependence of Re and Rr
        C1 = 0.89 * Rr / (Re + Rr)
        Rr = (Rr**cos(theta)) * (C1**(1-cos(theta)))
        Re = (Re**cos(theta)) * ((0.89-C1)**(1-cos(theta)))

      elseif (ReRr == 2) then     ! Empirical fit from LHC
        a0 = 20.69989
        a1 = -7.07605
        a2 = 0.483547
        a3 = 0
        e0 = 56.914686
        f = exp(a0 + a1*log(penf1+e0) + a2*((log(penf1+e0))**2) + a3*((log(penf1+e0))**3))

```

```
      ! Based on publications saying that 7% and 3% of secondaries are elastically and
      inelastically reflected respectively
      Re = f*0.3
      Rr = f-Re
    endif

!open(unit=63,file='ReRr.txt',status='unknown',position='append')
!write(63,*) Re,Rr
!close(unit=63)

    ! Determine probability of events
    Pe = Re             ! Prob that electron is elastically reflected
    Pr = Rr             ! Prob that electron is inelastically reflected
    Pe = 0.03
    Pr = 0.07

    ! Now generate a random number uniformly distributed between 0 and 1
    if (atype == 8) then
      read(21,*) r
    elseif (atype == 9) then
      read(18,*) r
    else
      r = taus88()
    endif

    if (r < Pe) then

      ! Reflect electron elastically
      pvf = reflect_electron(pcol,penf1*(-e),pvf1,1)
!DEBUG
      if (allocated(SEYvsE)) then
        call add23Darray(SEYvsE,(/0,0,1/))
        k = size(SEYvsE,3)
        SEYvsE(1,1,k) = penf1
        SEYvsE(1,2,k) = delta
      else
        allocate (SEYvsE(4,2,1))
        SEYvsE(1,1,1) = penf1
        SEYvsE(1,2,1) = delta
      endif

    elseif (r >= Pe .AND. r < (Pe+Pr)) then

      ! Electron is reflected inelastically
      !pvf = reflect_electron(pcol,penf1*(-e),pvf1,2,atype)

      allocate (thetas(1))
      allocate (phis(1))
      allocate (penf(1))
      allocate (vs(1))

      penf = taus88() * penf1
      vs = sqrt(2*penf*(-eme))

      ! Determine which is the normal component of the outgoing velocity
      if (pcol(1)==1 .OR. pcol(3)==1) then
        i = 1               ! Normal component
        j = 2               ! Parallel component(not z)
      elseif (pcol(2)==1 .OR. pcol(4)==1) then
        i = 2               ! Normal component
        j = 1               ! Parallel component (not z)
      endif

```

```
      rt = taus88()
      phis = rt * 2 * pi        ! Calculate emission azimuthal angle of secondaries

      rt = taus88()
      thetas = asin(2*rt - 1)   ! Calculate emission angle of secondaries with respect to the normal

      pvf(1,i) = vs(1) * cos(thetas(1))   ! Normal component of secondary velocity

      pvf(1,j) = sin(phis(1)) * vs(1) * sin(thetas(1))
      pvf(1,3) = cos(phis(1)) * vs(1) * sin(thetas(1))

      if (pcol(1) == 1 .OR. pcol(2) == 1) then
        pvf(1,i) = - abs(pvf(1,i))   ! Normal velocity pointing into the inside of the waveguide
      elseif (pcol(3) == 1 .OR. pcol(4) == 1) then
        pvf(1,i) = abs(pvf(1,i))   ! Normal velocity pointing into the inside of the waveguide
      endif

      deallocate (thetas)
      deallocate (phis)
      deallocate (penf)
      deallocate (vs)
!DEBUG
      if (allocated(SEYvsE)) then
        call add23Darray(SEYvsE,(/0,0,1/))
        k = size(SEYvsE,3)
        SEYvsE(2,1,k) = penf1
        SEYvsE(2,2,k) = delta
      else
        allocate (SEYvsE(4,2,1))
        SEYvsE(2,1,1) = penf1
        SEYvsE(2,2,1) = delta
      endif

