

Recent Developments in Electron Cloud Effects

M. Blaskiewicz

Spent June 3 through 7 at LBNL working with M. Furman.

The bulk of the time was spent optimizing Miguel's POSINST code.

We inserted timing calls and found that the majority of the time was spent calculating the energy of electrons coming off the wall.

The original algorithm for choosing the secondary energy went as follows

- 1) For a given incident energy E_0 , calculate the secondary energy spectrum $P(E)$
- 2) Find the maximum value of $P(E)$, call it P_{max}
- 3) Create a random deviate uniformly distributed on $[0, E_0]$, call it E_1
- 4) Evaluate $P(E_1)/P_{max}$
- 5) Create a random deviate uniformly distributed on $[0, 1]$, call it x_1
- 6) If $x_1 \leq P(E_1)/P_{max}$ the secondary energy is E_1 , otherwise go to 3)

For our PSR test case more than 99% of the time was spent in this computation.

The improvement followed from noticing that the secondary energy distribution was of the form

$$P(E) = \sum_{n=1}^N a_n P_n(E) \quad (1)$$

where $a_n > 0$ and

$$\sum_{n=1}^N a_n = 1$$

Also, efficient random number generators exist for each of the distributions $P_n(E)$.

In our case $N = 3$: $P_1(E)$ was a Gaussian, $P_2(E)$ was a truncated power law and $P_3(E)$ was an incomplete gamma function.

The breakthrough came by noticing that the distribution given by equation (1) can be evaluated nearly as quickly as evaluating only one of the efficient generators

Define the N partial sums

$$A_j = \sum_{n=1}^j a_n$$

Using these define the functions $I_j(x) = H(A_j - x) - H(A_{j-1} - x)$ where $H(x) = 1$ for $x > 0$ and $H(x) = 0$ for $x \leq 0$.

Let the random variables X_n be chosen according to $P_n(E)$

Let X_0 be a uniform deviate on $[0, 1]$

Then, a random deviate distributed according to equation (1) is given by

$$X = \sum_{j=1}^N I_j(X_0) X_j \quad (2)$$

A simple way to prove equation (2) is to verify that $\langle \exp(-\lambda X) \rangle$ is the Laplace transform of equation (1).

Here's some Fortran

```
x0 = ran0()  
asum=0  
do k=1,N  
    asum = asum + a(k)  
    if(x0 .le. asum )then  
        x = rank()  
        go to 25  
    endif  
enddo  
25 return
```

This improved the speed of our test case by a factor of 200.

It seems too simple to be new, any references would be appreciated.

The other main achievement was fixing a bug in my code NCSEC
This involved an error in calculating the secondary emission flux
Let the total secondary emission yield for a given input energy be

$$SEY = \frac{Q_{out}}{Q_{in}} \quad (3)$$

where Q_{in} is the total charge hitting the surface and Q_{out} is the total charge leaving the surface

Let P_r be the probability of reflection.

Originally my code went as follows

```
x0 = ran0()
if(x0 .le. P_r )then
    Qout = Qin
else
    Qout = (SEY - Pr)Qin
endif
```

