Lesson #7: Stochastic Simulation

Generating Pseudo-Uniforms (Our own)

In[1]:=
```
a = 1 664 525; (*1664525*)
c = 1 013 904 223; (*1013904223*)
m = 2^32;
next[x_] := Mod[a x + c, m]
```

In[14]:=
```
x0 = 3;
nn = 10^5;
lst = N[1/m NestList[next, x0, nn]];
Mean[lst]
Variance[lst]
Variance[UniformDistribution[{0, 1}]] // N
Covariance[Drop[lst, 1], Drop[lst, -1]] / Variance[lst]
Manipulate[ListPlot[Take[lst, {n, n + 1000}], Ticks → {False, Automatic}], {n, 1, nn - 1000}]
ListPlot[1/nn BinCounts[lst, {0, 1, 0.01}], Filling → Axis, PlotRange → {0, 0.01 * 1.1}]
```

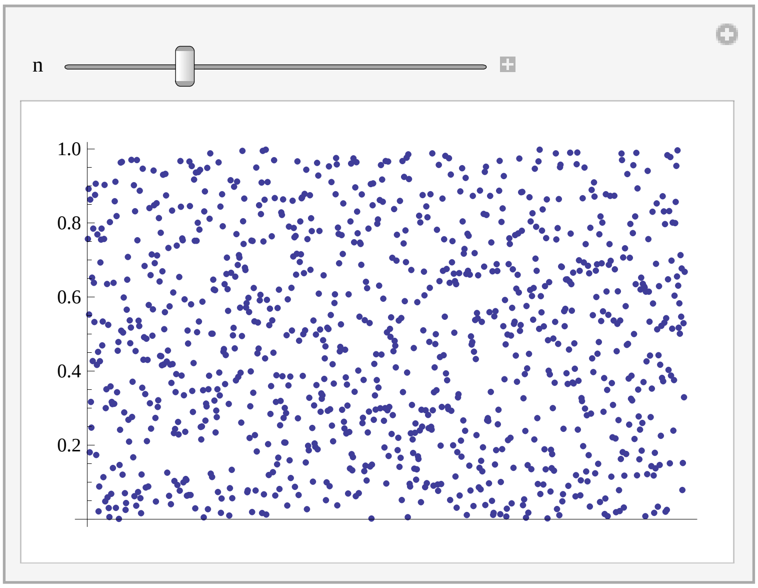
Out[17]= 0.49897

Out[18]= 0.0829005

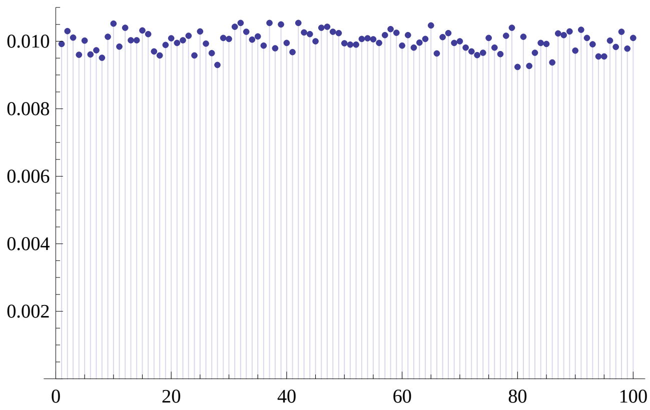
Out[19]= 0.0833333

Out[20]= -0.000806454

Out[21]=



Out[22]=



Using *Mathematica*'s RandomReal[] and SeedRandom[]

In[23]:=
```
SeedRandom[3];
lst = Table[RandomReal[], {nn}];
Mean[lst]
Variance[lst]
Covariance[Drop[lst, 1], Drop[lst, -1]] / Variance[lst]
```

