Clear all symbols from previous evaluations to avoid problems

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

# Generating random variables with a specified distribution

## Special methods for the normal distribution

The normal distridution $N(\mu, \sigma^2)$

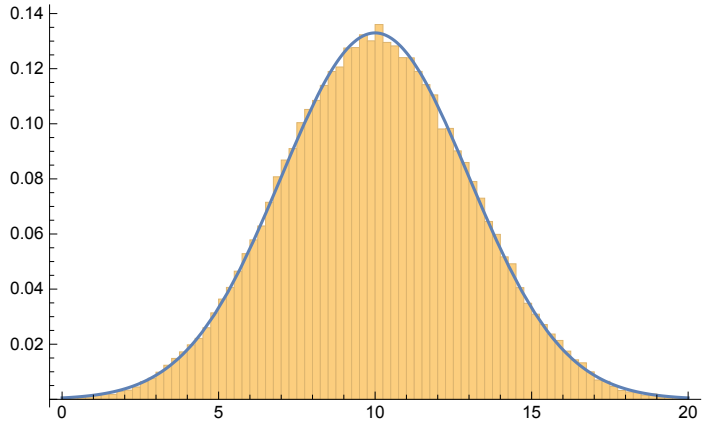
$$\rho(x) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-(x-\mu)^2/2\,\sigma^2\right) \tag{1}$$

can be efficiently generated for example using the central limit theorem. If we sum up 12 uniformally distributed ($\mu = 1/2,\ \sigma^2 = 1/12$) random variables we get $N(6, 1)$. Thus by shifting to 0 and then by multiplying by $\sigma$ and adding $\mu$ we get the general normal distribution:

In[2]:=
```
myNormalSums[n_, par_] := Module[{out = ConstantArray[0.0, {n}]},
   Do[
    out[[i]] = 0.0;
    Do[
     out[[i]] = out[[i]] + RandomReal[],
     {j, 1, 12}];
    out[[i]] = par[[1]] + par[[2]] (out[[i]] - 6.0),
    {i, 1, n}
   ];
   Return[out]
  ];
```

In[3]:=
```
μ = 10; σ2 = 3;
data = myNormalSums[100000, {μ, σ2}];
Show[Histogram[data, {0, 20, 1/4}, "ProbabilityDensity"],
 Plot[PDF[NormalDistribution[μ, σ2], x], {x, 0, 20}]]
```

Out[5]=




Another efficient way of generating the normal distribution is to consider a Gaussian distribution in two dimensions and use the polar coordinates to transform it to a product of exponential and uniform distribution:

$$\exp\left(-(x_1^2 + x_2^2)/2\right) dx_1\, dx_2 \sim \exp\left(-r^2/2\right) r\, dr\, d\theta \sim \exp(-u)\, du\, d\theta \tag{2}$$

If $u$ is generated between 0 and $\infty$ with an exponential distribution using

$$y(u) = \int_0^u \exp(-t)\, dt = 1 - \exp(-u) \quad \Longrightarrow \quad u(y) = -\ln(1 - y) \tag{3}$$

and $\theta$ is uniformly distributed between 0 and $2\pi$ then

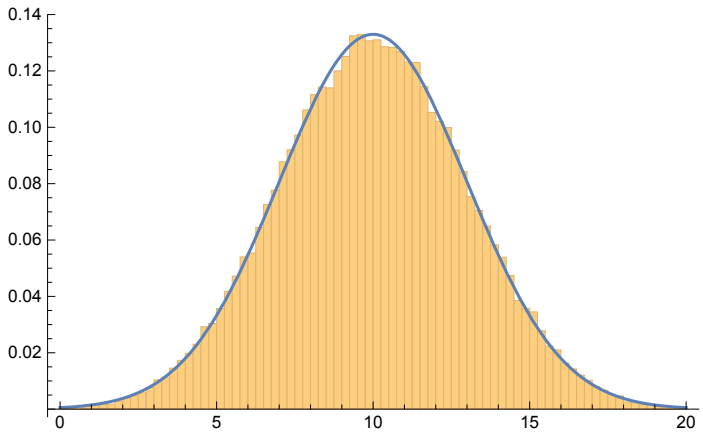
$$x_1 = \sqrt{2\,u}\ \cos(\theta),\ \ x_2 = \sqrt{2\,u}\ \sin(\theta) \tag{4}$$

are distributed normally.

