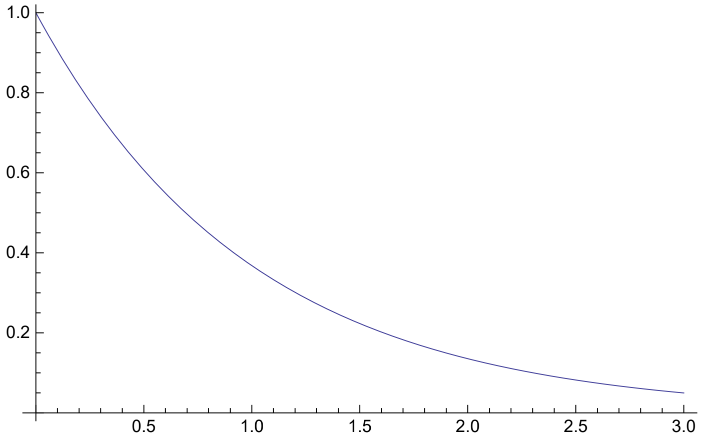
Plots & Animations

Simple 2d plots

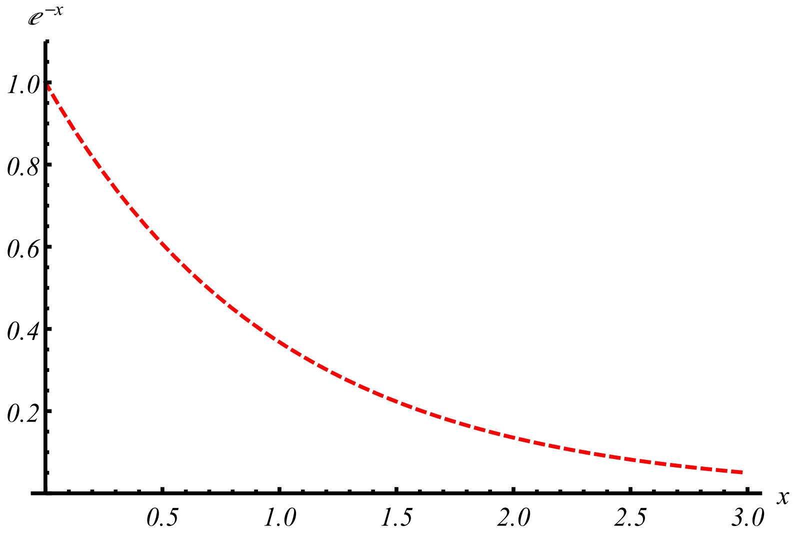
Physical processes and corresponding mathematical formulas can be visualized by plots. The simplest plot of a single function f[x] can be created as follows

```
f[x_] := Exp[-x];  (* Comment: Define function *)
Plot[f[x], {x, 0, 3}]  (* Plot function *)
```



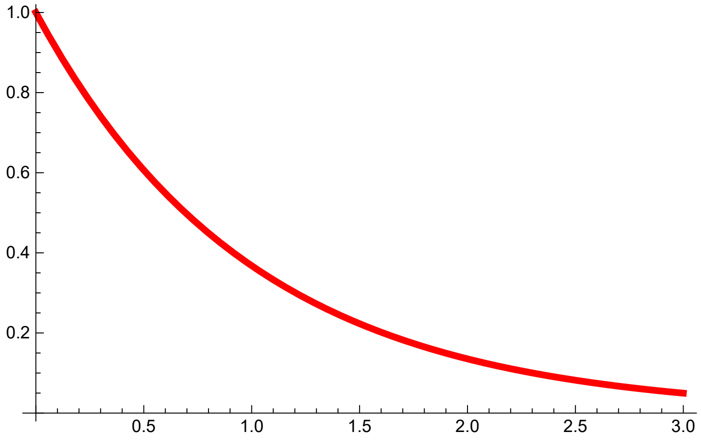
One can add options for the line color, thickness, and style, as well as for the vertical plot range, axes style and labeling to obtain a publishing-quality plot

```
Plot[f[x], {x, 0, 3}, PlotRange → {0, 1.1},
 PlotStyle → {Red, Thick, Dashed}, AxesStyle → Thick, AxesLabel → {x, Exp[-x]},
 BaseStyle → {FontSize → 14, FontFamily → "Times", Italic}]
```



