TFY4235/FY8904 Computational Physics, Spring 2013

Solution Set 6

Problem 1.

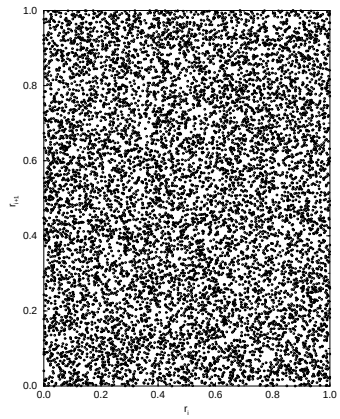
The programs that generate the sequence of random numbers are

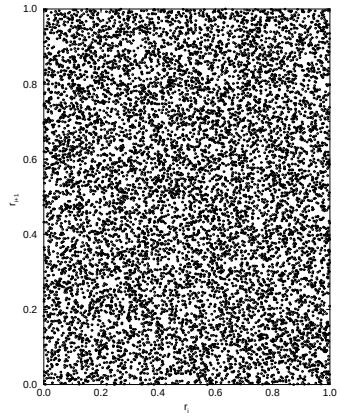
```
      program r16807
      rinv=0.5/(2.**31-1)
      ibm=4711333
      do i=1,1000
      ibm=ibm*16807
      enddo
      rold=0.
      rnew=0.
      do i=1,10000
100   rold=rnew
      ibm=ibm*16807
      rnew=float(ibm)*rinv+0.5
      if(rold.lt.0.001.and.rnew.lt.0.001) then
      write(*,*) rold,rnew
      else
      goto 100
      endif
      enddo
      end
```

and

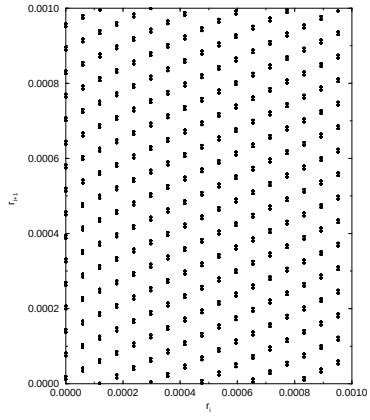
```
      program rinnb
      rinv=0.5/(2.**31-1)
      rold=0.
      rnew=0.
      do i=1,10000
100   rold=rnew
      rnew=raddran()
      if(rold.lt.0.001.and.rnew.lt.0.001) then
      write(*,*) rold,rnew
      else
      goto 100
      endif
      enddo
      end
```

Here are 10 000 random numbers generated with the "16807" algorithm.
Here is the result from the build-in generator in my compiler:

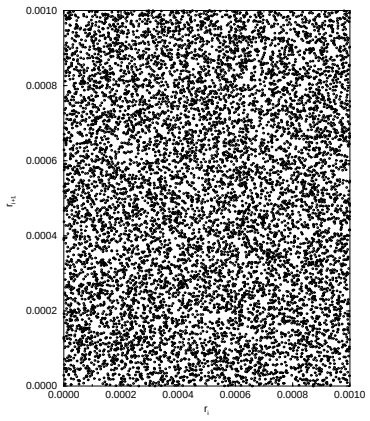




Here is the result of 10 000 numbers generated with the "16807" generator but limited to the square $(0, 0)$, $(0, 0.001)$, $(0.001, 0)$, $(0.001, 0.001)$.



The is not so nice.
Here is the result from the build-in generator. It is absolutely ok.
Should we conclude that the "16807" generator is not useable? No, the only conclusion we can make is that it is not good for this particular use. Any generator should be tested



in the context it is to be used.

Problem 2.

The following program uses the Box-Müller algorithm to generate random numbers following a gaussian with unit variance and zero mean. The programs groups the numbers into batches of size `nnum`= N. It identifies the largest number in each batch and then averages over these numbers to produce the average of the largest number among N numbers, $\langle x_{\max} \rangle$.

```
      program bboxm
c
c Box-Mueller algorithm for generating gaussian random numbers
c
      parameter(msamp=10000,nnum=50000)
c
      ibm=1955
      rinv=0.5/(2.**31-1.)
      do i=1,1000
      ibm=ibm*16807
      enddo
c
      pi2=8*atan(1.)
c
      avemax=0.
c
      do isamp=1,msamp
c
      xmax=0.
c
      do i=1,nnum/2
      ibm=ibm*16807
```

```
      y1=rinv*float(ibm)+0.5
      ibm=ibm*16807
      y2=rinv*float(ibm)+0.5
      x1=sqrt(-2.*log(y1))*cos(pi2*y2)
      x2=sqrt(-2.*log(y1))*sin(pi2*y2)
      xmax=max(xmax,x1)
      xmax=max(xmax,x2)
      enddo
c
      avemax=avemax+xmax
c
      enddo
c
      avemax=avemax/msamp
      print *,nnum,avemax
      end
```

