Clear all symbols from previous evaluations to avoid problems

In[1]:=
```
Clear["Global`*"]
```

Examples of generators of pseudorandom numbers

Linear congruent generator

Pseudorandom numbers are generated by

$$x_{i+1} = (a\, x_i + c) \bmod m \tag{1}$$

According to the choice of a, c, m, and x_0 we get pseudorandom number generators of different quality

In[2]:=
```
myRandom[n_, dim_, par_, seed_] := Module[{out = ConstantArray[0, {n, dim}]},
   xRandom = seed;
   Do[
    Do[
     xRandom = Mod[par[[1]] xRandom + par[[2]], par[[3]]];
     out[[i, j]] = xRandom,
     {j, 1, dim}],
    {i, 1, n}
   ];
   Return[out]
  ];
```

Generator RANDU

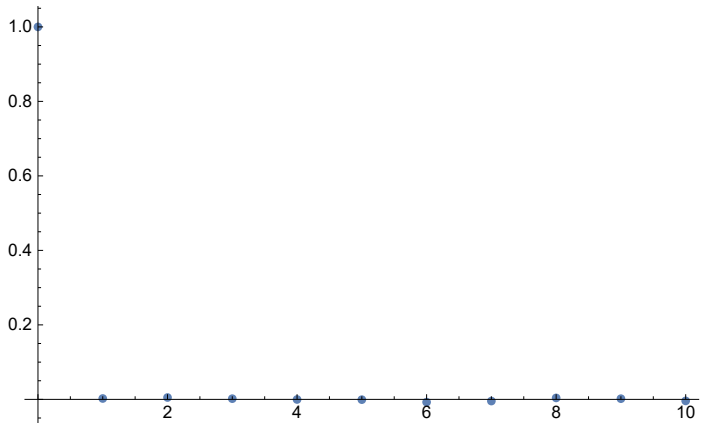
Expected value, variance and autocorrelation test:

In[3]:=
```
n = 10^5;
points = N[Flatten[myRandom[n, 1, {2^16 + 3, 0, 2^31}, 100] / 2^31]];
expectedValue = Total[points] / n ;
variance = Total[points^2] / n - expectedValue^2 ;
Print["Expected value and variance should be: ", N[1 / 2], ", ", N[1 / 12]];
Print["Expected value and variance obtained:  ", expectedValue, ", ", variance];
kmax = 10; corr = ConstantArray[1.0, kmax + 1];
Do[
  corr[[k + 1]] =
   {k, (points[[1 ;; n - k]].points[[k + 1 ;; n]] / (n - k) - expectedValue^2) / variance},
  {k, 0, kmax}
 ];
ListPlot[corr, PlotRange → All]
```

```
Expected value and variance should be: 0.5, 0.08333333333

Expected value and variance obtained:  0.5007657519, 0.08315221372
```

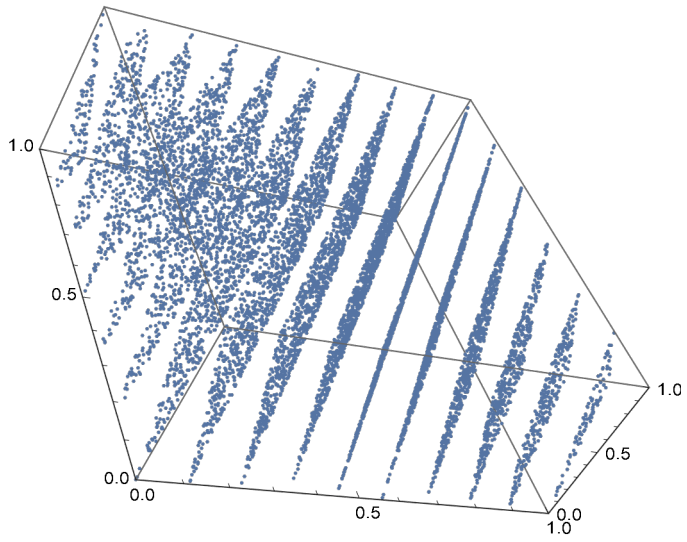
Out[11]=



But points in 3D are in 15 planes:

In[12]:=
```
n = 10^4;
points = myRandom[n, 3, {2^16 + 3, 0, 2^31}, 1] / 2^31;
ListPointPlot3D[points, PlotStyle → PointSize[Small],
 PlotRange → {{0, 1}, {0, 1}, {0, 1}}]
```

Out[14]=



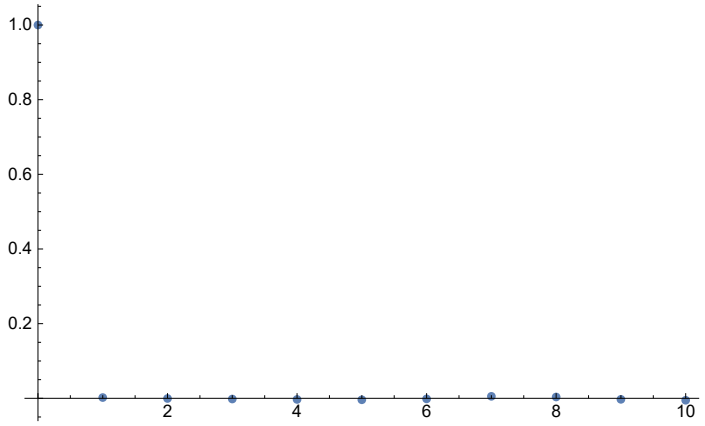
Generator from Apple CarbonLib

Expected value, variance and autocorrelation test:

In[15]:=
```
n = 10^5;
m = 2^31 - 1;
points = N[Flatten[myRandom[n, 1, {16807, 0, m}, 100] / m]];
expectedValue = Total[points] / n ;
variance = Total[points^2] / n - expectedValue^2 ;
Print["Expected value and variance should be: ", N[1 / 2], ", ", N[1 / 12]];
Print["Expected value and variance obtained:  ", expectedValue, ", ", variance];
kmax = 10; corr = ConstantArray[1.0, kmax + 1];
Do[
  corr[[k + 1]] =
   {k, (points[[1 ;; n - k]].points[[k + 1 ;; n]] / (n - k) - expectedValue^2) / variance},
  {k, 0, kmax}
 ];
ListPlot[corr, PlotRange → All]
```

```
Expected value and variance should be: 0.5, 0.08333333333
Expected value and variance obtained:  0.501179104, 0.08311243499
```

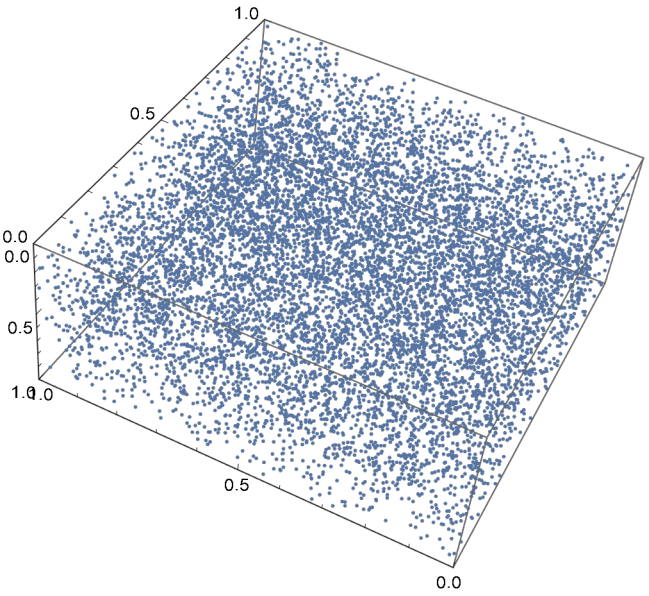
Out[24]=



In[25]:=
```
m = 2^31 - 1;
N[(3 ! m)^1/3]
ListPointPlot3D[myRandom[10000, 3, {16807, 0, m}, 1] / m,
 PlotStyle → PointSize[Small], PlotRange → {{0, 1}, {0, 1}, {0, 1}}]
```

Out[26]=
```
2344.374768
```

Out[27]=



Mathematica's LCG generator

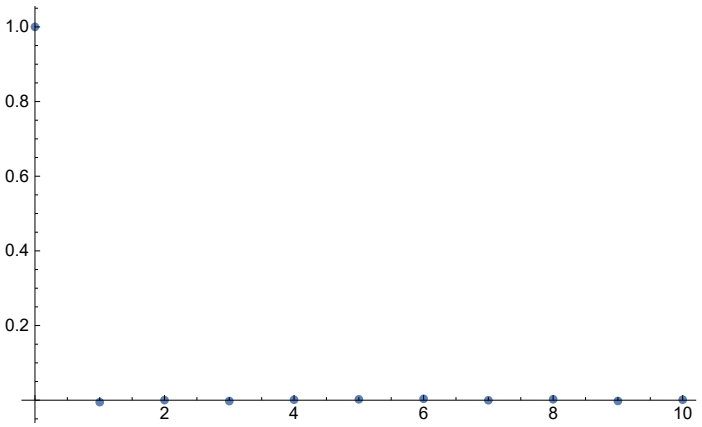
Expected value, variance and autocorrelation test:

In[28]:=
```
n = 10^5;
points = RandomReal[{0, 1}, n];
expectedValue = Total[points] / n ;
variance = Total[points^2] / n - expectedValue^2 ;
Print["Expected value and variance should be: ", N[1 / 2], ", ", N[1 / 12]];
Print["Expected value and variance obtained:  ", expectedValue, ", ", variance];
kmax = 10; corr = ConstantArray[1.0, kmax + 1];
Do[
  corr[[k + 1]] =
   {k, (points[[1 ;; n - k]].points[[k + 1 ;; n]] / (n - k) - expectedValue^2) / variance},
  {k, 0, kmax}
 ];
ListPlot[corr, PlotRange → All]
```

```
Expected value and variance should be: 0.5, 0.08333333333

Expected value and variance obtained:  0.5000830409, 0.08294680256
```

Out[36]=



In[37]:=

```
SeedRandom[1, Method → "Congruential"];
ListPointPlot3D[RandomReal[{0, 1}, {10000, 3}],
 PlotStyle → PointSize[Small], PlotRange → {{0, 1}, {0, 1}, {0, 1}}]
```

Out[38]=