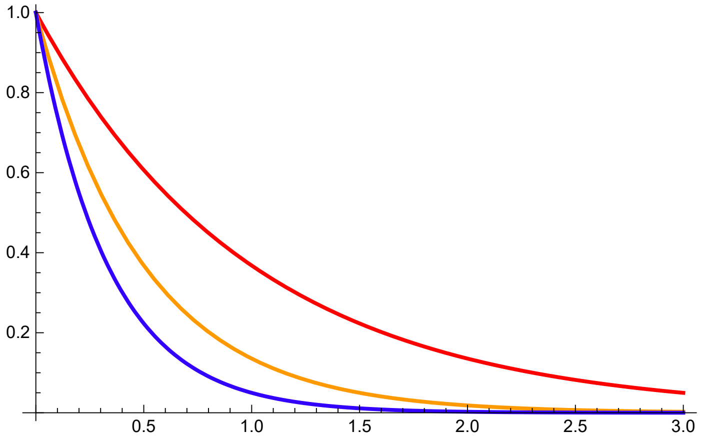
Attributes of the plot style can be set in a more detailed form

```
Plot[f[x], {x, 0, 3}, PlotStyle → {RGBColor[1, 0, 0], Thickness[0.01]}]
```



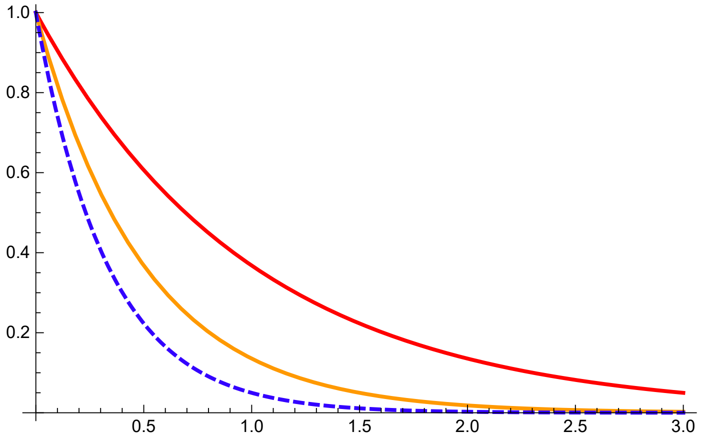
One can combine several plots with the **Show** command

```
Plot1 = Plot[ⅇ^-x, {x, 0, 3}, PlotStyle → {Hue[0], Thick}];
Plot2 = Plot[ⅇ^(-2 x), {x, 0, 3}, PlotStyle → {Hue[0.1], Thick}];
Plot3 = Plot[ⅇ^(-3 x), {x, 0, 3}, PlotStyle → {Hue[0.7], Thick}, PlotRange → All];
Show[Plot1, Plot2, Plot3]
```



Also different curves can be plotted on the same plot

```
Plot[{ⅇ^-x, ⅇ^(-2 x), ⅇ^(-3 x)}, {x, 0, 3},
 PlotStyle → {{Hue[0], Thick}, {Hue[0.1], Thick}, {Hue[0.7], Thick, Dashed}}]
```

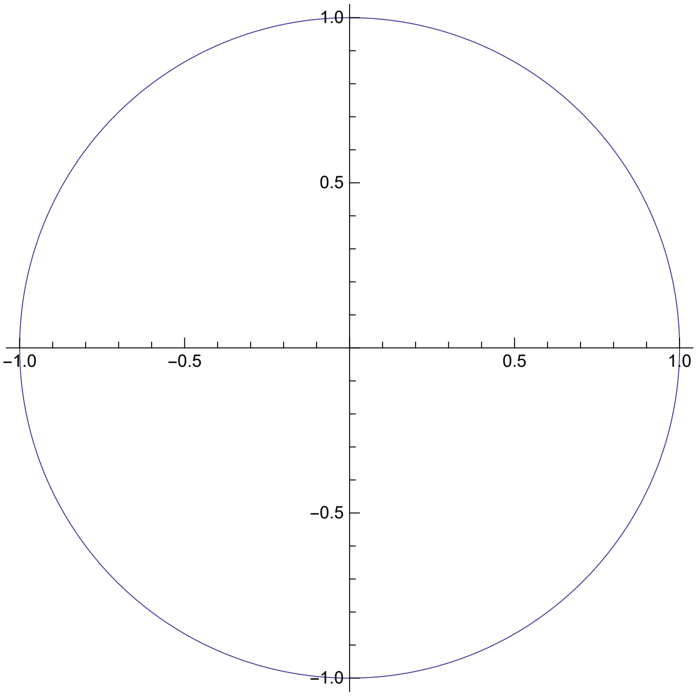


Parametric plots

In some cases it is convenient to represent both the function and its argument in the parametric form as a function of a parameter. An example are multivalued functions that can be plotted only parametrically.