Out[25]= 0.499649

Out[26]= 0.083959

Out[27]= 0.00741564

The Inverse Probability Transform

In[28]:=
```
λ = 1 / 2;
```

In[29]:=
```
inv[x_] := InverseCDF[ExponentialDistribution[λ], x]
```

In[30]:=
```
inv[x]
```

Out[30]= $-2\,\text{Log}[1-x]$

In[31]:=
```
nn = 500;
```

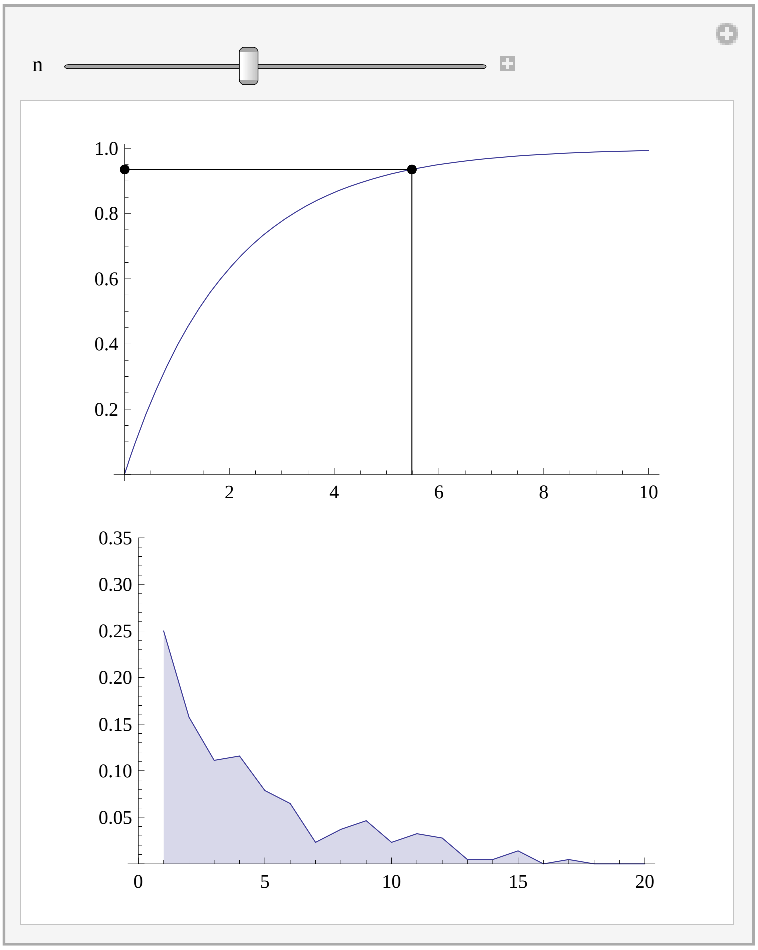
In[32]:=
```
unifs = Table[RandomReal[{0, 1}], {nn}];
```

In[33]:=
```
element[un_] := {PointSize[Medium], Point[{0, un}], Line[{{0, un}, {inv[un], un}}],
  Point[{inv[un], un}], Line[{{inv[un], un}, {inv[un], 0}}]}
```

In[34]:=
```
grps = Map[element, unifs];
```

In[35]:=
```
Manipulate[
 GraphicsColumn[{Plot[CDF[ExponentialDistribution[1 / 2], x], {x, 0, 10},
    Epilog → grps[[n]]],
   ListPlot[1/n BinCounts[Map[inv, Take[unifs, n]], {0, 10, 0.5}],
    AxesOrigin → {0, 0}, Filling → Axis, Joined → True, PlotRange → {0, 0.35}]}],
 {n,
  1,
  nn,
  1}]
```

Out[35]=



Specialized Methods

In[36]:=
```
Map[InverseCDF[NormalDistribution[], #] &, {0, 0.25, 0.5, 0.75, 1}]
```

Out[36]= {-∞, -0.67449, 0., 0.67449, ∞}

In[37]:=
```
Timing[Table[InverseCDF[NormalDistribution[], RandomReal[{0, 1}]], {10 000}];]
```

Out[37]= {3.484, Null}

In[38]:=
```
myNormal[] := Module[{},
  angle = 2 Pi RandomReal[];
  radius = Sqrt[-Log[RandomReal[]]];
  {radius Cos[angle], radius Sin[angle]}
 ]
```