In[6]:=
```
myNormal2D[n_, par_] := Module[{r, θ, out = ConstantArray[0.0, {n + Mod[n, 2]}]},
   Do[
    r = Sqrt[-2 Log[1 - RandomReal[]]];
    θ = 2 π RandomReal[];
    out[[i]] = par[[1]] + par[[2]] r Cos[θ];
    out[[i + 1]] = par[[1]] + par[[2]] r Sin[θ],
    {i, 1, n + Mod[n, 2], 2}
   ];
   Return[out[[1 ;; n]]]
  ];
```

In[7]:=
```
μ = 10; σ2 = 3;
data = myNormalSums[100000, {μ, σ2}];
Show[Histogram[data, {0, 20, 1/4}, "ProbabilityDensity"],
 Plot[PDF[NormalDistribution[μ, σ2], x], {x, 0, 20}]]
```

Out[9]=



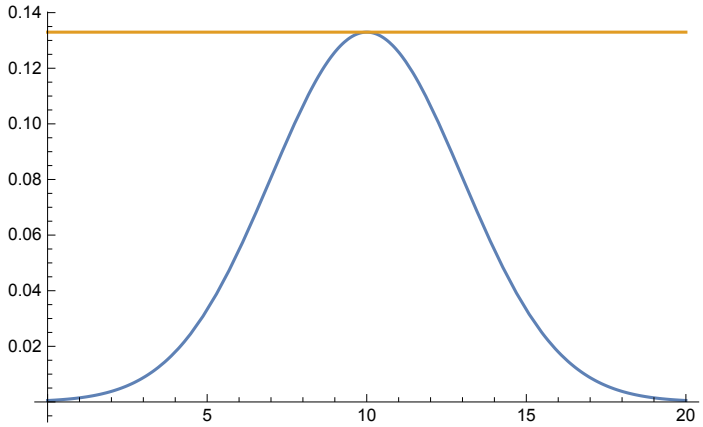
## von Neumann method

In[10]:=
```
myCounter = 0;
myRandomNeumann[ρ_, a_, b_, R_, n_] := Module[{g, x, y, out = ConstantArray[0.0, n]},
   Do[
    g = 1;
    While[g == 1,
     x = a + RandomReal[] (b - a);
     y = R RandomReal[];
     If[y < ρ[x], g = 0, myCounter++]
    ];
    out[[i]] = x,
    {i, 1, n}
   ];
   Return[out]
  ];
```

Normal distribution generated by von Neumann method:

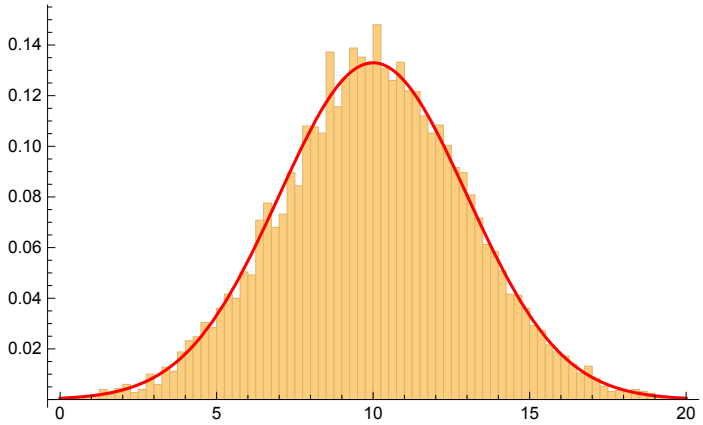
In[12]:=
```
myCounter = 0;
a = 0; b = 20;
μ = 10;  σ2 = 3;
ρ := PDF[NormalDistribution[μ, σ2], #] &;
R := PDF[NormalDistribution[μ, σ2], μ];
Plot[{ρ[x], R}, {x, a, b}]
data = myRandomNeumann[ρ, a, b, R, 10000];
Print[myCounter, " points rejected"]
Show[Histogram[data, {a, b, 1/4}, "ProbabilityDensity"],
 Plot[PDF[NormalDistribution[μ, σ2], x], {x, a, b}, PlotStyle → Red]]
```