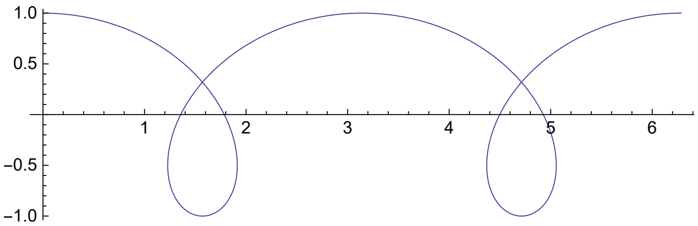
Circle

```
x[ϕ_] := Sin[ϕ];     f[ϕ_] := Cos[ϕ];
ParametricPlot[{x[ϕ], f[ϕ]}, {ϕ, 0, 2 π}] (* Circle *)
```



Cycloide

```
x[ϕ_] := Sin[ϕ] + 1/2 ϕ;               f[ϕ_] := Cos[ϕ];
ParametricPlot[{x[ϕ], f[ϕ]}, {ϕ, 0, 4 π}] (* Cycloide *)
```



List plots

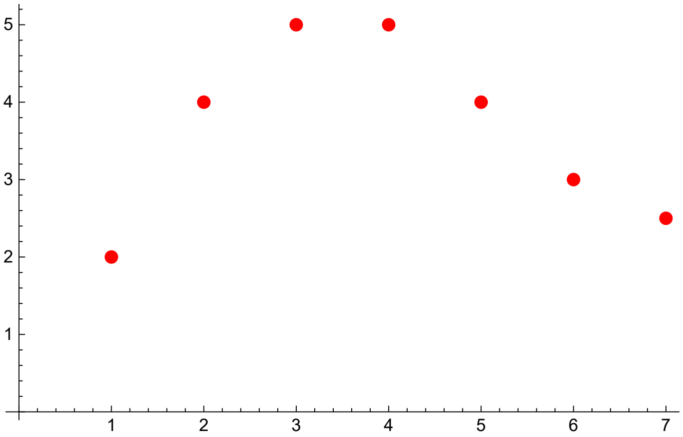
Point data (such as experimental data) stored in Lists of the type

$\{\{x_1, y_1\}, \{x_2, y_2\}, \{x_3, y_3\}, \{x_4, y_4\}, \ldots\}$

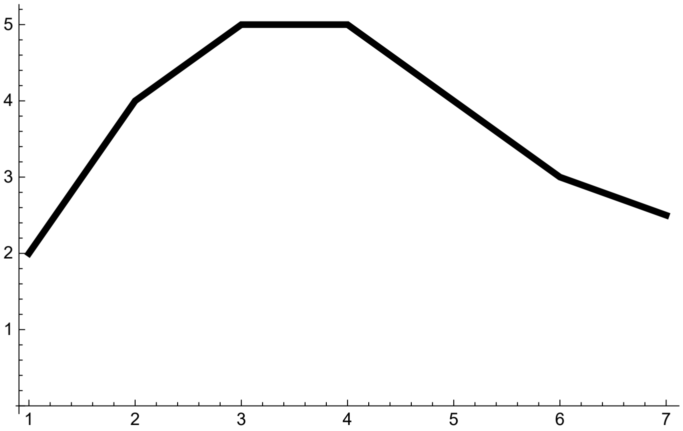
can be plotted with ListPlot

```
In[23]:= DataList = {{1, 2}, {2, 4}, {3, 5}, {4, 5}, {5, 4}, {6, 3}, {7, 2.5}};
         L1 = ListPlot[DataList, PlotStyle → {Red, PointSize[0.02]}]
         L2 =
          ListPlot[DataList, PlotStyle → {Black, Thickness[0.01]}, Joined → True, PlotRange → All]
         Show[L2, L1]
```