In[39]:=
```
Timing[Table[myNormal[][[1]], {10 000}];]
```

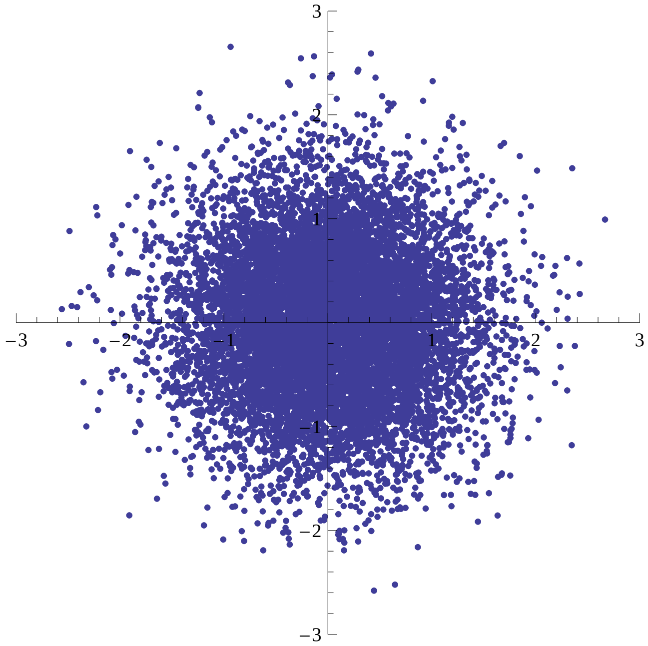
Out[39]=
```
{0.172, Null}
```

In[40]:=
```
Timing[Table[RandomReal[NormalDistribution[]], {10 000}];]
```

Out[40]=
```
{0.047, Null}
```

In[41]:=
```
ListPlot[Table[myNormal[], {10 000}] ,
 AspectRatio → Automatic, PlotRange → {{-3, 3}, {-3, 3}}]
```