    else
!DEBUG
      if (allocated(SEYvsE)) then
        call add23Darray(SEYvsE,(/0,0,1/))
        k = size(SEYvsE,3)
        SEYvsE(3,1,k) = penf1
        SEYvsE(3,2,k) = delta
      else
        allocate (SEYvsE(4,2,1))
        SEYvsE(3,1,1) = penf1
        SEYvsE(3,2,1) = delta
      endif

      allocate (yn(1))
      allocate (thetas(1))
      allocate (phis(1))
      allocate (penf(1))
      allocate (vs(1))

      ! Initialize variables
      pvf = 0.       ! velocity of 1 secondary electron

      ! Scalar parameters needed to obtain velocity components
      call gratio(real(p_n*s,8), real(penf1/Eom,8), ans, qans, 0)
      if (atype == 8) then
        read(21,*) rt
        ans = ans*rt
      elseif (atype == 9) then
```

```fortran
          read(20,*) rt
          ans = ans*rt
        else
          ans = ans*taus88()
        endif
        call gaminv(real(p_n*s,8), yn(1), real(0,8), ans, 1-ans, ierr)

        if (ierr < 0) then
          write(*,*) 'error in calculating inverse incomplete gamma function. Terminating.'
          call exit
        endif

        yn = sqrt( yn ) ! magnitude of normalized velocity of secondary electrons

        penf = Eom * yn * yn
        vs = sqrt(2*penf*(-eme))

        ! Determine which is the normal component of the outgoing velocity
        if (pcol(1)==1 .OR. pcol(3)==1) then
          i = 1              ! Normal component
          j = 2              ! Parallel component(not z)
        elseif (pcol(2)==1 .OR. pcol(4)==1) then
          i = 2              ! Normal component
          j = 1              ! Parallel component (not z)
        endif

        if (atype == 8) then
          read(21,*) rt
        elseif (atype == 9) then
          read(20,*) rt
        else
          rt = taus88()
        endif
        phis = rt * 2 * pi        ! Calculate emission azimuthal angle of secondaries

        if (atype == 8) then
          read(21,*) rt
        elseif (atype == 9) then
          read(20,*) rt
        else
          rt = taus88()
        endif
        thetas = asin(2*rt - 1)   ! Calculate emission angle of secondaries with respect to the normal

        pvf(1,i) = vs(1) * cos(thetas(1))   ! Normal component of secondary velocity

        pvf(1,j) = sin(phis(1)) * vs(1) * sin(thetas(1))
        pvf(1,3) = cos(phis(1)) * vs(1) * sin(thetas(1))

        if (pcol(1) == 1 .OR. pcol(2) == 1) then
          pvf(1,i) = - abs(pvf(1,i))   ! Normal velocity pointing into the inside of the waveguide
        elseif (pcol(3) == 1 .OR. pcol(4) == 1) then
          pvf(1,i) = abs(pvf(1,i))   ! Normal velocity pointing into the inside of the waveguide
        endif

        deallocate (yn)
        deallocate (thetas)
        deallocate (phis)
        deallocate (penf)
        deallocate (vs)

```

```
    endif
  elseif (s > 1) then
!DEBUG
      if (allocated(SEYvsE)) then
        call add23Darray(SEYvsE,(/0,0,1/))
        k = size(SEYvsE,3)
        SEYvsE(3,1,k) = penf1
        SEYvsE(3,2,k) = delta
      else
        allocate(SEYvsE(4,2,1))
        SEYvsE(3,1,1) = penf1
        SEYvsE(3,2,1) = delta
      endif

    allocate (pvf(s,3))
    allocate (yn(s))
    allocate (thetas(s))
    allocate (phis(s))
    allocate (penf(s))
    allocate (vs(s))