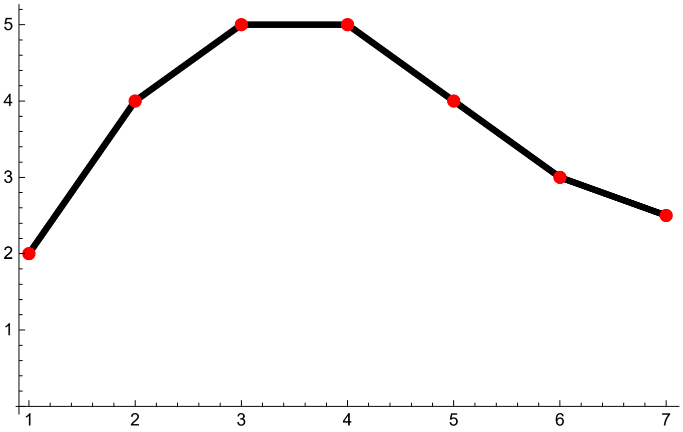
Out[24]=



Out[25]=



Out[26]=

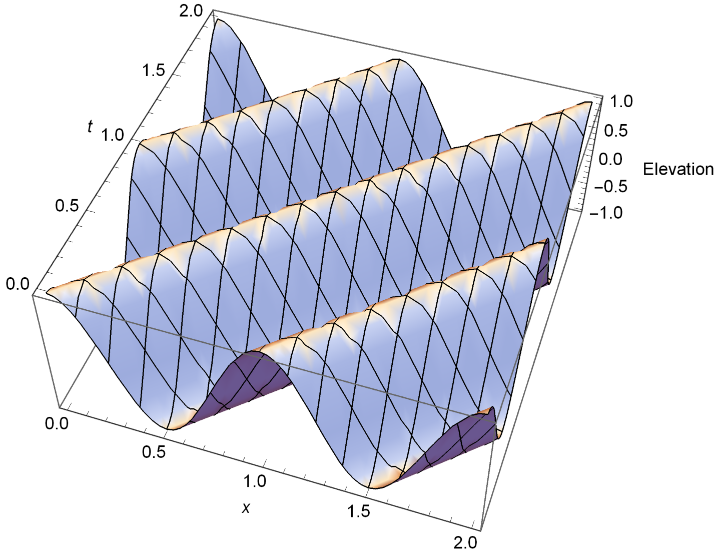


3D plots and Contour plots

3D plots

3d plots are used to represent functions of two variables

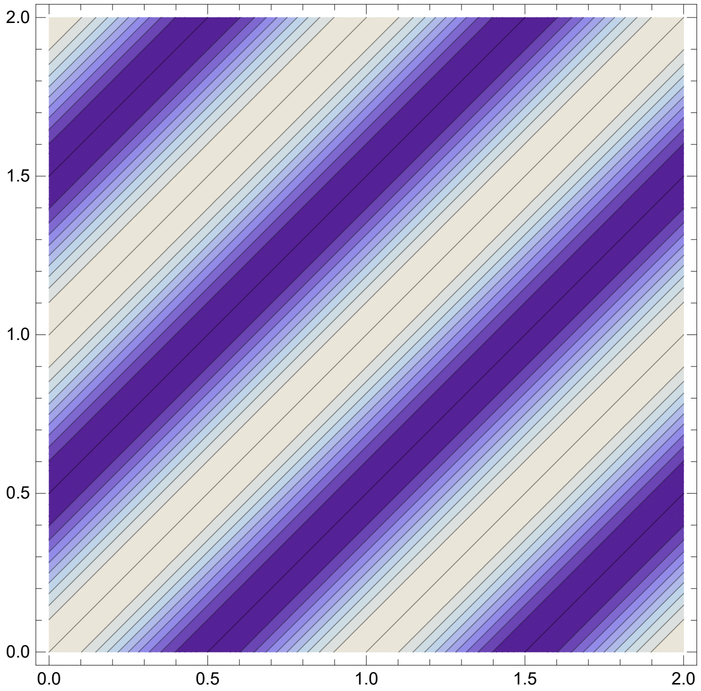
```
(* Plain wave *)
A = T = λ = 1;
fxy[x_, t_] := A Cos[(2 π)/T t - (2 π)/λ x]
Plot3D[fxy[x, t], {x, 0, 2 λ}, {t, 0, 2 T}, AxesLabel → {x, t, "Elevation"}]
```



Contour plots

For the same purpose one can use ContourPlot

```
ContourPlot[fxy[x, t], {x, 0, 2 λ}, {t, 0, 2 T}]
```



Example: Equipotential lines in a system of electric charges

Three charges forming an equilateral triangle

In[69]:=
$$\mathtt{V[x\_,\ y\_]\ :=\ }\frac{1}{\sqrt{x^2+y^2}};$$

$$x_1 = 0;\quad y_1 = \frac{1}{\sqrt{3}};$$

$$x_2 = \frac{-1}{2};\quad y_2 = -\frac{1}{2\sqrt{3}};$$

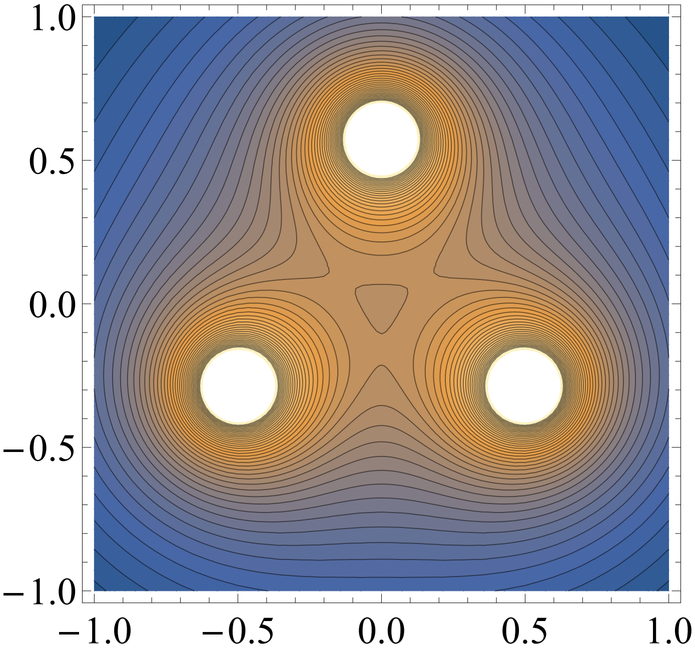
$$x_3 = \frac{1}{2};\quad y_3 = -\frac{1}{2\sqrt{3}};$$