Out[17]=




```
16468 points rejected
```

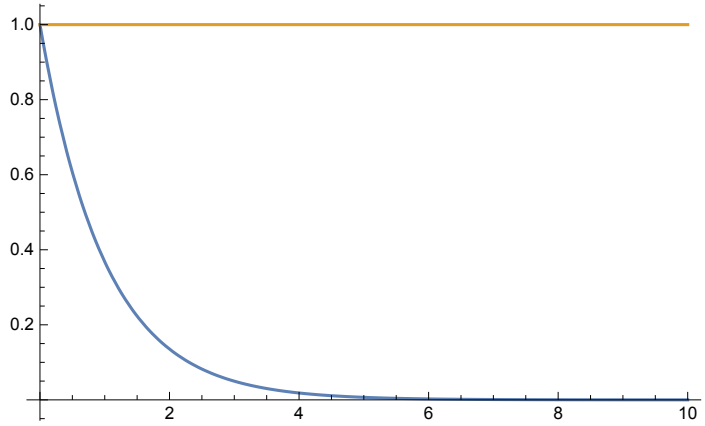
Out[20]=




Exponential distribution generated by von Neumann method (not very efficient):

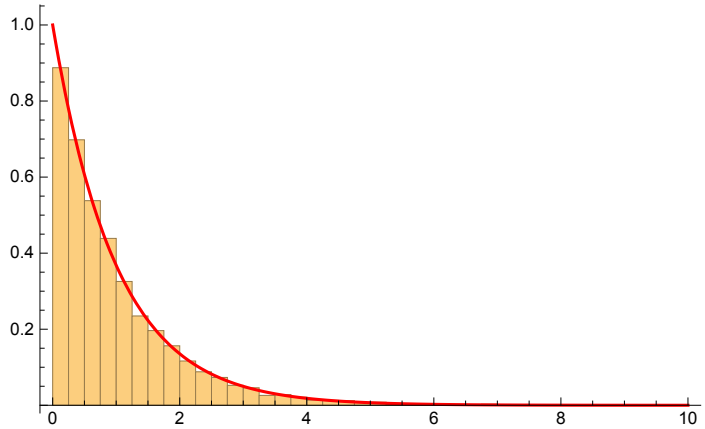
In[21]:=
```
myCounter = 0;
a = 0; b = 10;
ρ := Exp[-#] &;
R := 1;
Plot[{ρ[x], R}, {x, a, b}]
data = myRandomNeumann[ρ, a, b, R, 10000];
Print[myCounter, " points rejected"]
Show[Histogram[data, {a, b, 1/4}, "ProbabilityDensity"],
 Plot[Exp[-x], {x, a, b}, PlotRange → {0, 1}, PlotStyle → Red]]
```

Out[25]=




```
91532 points rejected
```

Out[28]=




## Metropolis algorithm

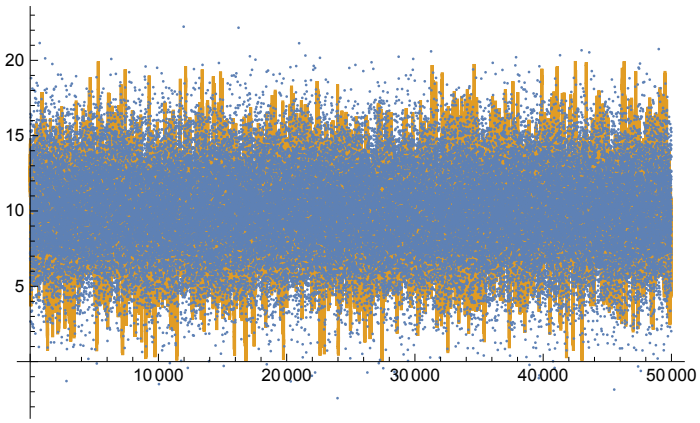
One dimensional case, $\rho$ must be a function of $x$:

In[29]:=
```
myRandomMetropolis1D[n_, ρ_, a_, b_, x0_, δ_] :=
  Module[{x = x0, xtrial, ratio, ρx, out = ConstantArray[0.0, n]},
   ρx = ρ[x];
   Do[
    (* try a new random step *)
    xtrial = x + δ RandomReal[{-1, 1}];
    (* check if we are still in the given region, if not, adjust *)
    If[xtrial < a, xtrial = a + δ RandomReal[]];
    If[xtrial > b, xtrial = b - δ RandomReal[]];
    ratio = ρ[xtrial] / ρx;
    (* decide whether to make this step according to the distribution *)
    If[ratio ≥ 1 || ratio > RandomReal[], x = xtrial; ρx = ρx ratio];
    out[[i]] = x,
    {i, 1, n}
   ];
   Return[out]
  ];
```

