Physics of Free Electron Lasers - Homework 1- Particle Loading

For simulations in beam physics, it is difficult to represent each and every particle in the beam with a numerical counterpart. For example, 1nC of charge corresponds to 6 billion particles and if each numerical particle requires only four bytes of memory, the net requirement exceeds 10Gbytes. In view of this we attempt to represent the beam behavior with that of a collection of "macro-particles", numbering from the 100-100,000 (depending on the problem). However, this introduces a problem of numerical "graininess" or noise. To see this devise a routine that loads N=128 macroparticles according to

$\gamma=\gamma_m+\Delta\gamma\,(2\,R_1-1)$
$\psi=(2\,R_2-1)\,\pi$

where $R_1$and R_2 are random deviates, for example "system ran". (Use any software you please).

(1) Make a scatter plot of the loading, for γ_m =10 and $\Delta\gamma$ =0.1. With this loading, compute the averages **(2)** $<\sin\psi>$ **(3)** $<\cos\psi>$ **(4)** $<\gamma-\gamma_m>$
(5) $(<\gamma^2>-<\gamma>^2)^{1/2}/\Delta\gamma$

Next, repeat this exercise **(6)-(10),** but using Hammersely sequences. In Fortran, these sequences can be obtained by successive calls to ham(1), ham(2), where

```
      real function ham(i)
      dimension j(4),k(4)
      data j/2,3,5,7/
      data k/0,0,0,0/
      ham=0.
      b=1.
      k(i)=k(i)+1
      m=k(i)
 10   n=m/j(i)
      b=b/j(i)
      ham=ham+(m-j(i)*n)*b
      m=n
      if(m.gt.0) goto 10
      end
```

Feel free to take a look at the loading program, **loading.for** at the class web-site. As you will see by comparing the scatter plots, Hammersley deviates, among other pseudorandom deviates, are generally "quieter" than typical linear congruential generators (such as most system ran functions). Additional symmetrization is employed in **loading.for** and such a loading routine provides a useful tool for klystron, FEL and any other longitudinal phase-space simulation.

Homework 1- Solutions - Physics of Free Electron Lasers

Particle Loading

Here are the results of loading

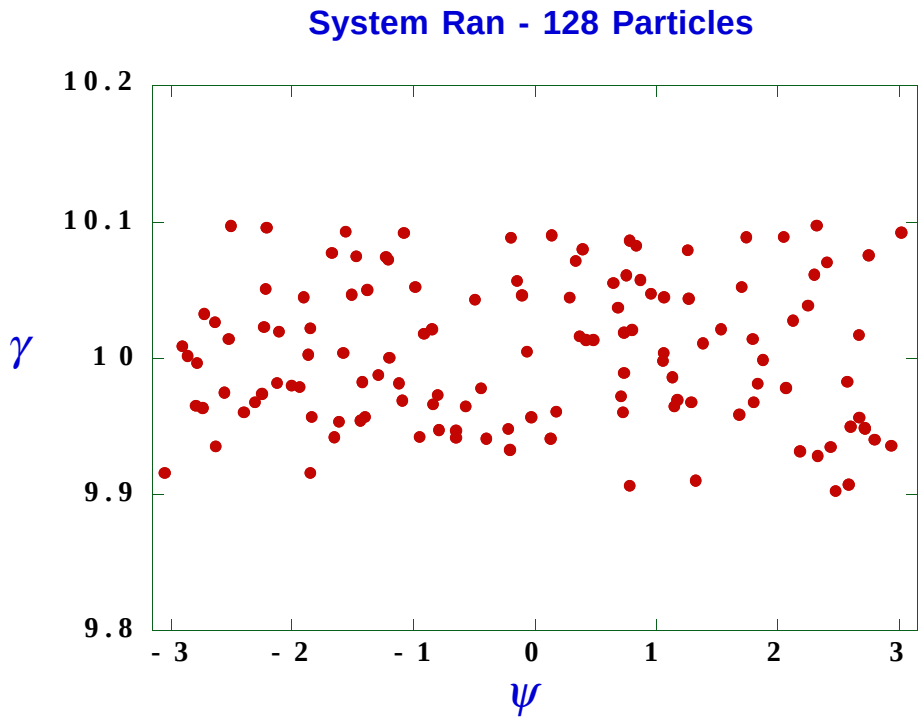
$\gamma=\gamma_m+\Delta\gamma\,(2\,R_1-1)$
$\psi=(2\,R_2-1)\,\pi$

with N=128 macroparticles for γ_m =10 and $\Delta\gamma$ =0.1.