$$\mathtt{F[x\_,\ y\_]} = \mathtt{V}[x - x_1, y - y_1] + \mathtt{V}[x - x_2, y - y_2] + \mathtt{V}[x - x_3, y - y_3]$$

Out[73]=
$$\frac{1}{\sqrt{x^2+\left(-\frac{1}{\sqrt{3}}+y\right)^2}}+\frac{1}{\sqrt{\left(-\frac{1}{2}+x\right)^2+\left(\frac{1}{2\sqrt{3}}+y\right)^2}}+\frac{1}{\sqrt{\left(\frac{1}{2}+x\right)^2+\left(\frac{1}{2\sqrt{3}}+y\right)^2}}$$

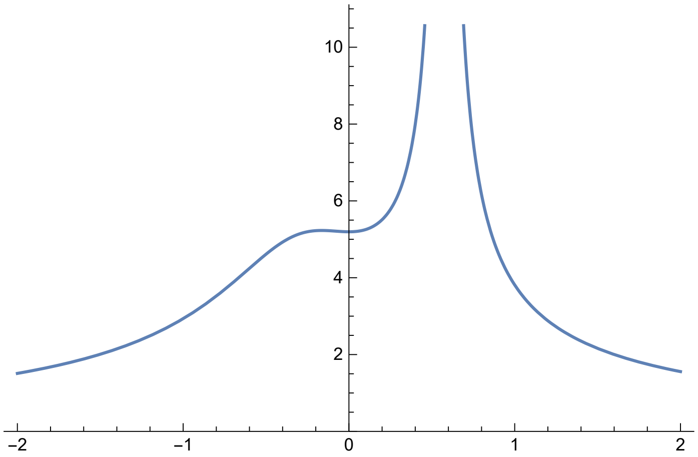
```
In[60]:= ContourPlot[F[x, y], {x, -1, 1}, {y, -1, 1}, Contours → 40,
          BaseStyle → {FontSize → 20, FontFamily → "Times", Plain}]
```

Out[60]=



```
In[63]:= Plot[F[0, y], {y, -2, 2}, PlotRange → {0, Automatic}]
```

Out[63]=

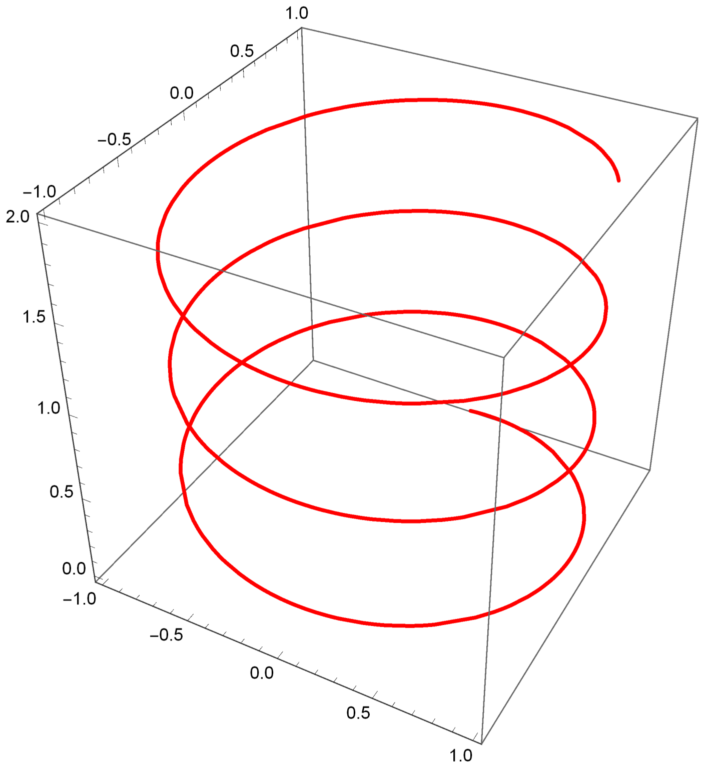


3D parametric plots

In more complicated cases one can use parametric plots. Curve in a 3d space can be represented with the help of one parameter.