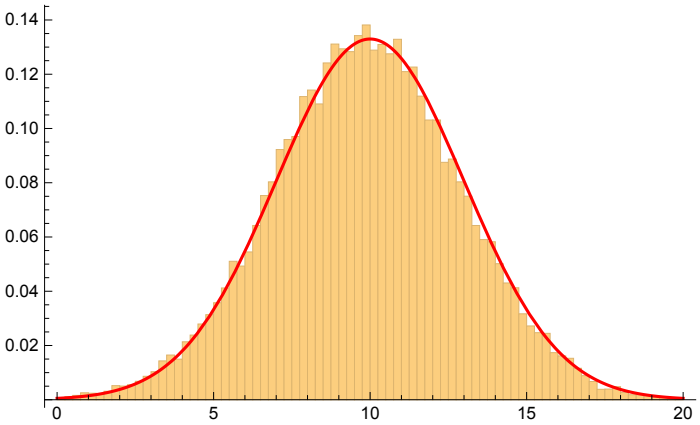
Applied to the normal distribution:

In[30]:=
```
n = 50000;
a = 0; b = 20;
μ = 10;  σ2 = 3;
ρ := PDF[NormalDistribution[μ, σ2], #] &;
data1 = myRandomMetropolis1D[n, ρ, a, b, 10.0, 2.0];
data2 = RandomReal[NormalDistribution[μ, σ2], {n}];
ListPlot[{data2, data1}, Joined → {False, True}]
Show[Histogram[data1, {a, b, 1/4}, "ProbabilityDensity"],
 Plot[PDF[NormalDistribution[μ, σ2], x], {x, a, b}, PlotStyle → Red]]
```

Out[36]=



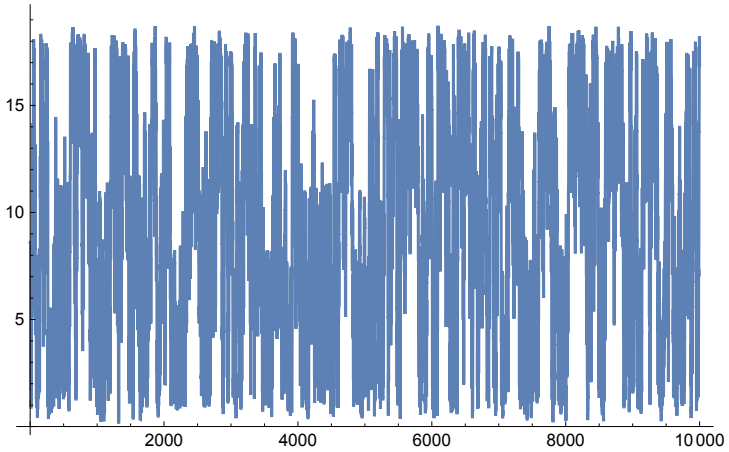

Out[37]=



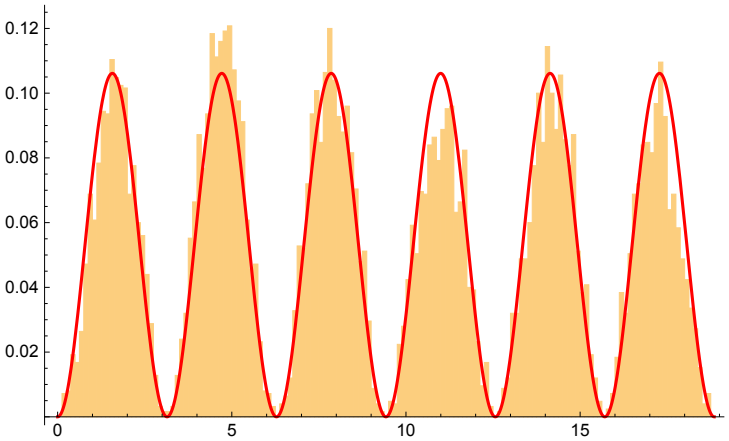
Applied to the distribution with several maxima (the choice of $\delta$ plays an important role in whether we get reasonable results):

In[38]:=
```
n = 10 000;
a = 0; b = 6 π;
ρ := 2 Sin[#]^2 / b + 0.00001 &;
data1 = myRandomMetropolis1D[n, ρ, a, b, (a + b) / 2, (b - a) / 5];
ListPlot[{data1}, Joined → {True}]
Show[Histogram[data1, {a, b, 1 / 8}, "ProbabilityDensity"],
 Plot[ρ[x], {x, a, b}, PlotStyle → Red]]
```

