Lecture Series : Tools for Physicists / Werkzeuge für Physiker 2023

Organiser: Dr. Peter-Bernd Otte

Short Tour of *Mathematica*

Dr.Michael Distler, KPH
Mainz, 7 June 2023

Content :

Interactive Usage

You can use *Mathematica* just like a calculator: you type in questions, and *Mathematica* prints back answers.

Ask *Mathematica* what 3 + 5 is; it prints back 8. *Mathematica* adds the In and Out labels; you do not type them. You end each line with SHIFT+ENTER.

```
In[1]:= 3 + 5
Out[1]= 8
```

The ^ stands for “to the power of”.

```
In[2]:= 57.1^100
Out[2]= 4.60904 × 10^175
```

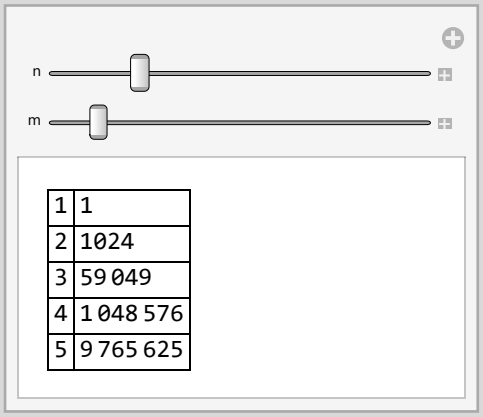
```
In[3]:= 57.1^100
Out[3]= 4.60904 × 10^175
```

Basic Math Assistant and other palettes (Keyboard Entry)
and Assistant Basic Entry Keyboard Math other palettes

Manipulation with simple calculations

In[4]:=
```
Manipulate[Grid[Table[{i, i^m}, {i, 1, n}], Alignment → Left, Frame → All],
 {n, 1, 20, 1}, {m, 1, 100, 1}]
```

Out[4]=



1	1
2	1024
3	59 049
4	1 048 576
5	9 765 625

This asks *Mathematica* to work out the inverse of a 2 x 2 matrix.

In[5]:=
```
Inverse[{{1, 2}, {3, 4}}] // MatrixForm
```

Out[5]//MatrixForm=

$$\begin{pmatrix} -2 & 1 \\ \frac{3}{2} & -\frac{1}{2} \end{pmatrix}$$

Postfix operator //

Mathematica can handle formulas as well as numbers.

This asks *Mathematica* to integrate a simple function.

In[6]:=
```
Integrate[Sqrt[x] Sqrt[x + 1], x]
```

Out[6]=

$$\frac{1}{4}\left(\sqrt{x}\ \sqrt{1+x}\ (1+2x) + \text{Log}\left[-\sqrt{x} + \sqrt{1+x}\right]\right)$$

This asks *Mathematica* to solve a quadratic equation. The result is a list of rules for x convenient for use in other calculations.

In[7]:=
```
Solve[x^2 + x == a, x]
```

Out[7]=

$$\left\{\left\{x \to \frac{1}{2}\left(-1 - \sqrt{1+4a}\right)\right\}, \left\{x \to \frac{1}{2}\left(-1 + \sqrt{1+4a}\right)\right\}\right\}$$

Built - in Functions

The Wolfram Language has nearly **6,000 built - in functions**. All have names in which each word

starts with a capital letter :

In[8]:=
```
Range[20]
```

Out[8]=
```
{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20}
```

In[9]:=
```
NestList[f, x, 5]
```

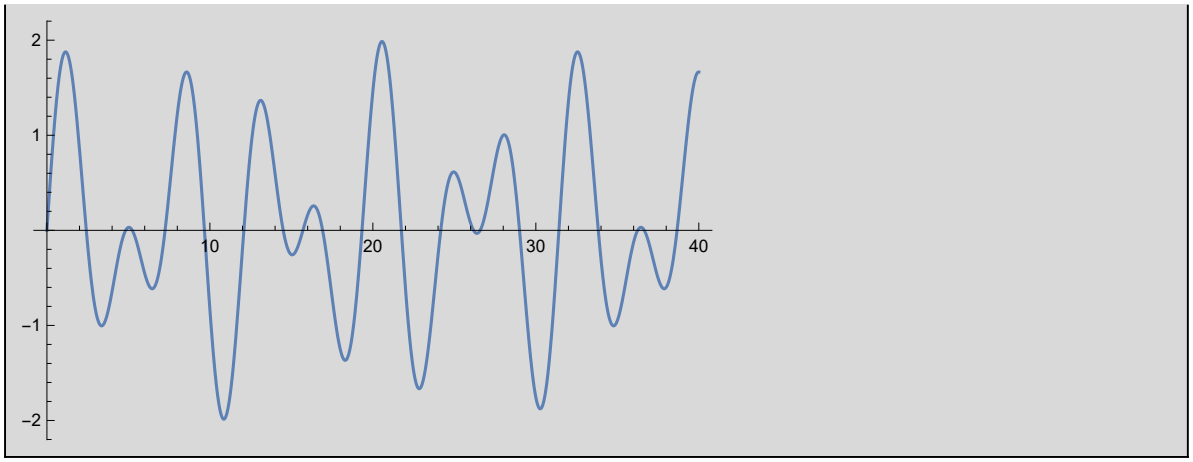
Out[9]=
```
{x, f[x], f[f[x]], f[f[f[x]]], f[f[f[f[x]]]], f[f[f[f[f[x]]]]]}
```