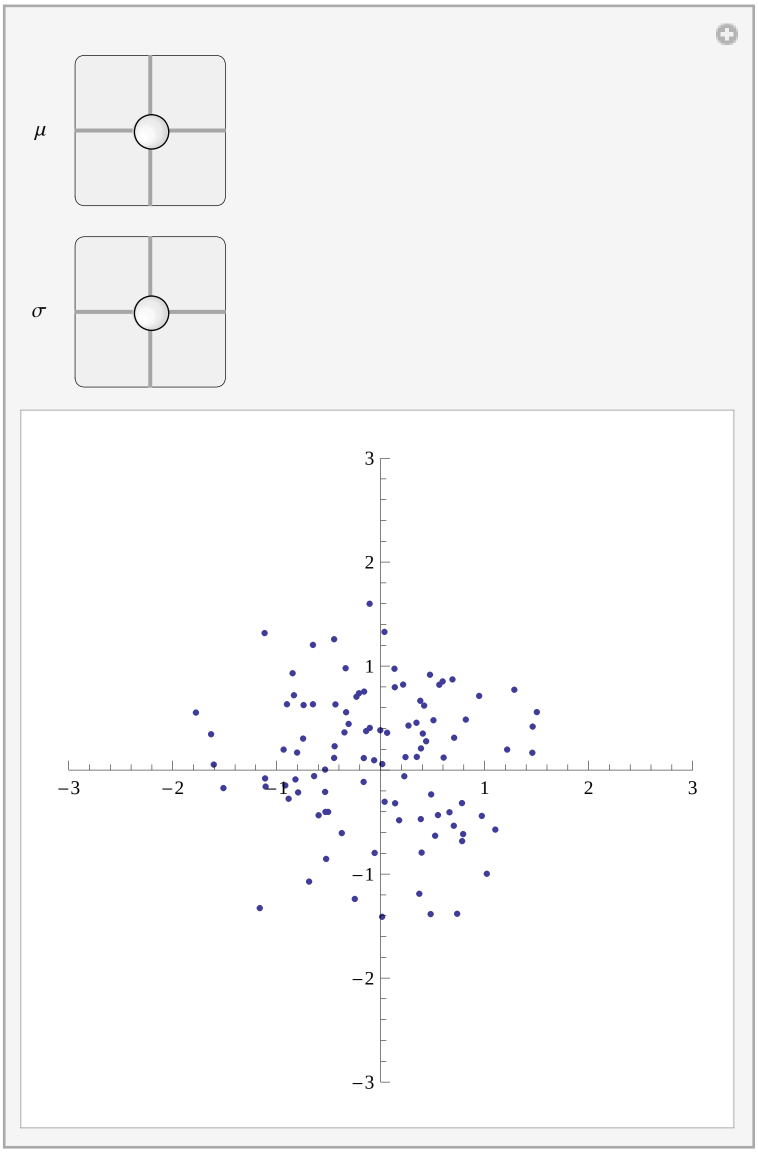
Out[41]=



Scaling Random Variables

In[42]:=
```
Manipulate[
 ListPlot[Map[μ + σ # &, Table[myNormal[], {100}]], AspectRatio → Automatic,
  PlotRange → {{-3, 3}, {-3, 3}}], {{μ, {0, 0}}, {-2, -2}, {2, 2}},
 {{σ, {1, 1}}, {0, 0}, {2, 2}}]
```

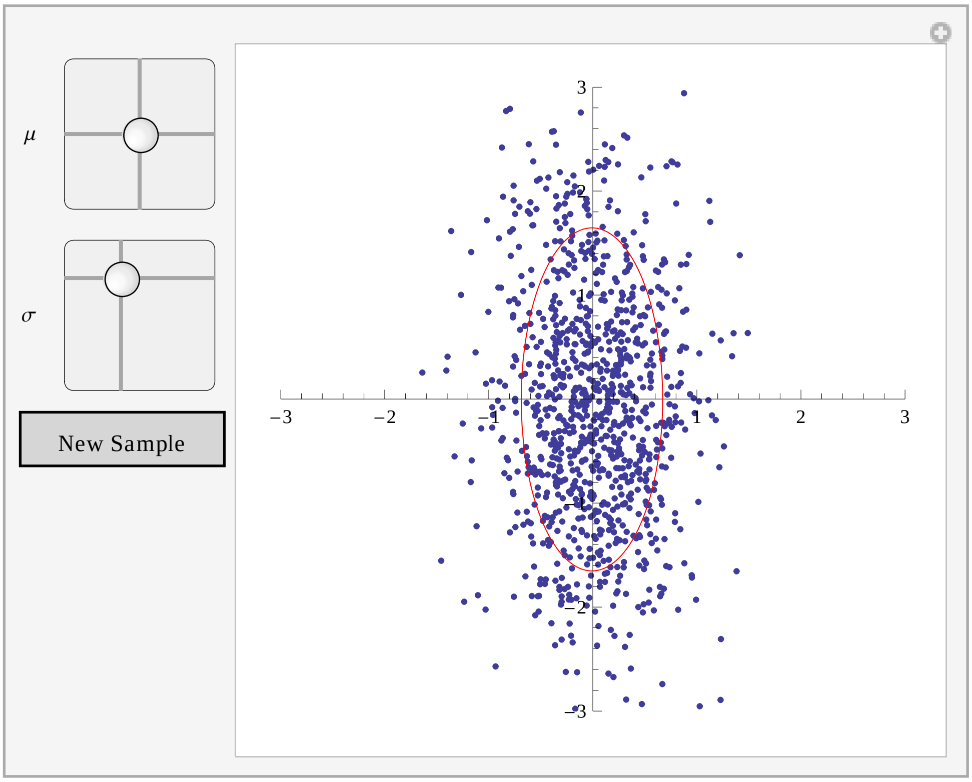
Out[42]=



In[43]:=
```
rvs = Table[myNormal[], {1000}];
```

In[44]:=
```
Manipulate[
 ListPlot[Map[μ + σ # &, rvs], AspectRatio → Automatic, PlotRange → {{-3, 3}, {-3, 3}},
  Epilog → {Red, Circle[μ, σ]}], {{μ, {0, 0}}, {-2, -2}, {2, 2}},
 {{σ, {1, 1}}, {0, 0}, {2, 2}}, Button["New Sample", (rvs = Table[myNormal[], {1000}]) &],
 ControlPlacement → Left
]
```