Results for System Ran:

Answer 2 - <sinψ> = -1.6E-02
Answer 3 - <cosψ> = 4.2E-02
Answer 4 - $<\gamma-\gamma_m>$ = 3.1E-03
Answer 5 - $(<\gamma^2>-<\gamma>^2)^{1/2}/\Delta\gamma$ = 0.5245956

Note: Actual answers will depend on the system ran you use.

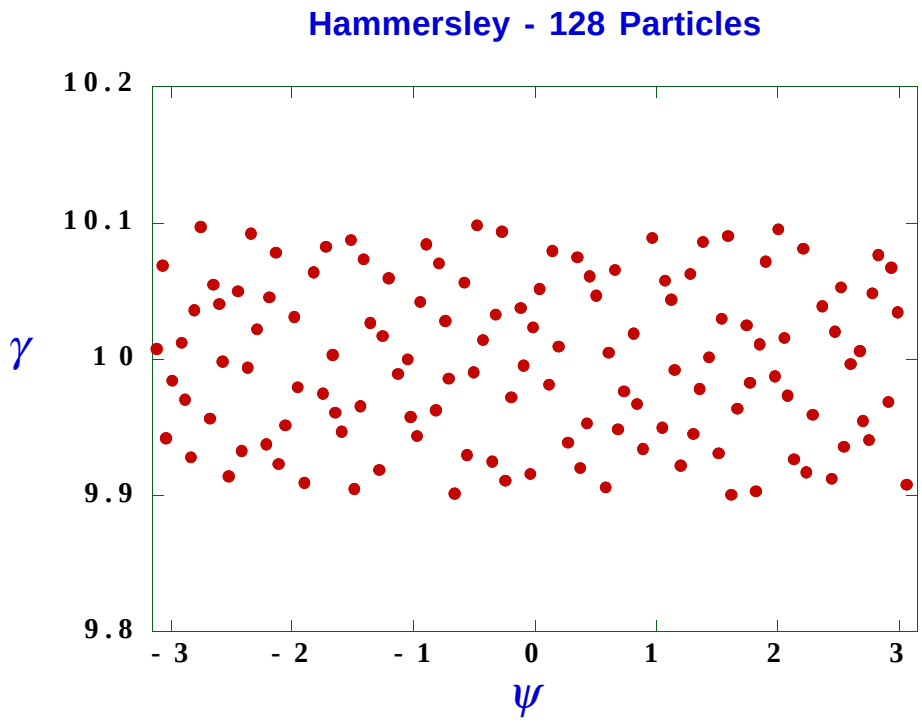


Results for Hammersley:

Answer 6 - $\langle \sin\psi \rangle$ = 1.9E-08
Answer 7 - $\langle \cos\psi \rangle$ = 7.8E-03
Answer 8 - $\langle \gamma - \gamma_m \rangle$ = -7.8E-04
Answer 9 - $(\langle \gamma^2 \rangle - \langle \gamma \rangle^2)^{1/2}/\Delta\gamma$ = 0.5772280

Note: The exact answer for the rms figure is $1/3^{1/2}$ or 0.57735.

See plots attached, and “solution” loading.for at the class web site.



For FEL simulations non-zero initial values for $\langle \sin\psi \rangle$, $\langle \cos\psi \rangle$ correspond to pre-bunching of the beam. Both real and numerical FELs depend sensitively on any initial noise in the beam. This exercise shows that the Hammersley deviates provide a quieter start than the system ran function I use. However, even this quieter distribution exhibits some pre-bunching, at the 1% level. This can be alleviated by using more particles, however a more

straightforward improvement is gained by loading with symmetrization, *i.e.*, loading N/2 particles according to