Out[42]=




Out[43]=

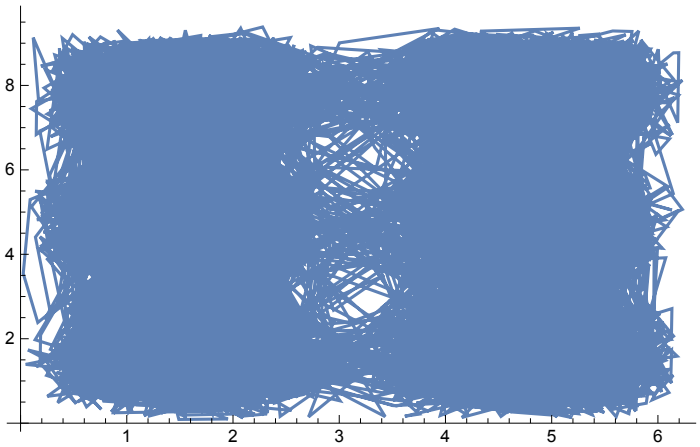



Metropolis algorithm for $n$-dimensional case:

In[44]:=
```
myRandomMetropolisND[n_, dim_, ρ_, a_, b_, x0_, δ_] :=
  Module[{x = x0, xtrial, ratio, ρx, out = ConstantArray[0.0, {n, dim}]},
   ρx = ρ[x];
   Do[
    (* try a new random step *)
    xtrial = x + δ RandomReal[{-1, 1}, dim];
    (* check if we are still in the given region, if not, adjust *)
    Do[
     If[xtrial[[d]] < a[[d]], xtrial[[d]] = a[[d]] + δ[[d]] RandomReal[]];
     If[xtrial[[d]] > b[[d]], xtrial[[d]] = b[[d]] - δ[[d]] RandomReal[]],
     {d, 1, dim}
    ];
    (* decide whether to make this step according to the distribution *)
    ratio = ρ[xtrial] / ρx;
    If[ratio ≥ 1 || ratio > RandomReal[], x = xtrial; ρx = ρx * ratio];
    out[[i]] = x,
    {i, 1, n}
   ];
   Return[out]
  ];
```

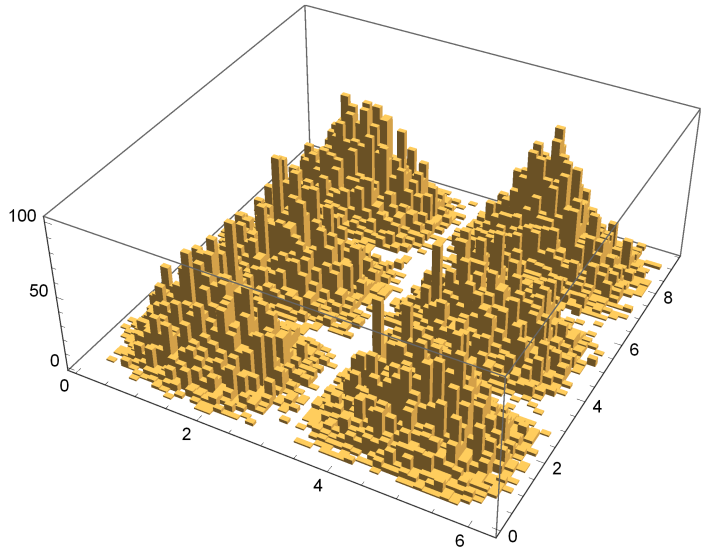
In[45]:=
```
n = 50000;
a = {0, 0}; b = {2 π, 3 π};
(* ρ:=2Sin[#1]^2Sin[#2]^2/Times@@b &; *)
ρ := 2 Sin[#[[1]]]^2 Sin[#[[2]]]^2 / (Times @@ b) + 0.00001 &;
x0 = {3, 4}; δ = {1, 2};
data1 = myRandomMetropolisND[n, 2, ρ, a, b, x0, δ];
ListPlot[{data1}, Joined → {True}, PlotRange → All]
```

Out[50]=



In[51]:=
```
Histogram3D[data1, {{a[[1]], b[[1]], 1/8}, {a[[2]], b[[2]], 1/8}}]
```

Out[51]=




In[52]:=
```
Plot3D[ρ[{x, y}], {x, a[[1]], b[[1]]}, {y, a[[2]], b[[2]]}, PlotRange → All]
```

Out[52]=