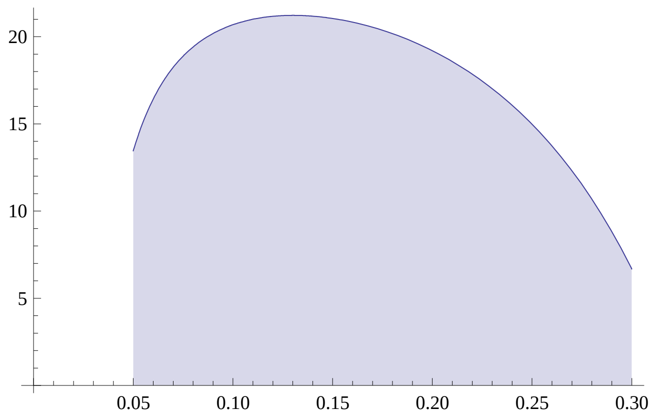
Out[44]=



Monte Carlo Integration

$$f[x\_] := 30 + \left(\frac{x^2 + \sqrt{\frac{\text{Cos}[x]}{\text{Sin}[x]}}}{\text{Log}[x]}\right)^7$$

```
Plot[f[x], {x, 0.05, 0.3}, Filling → Axis, AxesOrigin → {0, 0}]
```



```
Integrate[f[x], {x, 0.05, 0.3}]
```

```
$Aborted
```

```
NIntegrate[f[x], {x, 0.05, 0.3}]
```

```
4.41695
```

```
(0.30 - 0.05) * 15
```

```
3.75
```

```
Table[
  (Table[f[RandomReal[{0.05, 0.30}]], {10 000}] // Mean) * (0.30 - 0.05)
  ,
  {10}] // MatrixForm
```

$$\begin{pmatrix} 4.41274 \\ 4.42014 \\ 4.43274 \\ 4.40553 \\ 4.39311 \\ 4.42224 \\ 4.40408 \\ 4.41407 \\ 4.42084 \\ 4.41038 \end{pmatrix}$$

```
f[x_, y_, z_, w_] := Cos[100 x] Cos[100 y] Cos[100 z] Cos[100 w]
```

```
Integrate[f[x, y, z, w], {x, 0, 1}, {y, 0, 1}, {z, 0, 1}, {w, 0, 1}]
```

$$\frac{\mathrm{Sin}[100]^4}{100\,000\,000}$$

```
% // N
```

6.57441×10^{-10}

```
NIntegrate[f[x, y, z, w], {x, 0, 1}, {y, 0, 1}, {z, 0, 1}, {w, 0, 1}]
```

NIntegrate::slwcon :
Numerical integration converging too slowly; suspect one of the following: singularity, value of the integration is 0, highly oscillatory integrand, or WorkingPrecision too small. ≫

NIntegrate::eincr :
The global error of the strategy GlobalAdaptive has increased more than 2000 times. The global error is expected to decrease monotonically after a number of integrand evaluations. Suspect one of the following: the working precision is insufficient for the specified precision goal; the integrand is highly oscillatory or it is not a (piecewise) smooth function; or the true value of the integral is 0. Increasing the value of the GlobalAdaptive option MaxErrorIncreases might lead to a convergent numerical integration. NIntegrate obtained −0.0000612586 and 0.019804173024785054` for the integral and error estimates. ≫

$0. \times 10^{-2}$

```
Table[Apply[f, RandomReal[{0, 1}, 4]], {10 000}] // Mean
```

```
0.000137419
```

Class Exercise: Random Process Simulation

- **Random Walks**
- **Markov Chains**
- **Markov Jump Processes**