    ! Initialize variables
    pvf = 0.       ! m x 3 array of 3D velocities of m secondary electrons
    yn = 0.        ! magnitude of velocity of secondary electrons
    thetas = 0.     ! Emission angle of secondaries with respect to the normal
    phis = 0.       ! Azimuthal emission angle of secondaries
    penf = 0.
    vs = 0.

    call gratio(real(p_n*s,8), real(penf1/Eom,8), ans, qans, 0)
    if (atype == 8) then
      read(21,*) rt
    elseif (atype == 9) then
      read(20,*) rt
    else
      rt = taus88()
    endif
    ans = ans*rt
    call gaminv(real(p_n*s,8), y, real(0,8), ans, 1-ans, ierr)

    Y = sqrt( Y )    ! v^tilde

    !! Scalar parameters needed to obtain velocity components
    sint = 1.
    do k = 1,s-1
      lnbeta = betaln(real(p_n*(s-k),4),real(p_n,4))

      if (atype == 8) then
        read(21,*) rt
      elseif (atype == 9) then
        read(20,*) rt
      else
        rt = taus88()
      endif
      inbeta = betain( real(rt,8), real(p_n*(s-k),8), real(p_n,8), lnbeta, ifault )
      if (ifault /= 0) then
        write(*,*) 'error calculating incomplete beta function. Terminating.'
        call exit
      endif

      alpha = xinbta( real(p_n*(s-k),8), real(p_n,8), lnbeta, inbeta, ifault )
      if (ifault /= 0) then
        write(*,*) 'error calculating inverse incomplete beta function. Terminating.'
```

```
        call exit
      endif

      alpha = asin( sqrt( alpha ) )     ! Alpha angles used to calculate the magnitude of velocity

      yn(k) = Y * sint * cos(alpha)     ! magnitude of outgoing velocities

      sint = sint * sin(alpha)          ! Spherical coordinates factor
    enddo
    yn(s) = Y * sint

    ! Determine which is the normal component of the outgoing velocity
    if (pcol(1)==1 .OR. pcol(3)==1) then
      i = 1               ! Normal component
      j = 2               ! Parallel component(not z)
    elseif (pcol(2)==1 .OR. pcol(4)==1) then
      i = 2               ! Normal component
      j = 1               ! Parallel component (not z)
    endif

    do k = 1,s
      penf(k) = Eom * (yn(k))**2
      vs(k) = sqrt(2*penf(k)*(-eme))

      if (atype == 8) then
        read(21,*) rt
      elseif (atype == 9) then
        read(20,*) rt
      else
        rt = taus88()
      endif
      phis(k) = rt * 2 * pi        ! Calculate emission azimuthal angle of secondaries

      if (atype == 8) then
        read(21,*) rt
      elseif (atype == 9) then
        read(20,*) rt
      else
        rt = taus88()
      endif
      thetas(k) = asin(2*rt - 1)   ! Calculate emission angle of secondaries with respect to the normal

      pvf(k,i) = vs(k) * cos(thetas(k))   ! Normal component of secondary velocity
      pvf(k,j) = sin(phis(k)) * vs(k) * sin(thetas(k))
      pvf(k,3) = cos(phis(k)) * vs(k) * sin(thetas(k))
    enddo

    if (pcol(1) == 1 .OR. pcol(2) == 1) then
      do k = 1,s
        pvf(k,i) = - abs(pvf(k,i))   ! Normal velocity pointing into the inside of the waveguide
      enddo
    elseif (pcol(3) == 1 .OR. pcol(4) == 1) then
      do k = 1,s
        pvf(k,i) = abs(pvf(k,i))   ! Normal velocity pointing into the inside of the waveguide
      enddo
    endif

    deallocate (yn)
    deallocate (thetas)
    deallocate (phis)
```

```
    deallocate (penf)
    deallocate (vs)

  elseif (s == 0) then
    allocate (pvf(1,3))
    pvf = 0.
  endif
```