Mathematica can also create two and three-dimensional graphics.

This creates a 2D plot of a simple function.

In[10]:=
```
Plot[Sin[x] + Sin[1.6 x], {x, 0, 40}]
```

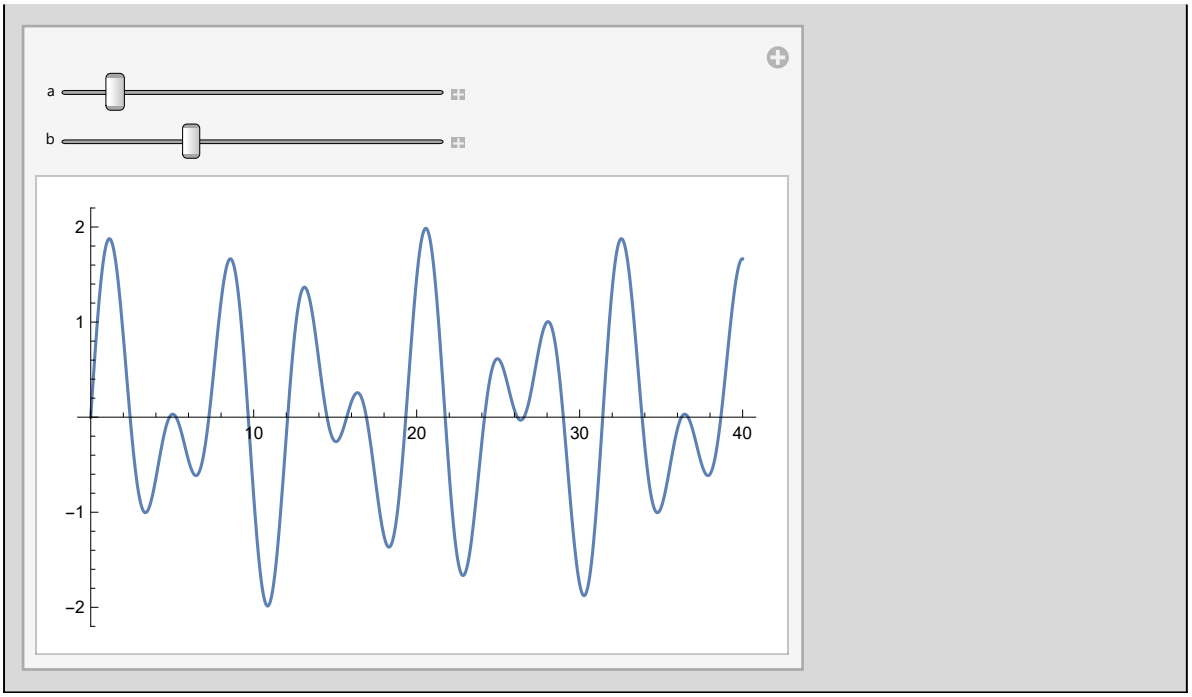
Out[10]=



This creates a simple Manipulation of a 2 D plot.

In[11]:=
```
Manipulate[Plot[Sin[a x] + Sin[b x], {x, 0, 40}],
 {{a, 1}, 0, 10},
 {{b, 1.6}, 0, 5}]
```