The task was to find $\langle x_{\max} \rangle$ as a function of N. It so happens that this is a classical problem of *extreme statistics*. If the random numbers x are distributed according to a cumulative probability $P(x) = \int_{-\infty}^{x} p(x')\, dx'$, then we have that

$$P(\langle x_{\max} \rangle) = 1 - \frac{1}{N} \; . \tag{1}$$

The cumulative probability of a Gaussian is

$$P(x) = \frac{1}{2} \left[1 + \mathrm{erf} \left(\frac{x}{\sqrt{2}} \right) \right] \; . \tag{2}$$

For large x, this expression is

$$P(x \to \infty) = 1 - \frac{e^{-x^2/2\sigma^2}}{x\sqrt{2\pi/\sigma^2}} \tag{3}$$

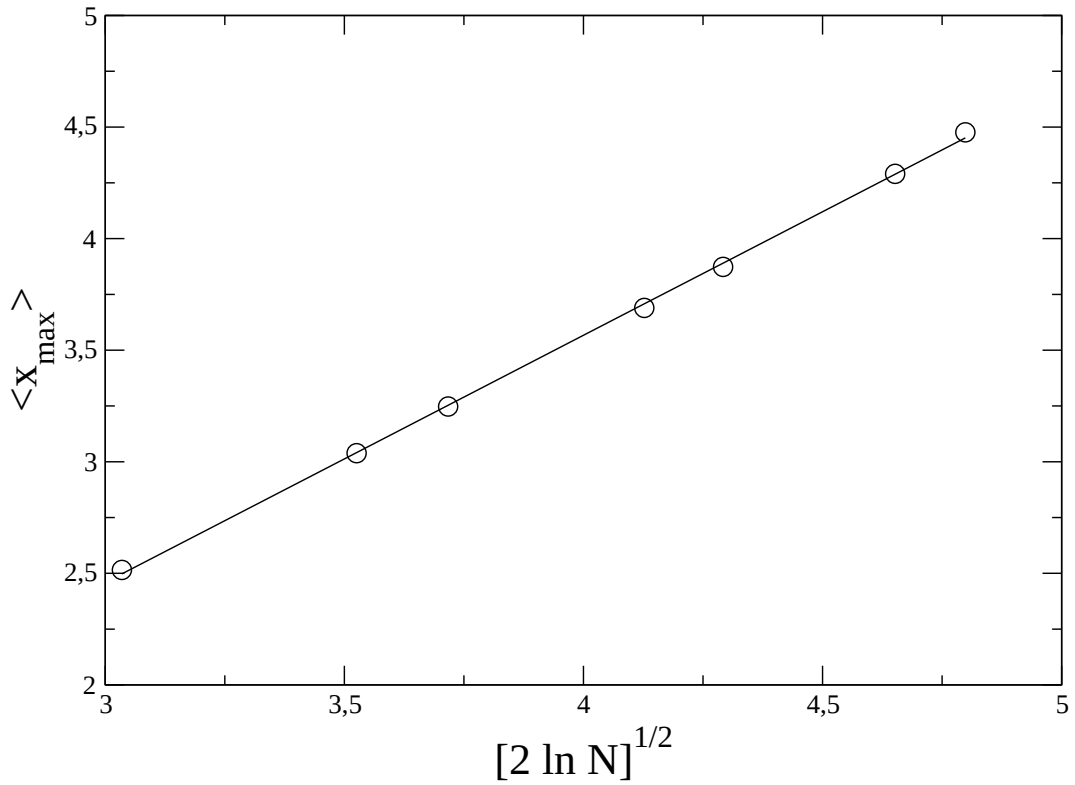
Combining Eqs. (1) and (3), we find

$$\langle x_{\max} \rangle = \sqrt{2\sigma^2 \ln N} \; . \tag{4}$$

We show the result of our simulation in the figure below. The straight line in the plot indicates that we find exactly the behavior expected from Eq. (4). I let N vary between 100 and 100 000, averaging each run over 10 000 samples.

I was in the lucky position here to know the functional form (4). Trying to fit $\langle x_{\max} \rangle$ vs. N without knowing this, is not so easy — but a typical task in computational physics.

The average height of people (from Wikipedia) is around 175 cm with a (guestimated) variance of around 10 cm. There are 6×10^9 people around. Plugging this into Eq. (4) gives $\langle x_{\max} \rangle = 175$ cm $+ 10$ cm $\sqrt{2 \ln 6 \times 10^9} = 242$ cm. This is right between the tallest and the second tallest person alive today. Hence, we may conclude that even the extremely tall follow the same Gaussian distribution as everybody else. Their height is in fact not unexpected. If they had not been that tall, somebody else would.


5
4,5
4
3,5
3
2,5
2
3
3,5
4
4,5
5
<X_max>
[2 ln N]^1/2