When averaged over x_0 this gives

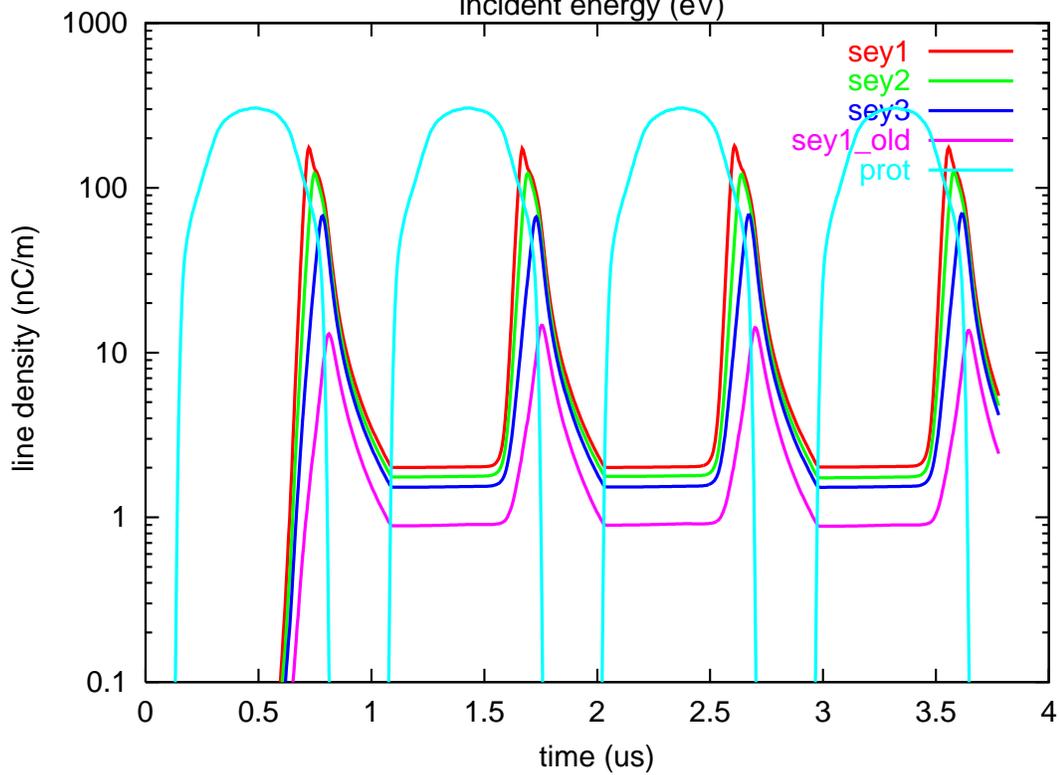
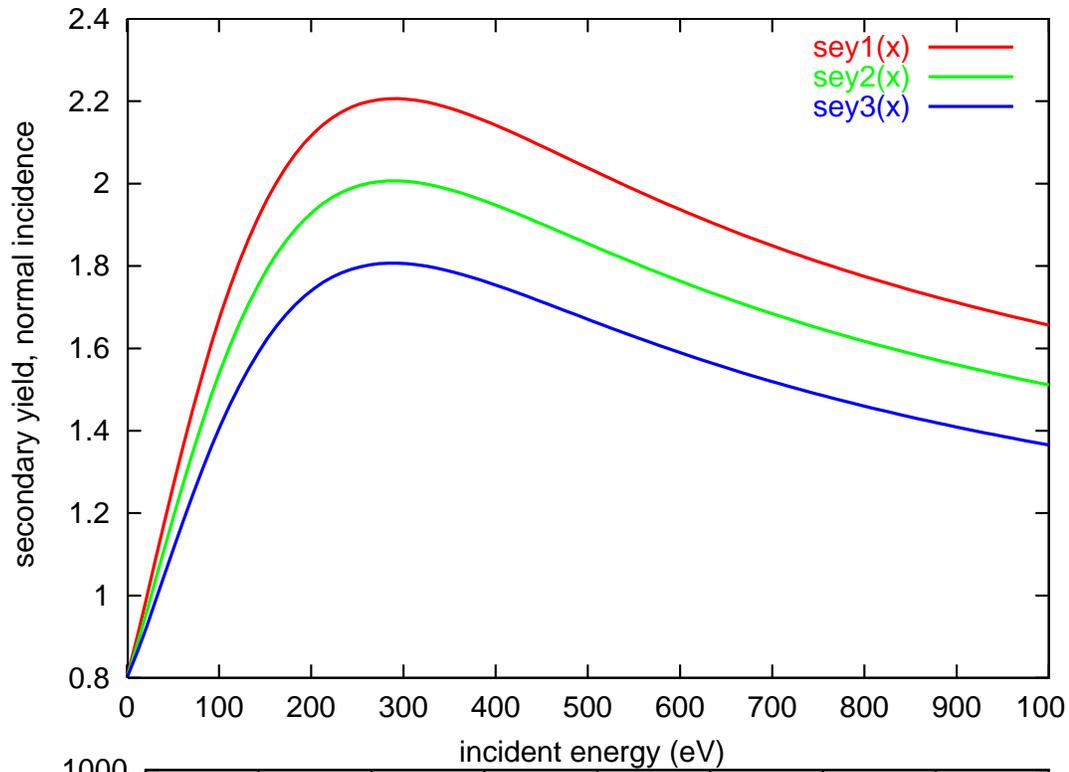
$$\langle Q_{out}/Q_{in} \rangle = (SEY - P_r)(1 - P_r) + P_r,$$