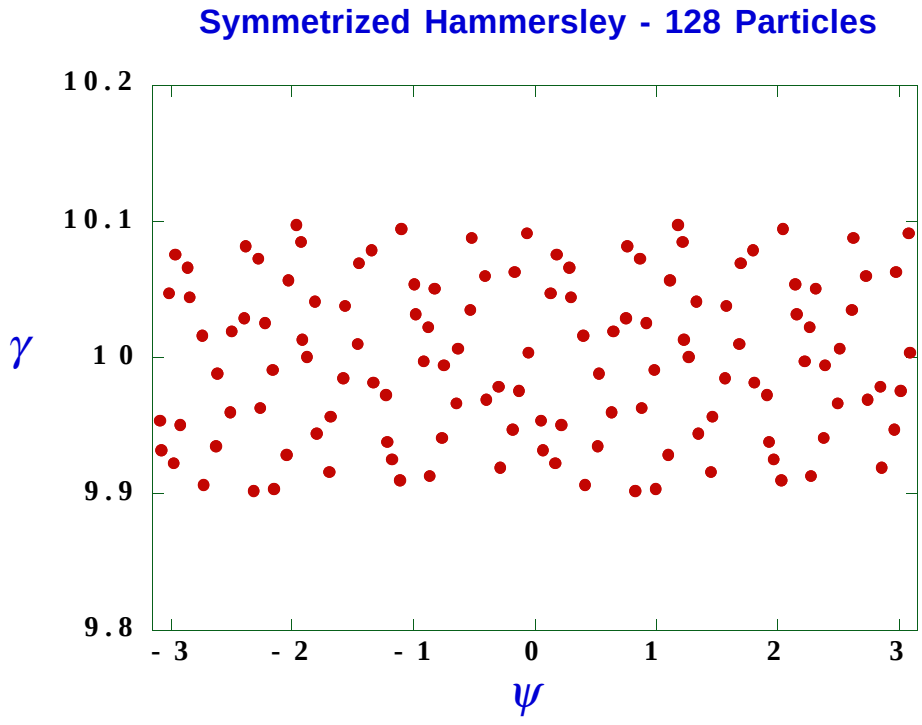
$\gamma=\gamma_m+\Delta\gamma\,(2\,R_1-1)$
$\psi=(2\,R_2-1)\,\pi$

and loading the remaining N/2, according to

$\gamma\,(i)=\gamma\,(i-N/2)$
$\psi\,(i)=\psi\,(i-N/2)+\pi$ (modulo $2\,\pi$)

Results for symmetrized Hammersley (not assigned):

Answer 2 - $\langle\sin\psi\rangle$ = 4.7E-10
Answer 3 - $\langle\cos\psi\rangle$ = -2.1E-08
Answer 4 - $\langle\gamma-\gamma_m\rangle$ = -7.6E-04
Answer 5 - $(\langle\gamma^2\rangle-\langle\gamma\rangle^2)^{1/2}/\Delta\gamma$ = 0.5768668



Note that without symmetrization in γ, the mean of the γ distribution is not quite that intended. For an FEL with gain parameter $\rho=10^{-4}$ or smaller, this error would result in an erroneous inferred value for energy or wiggler field for maximum gain. Thus in **loading.for** iopt=4 imposes a symmetrization in both variables.

More on Quasi-random Sequences

For practical simulation work, it is enough to appreciate that the loading makes a difference in the results, and that the system random number generator is usually not very satisfactory. At best it requires a larger number of particles than is necessary, and greater cpu time. At worst, it can provide misleading results. As a rule of thumb to gauge against sensitivity to the loading it is always good to repeat simulations with half the number of particles, and look for changes in results.

For the more hard-core FEL simulators, you will probably want to know that the Hammersley sequences are generated by radix-N digit reversal, where N=2,3,5,etc. Thus one starts with a natural number n (=1,2,3,...) and a radix R and from the base-R expansion,

$$n=a_0+a_1R+a_2R^2+...a_mR^m \quad (0\le a_i\le R)$$

one extracts the digits in reverse order and computes

$$\phi_R(n)= a_0\, R^{-1}+a_1R^{-2}+a_2R^{-3}+...a_mR^{-m-1}$$

to produce a number $\phi_R(n)\le 1$. This is what is accomplished by the subroutine provided with this homework. The algorithm, together with bounds on the uniformity of the distributions resulting, is described in Monte Carlo Methods, J.M Hammersley and D.C. Handscomb*, Monographs on Statistics and Applied Probability 3* (Chapman & Hall, London, 1992). (Available for loan from the lecturer). Comparison is made there with linear congruential generators.

It is interesting to note that Hammersley sequences are in fact highly correlated, they are “quasi-random” numbers. They accomplish uniformity of sampling at the expense of element to element “randomness”. This is rather like what would happen if one divided the rectangle [0,1] x [0,1] into a grid of $N^{1/2}$ x $N^{1/2}$ points, and loaded a particle at each point, except that this grid-sampling is quite tedious to code in higher dimensions, and in practice tends not to converge as rapidly as the quasi-random approach.

Also, notice that depending on what value of N is used, the loading can be rather poor; *i.e.*, the choice of N=128 was not arbitrary; a product of a power of 2 and a power of 3 accomplishes uniformity when base-2 and base-3 digit reversal are employed. Some discussion of this is found in Birdsall and Langdon (see References, Class URL).

Finally, the digit-reversed sequences are not unique examples and the subject of such uniform deviates is of considerable interest in Monte Carlo problems, where one wants to sample a distribution accurately with as few particles as possible. See for example:
ftp://ftp.iro.umontreal.ca/pub/simulation/nieder