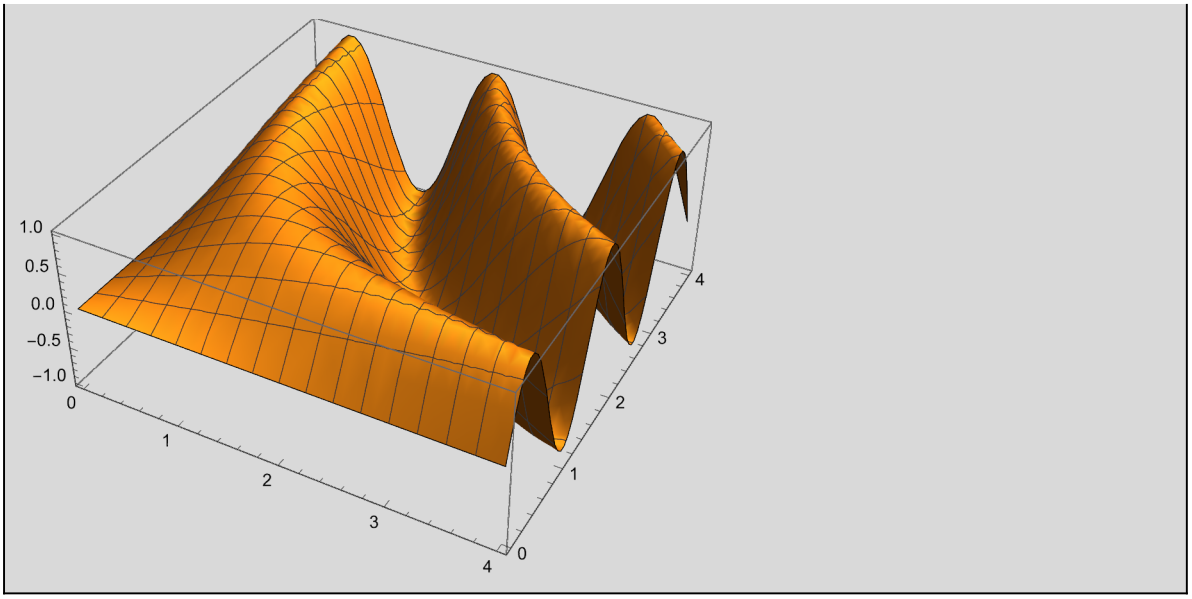
Out[11]=



Here is a 3D plot.

In[12]:=
```
Plot3D[Sin[x y], {x, 0, 4}, {y, 0, 4}]
```

Out[12]=



You can access many of the calculator features of *Mathematica* just by pushing buttons in standard palettes.

Power Computing with *Mathematica*

Even though you can use it as easily as a calculator, *Mathematica* gives you access to immense computational power.

Mathematica can handle numbers of any size.

In[13]:=
```
100!
```
Out[13]=
```
93 326 215 443 944 152 681 699 238 856 266 700 490 715 968 264 381 621 468 592 963 895 217 599 993 ⁖
 229 915 608 941 463 976 156 518 286 253 697 920 827 223 758 251 185 210 916 864 000 000 000 000 ⁖
 000 000 000 000
```

This works out a numerical 100-digit approximation to pi.

In[14]:=
```
N[Pi, 100]
```
Out[14]=
```
3.14159265358979323846264338327950288419716939937510582097494459230781640628620899 ⁖
 86280348253421170 68
```

The Wolfram Language by default does exact computation whenever it can :

In[15]:=
```
3 / 7 + 2 / 11
```
Out[15]=
$$\frac{47}{77}$$

Use N to get (potentially faster) numerical results :

In[16]:=
```
N[3 / 7 + 2 / 11]
```
Out[16]=
```
0.61039
```

Mathematica can work with formulas of any length—solving problems that would have taken years by hand.

This asks *Mathematica* to factor a polynomial.

In[17]:=
```
Factor[x^99 + y^99]
```
Out[17]=
$$\begin{aligned}&(x+y)\left(x^2-x\,y+y^2\right)\left(x^6-x^3\,y^3+y^6\right)\\&\left(x^{10}-x^9\,y+x^8\,y^2-x^7\,y^3+x^6\,y^4-x^5\,y^5+x^4\,y^6-x^3\,y^7+x^2\,y^8-x\,y^9+y^{10}\right)\\&\left(x^{20}+x^{19}\,y-x^{17}\,y^3-x^{16}\,y^4+x^{14}\,y^6+x^{13}\,y^7-x^{11}\,y^9-x^{10}\,y^{10}-x^9\,y^{11}+x^7\,y^{13}+\right.\\&\left.x^6\,y^{14}-x^4\,y^{16}-x^3\,y^{17}+x\,y^{19}+y^{20}\right)\left(x^{60}+x^{57}\,y^3-x^{51}\,y^9-x^{48}\,y^{12}+x^{42}\,y^{18}+\right.\\&\left.x^{39}\,y^{21}-x^{33}\,y^{27}-x^{30}\,y^{30}-x^{27}\,y^{33}+x^{21}\,y^{39}+x^{18}\,y^{42}-x^{12}\,y^{48}-x^9\,y^{51}+x^3\,y^{57}+y^{60}\right)\end{aligned}$$

Mathematica calls on sophisticated algorithms to simplify formulas.