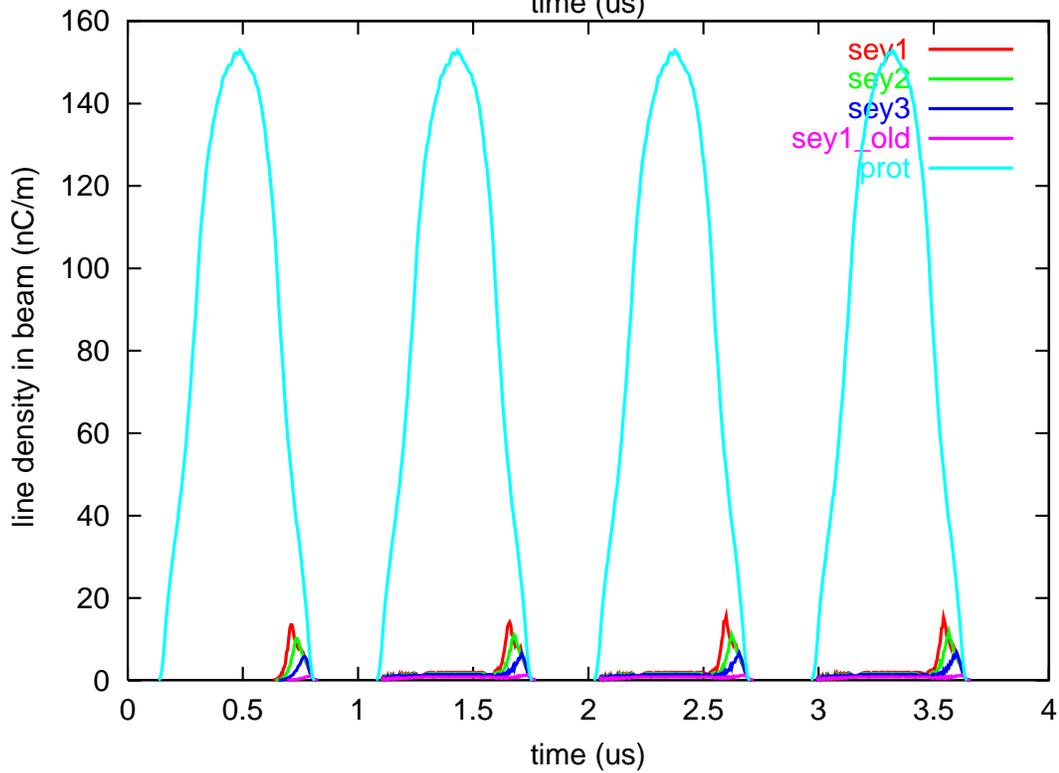
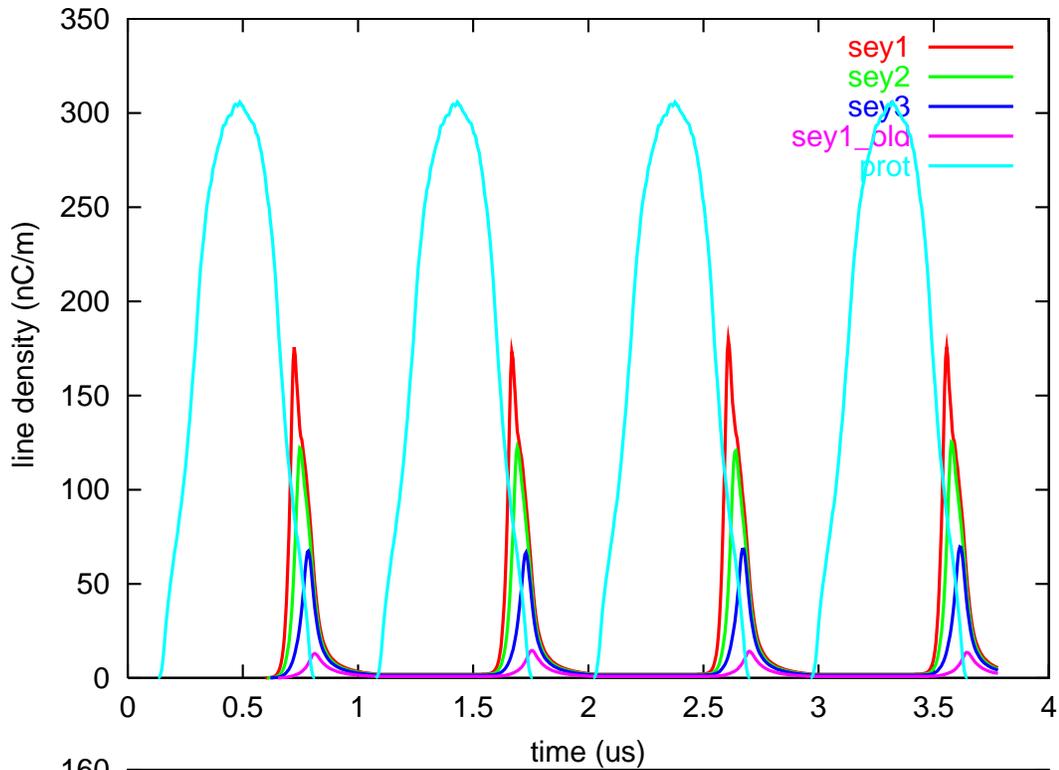
which is incorrect. The correct average is given by

```
x0 = ran0()
if(x0 .le. P_r )then
    Qout = Qin
else
    Qout = (SEY - Pr)Qin/(1 - Pr)
endif
```

This has a significant impact on electron cloud estimates.

Consider 2×10^{14} protons with losses that create 2×10^8 electrons per meter per turn.





There are still differences between POSINST and NCSEC.
 Hopefully, they are minor
 Comparison between the codes and estimates for SNS continue