Helix

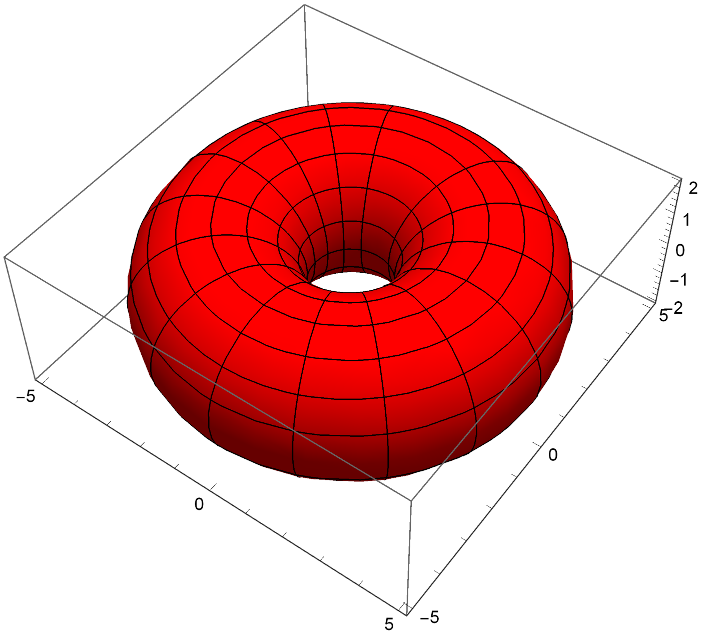
```
x[u_] := Sin[u];
y[u_] := Cos[u];
z[u_] := u / 10;
ParametricPlot3D[{x[u], y[u], z[u]}, {u, 0, 20}, PlotStyle → {Thick, Red}]
```



Surface in a 3d space can be represented with the help of two parameters.

Torus

```
R = 3;  r = 2;
x[u_, v_] := (R + r Cos[v]) Sin[u];
y[u_, v_] := (R + r Cos[v]) Cos[u];
z[u_, v_] := r Sin[v];
ParametricPlot3D[{x[u, v], y[u, v], z[u, v]}, {u, 0, 2 Pi}, {v, 0, 2 Pi}, PlotStyle → {Red}]
```



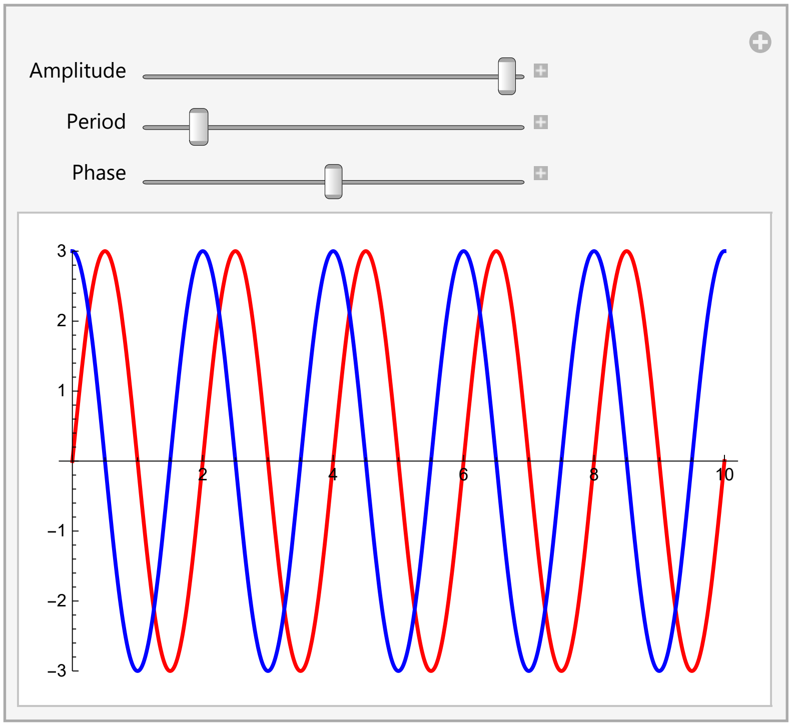
Manipulation and animation

Computers transcend the limitations of printed texts by making images and plots moving. The Manipulation command allows to change the plot (or other types of output) by moving controls on the screen and thus changing parameters on which plot depends

In[37]:=

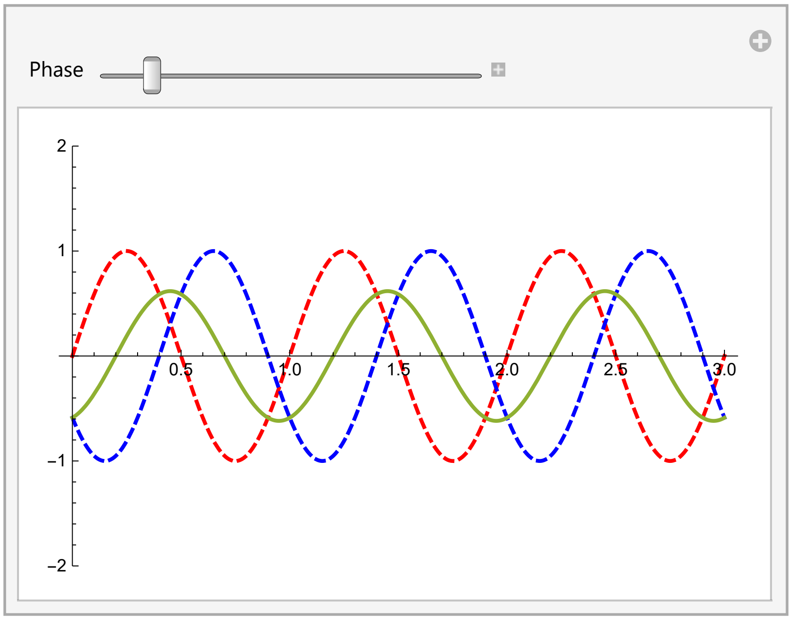
```
(* Harmonic oscillations *)
AMax = 3;
Manipulate[Plot[{A Sin[2 π/T t + ϕ0], A Cos[2 π/T t + ϕ0]}, {t, 0, 10},
  PlotRange → {-AMax, AMax}, PlotStyle → {{Thick, Red}, {Thick, Blue}}],
 {{A, AMax, "Amplitude"}, 0, AMax}, {{T, 2, "Period"}, 1, 10}, {{ϕ0, 0, "Phase"}, -10, 10}]
```

Out[38]=



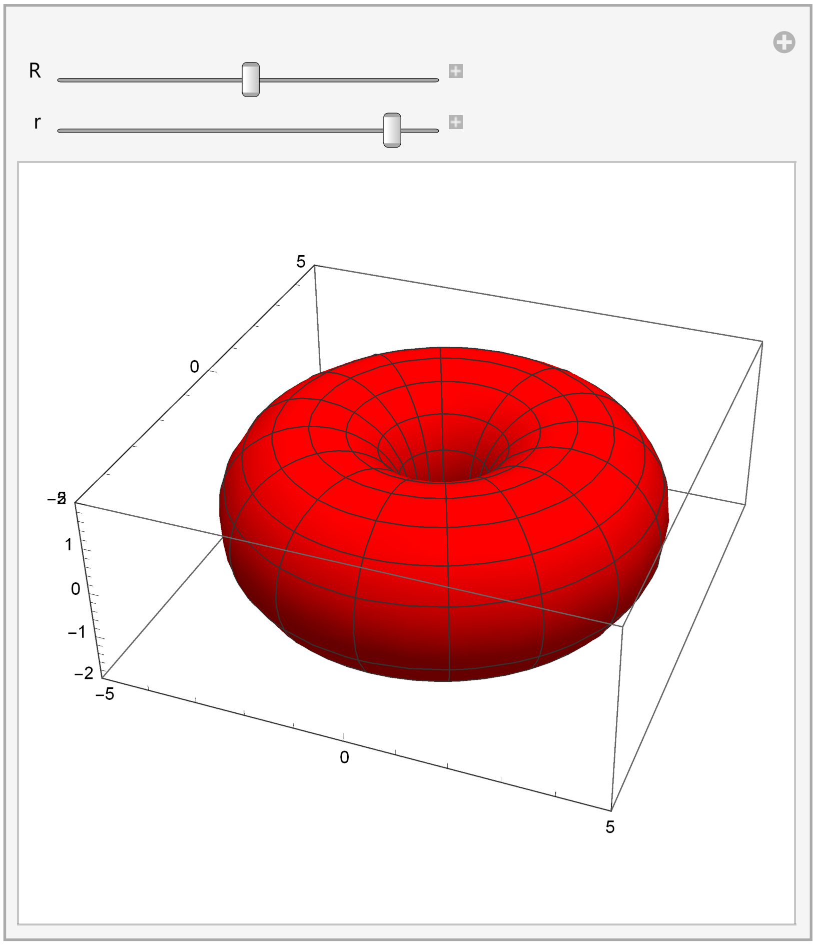
In[27]:=
```
(* Interference of Harmonic oscillations *)
T = 1;
Q1[t_] := Sin[2 π/T t];
Q2[t_, ϕ0_] := Sin[2 π/T t + ϕ0];
Manipulate[Plot[{Q1[t], Q2[t, ϕ0], Q1[t] + Q2[t, ϕ0]}, {t, 0, 3}, PlotRange → {-2, 2},
  PlotStyle → {{Thick, Red, Dashed}, {Thick, Blue, Dashed}, {Thick}}],
 {{ϕ0, -π, "Phase"}, -π, π}]
```

Out[30]=



```
(* Torus *)
RMax = 5;  rMax = 2;
x[u_, v_, R_, r_] := (R + r Cos[v]) Sin[u];
y[u_, v_, R_, r_] := (R + r Cos[v]) Cos[u];
z[u_, v_, R_, r_] := r Sin[v];
Manipulate[
 ParametricPlot3D[{x[u, v, R, r], y[u, v, R, r], z[u, v, R, r]}, {u, 0, 2 Pi}, {v, 0, 2 Pi},
  PlotStyle → {Red}, PlotRange → {{-RMax, RMax}, {-RMax, RMax}, {-rMax, rMax}}],
 {{R, 3}, 0, RMax}, {{r, 1}, 0, 2}]
```

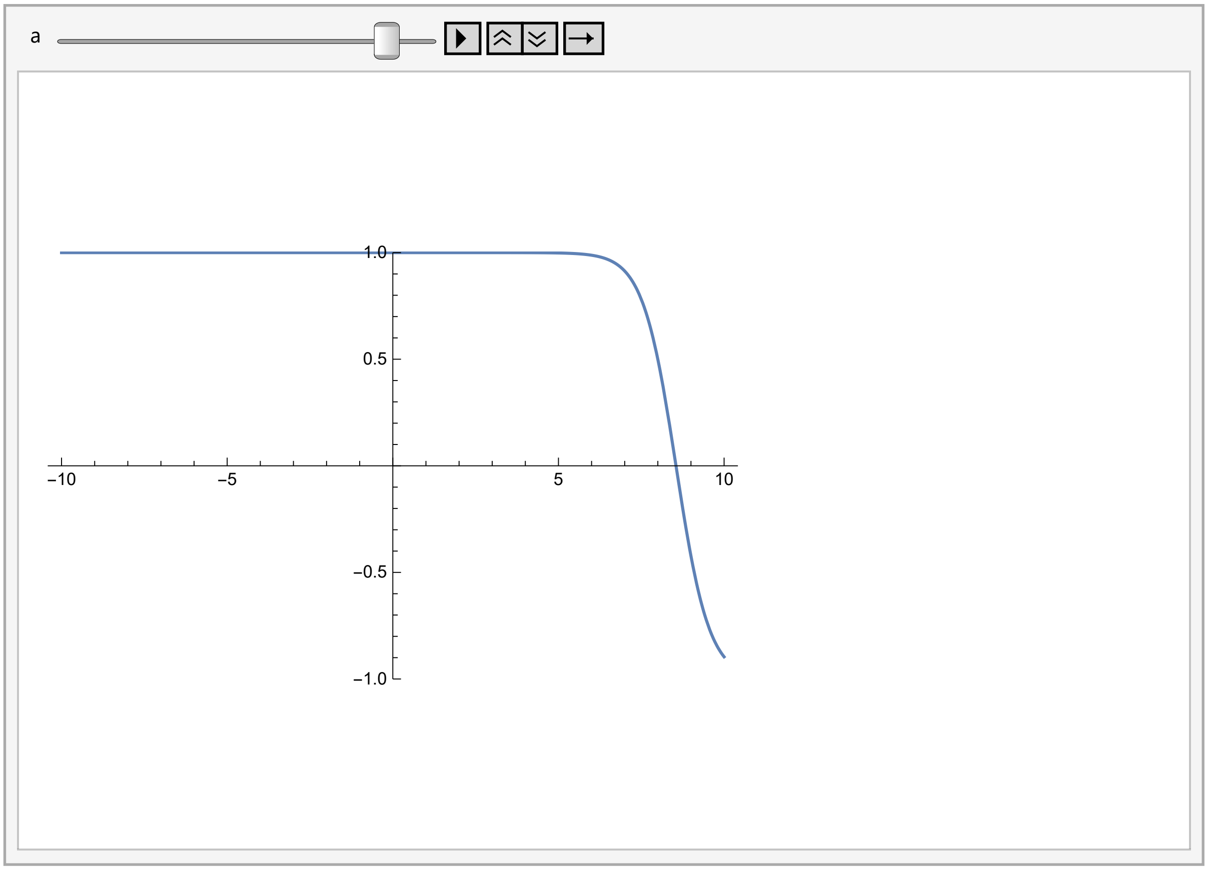
Out[35]=



Animate is self-operating and more powerful than Manipulate but it is limited to one parameter.

In[36]:=
```
Animate[Plot[-Tanh[x - a], {x, -10, 10}, PlotRange → {-1, 1}],
 {a, -10, 10}, AnimationRunning → False]
```

Out[36]=


a
1.0
0.5
-0.5
-1.0
-10
-5
5
10