In[18]:=
```
Simplify[%]
```

Out[18]=

$$x^{99} + y^{99}$$

In[ ]:=

Use the Out operator (%, %%, %%%, ...) with care.
It is usually fine to use % (the last result generated) within one cell.
But using % and %% across cells should be avoided :

In[19]:=
```
M = {{1, 2}, {3, 4}};
% // MatrixForm
M =.
```

Out[20]//MatrixForm=

$$\begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}$$

Accessing Algorithms in *Mathematica*

Whenever you use *Mathematica* you are accessing the world's largest collection of computational algorithms.

In each case, *Mathematica* automatically chooses the best algorithm to use.

In[22]:=
```
FindRoot[Cos[x] == x + Log[x], {x, 1}]
```

Out[22]=

```
{x → 0.840619}
```

In[23]:=
```
NIntegrate[Log[x + Sin[x]], {x, 0, 2}]
```

Out[23]=

```
0.555889
```

In[24]:=
```
NSolve[x^5 - 6 x^3 + 8 x + 1 == 0, x]
```

Out[24]=

```
{{x → -2.05411}, {x → -1.2915}, {x → -0.126515}, {x → 1.55053}, {x → 1.9216}}
```

In[25]:=
```
NMinimize[{Cos[x y] + x, x^2 + y^2 ≤ 10}, {x, y}]
```

Out[25]=

```
{-3.99011, {x → -2.99811, y → -1.00565}}
```

This generates a two-dimensional table corresponding to a matrix.

In[26]:=
```
m = Table[2^i + x^j, {i, 3}, {j, 4}]
```
Out[26]=
$$\left\{\left\{2+x,\ 2+x^2,\ 2+x^3,\ 2+x^4\right\},\ \left\{4+x,\ 4+x^2,\ 4+x^3,\ 4+x^4\right\},\ \left\{8+x,\ 8+x^2,\ 8+x^3,\ 8+x^4\right\}\right\}$$

This displays the table in matrix form.

In[27]:=
```
MatrixForm[m]
```
Out[27]//MatrixForm=
$$\begin{pmatrix} 2+x & 2+x^2 & 2+x^3 & 2+x^4 \\ 4+x & 4+x^2 & 4+x^3 & 4+x^4 \\ 8+x & 8+x^2 & 8+x^3 & 8+x^4 \end{pmatrix}$$

In[•]:=
Clear the assignment of a variable as soon as it is not needed any longer. Especially the letters that are used in symbolic expression should not be used as global variables.

In[28]:=
```
m =.
```

Mathematical Knowledge in *Mathematica*

Mathematica incorporates the knowledge from the world's mathematical handbooks—and uses its own revolutionary algorithms to go much further.

Mathematica knows about all the hundreds of special functions in pure and applied mathematics.

In[29]:=
```
LegendreQ[3, x]
```
Out[29]=
$$\frac{2}{3}-\frac{5x^2}{2}-\frac{1}{2}x\left(3-5x^2\right)\left(-\frac{1}{2}\operatorname{Log}[1-x]+\frac{1}{2}\operatorname{Log}[1+x]\right)$$

Mathematica can evaluate special functions with any parameters to any precision.

In[30]:=
```
N[MathieuC[1 + i, 2 i, 3], 50]
```
Out[30]=
$$3.9251311374125198643497646168158379203627176844794 + 1.8988239115433472411052747971439115776785813553761\,i$$

Mathematica is now able to do vastly more integrals than were ever before possible for either humans or computers.

In[31]:=
```
Integrate[Sqrt[x] ArcTan[x], x]
```

Out[31]=

$$\frac{1}{6}\left(-8\sqrt{x}-2\sqrt{2}\,\text{ArcTan}\left[1-\sqrt{2}\,\sqrt{x}\right]+2\sqrt{2}\,\text{ArcTan}\left[1+\sqrt{2}\,\sqrt{x}\right]+4x^{3/2}\,\text{ArcTan}[x]-\sqrt{2}\,\text{Log}\left[1-\sqrt{2}\,\sqrt{x}+x\right]+\sqrt{2}\,\text{Log}\left[1+\sqrt{2}\,\sqrt{x}+x\right]\right)$$

Here is a definite integral. The results often require special functions.

In[32]:=
```
Integrate[Log[x] Exp[-x^3], {x, 0, Infinity}]
```

Out[32]=

$$\frac{1}{81}\,\text{Gamma}\left[-\frac{2}{3}\right]\left(6\,\text{EulerGamma}+\sqrt{3}\,\pi+9\,\text{Log}[3]\right)$$

Here is a symbolic sum.

In[33]:=
```
Sum[1 / (k + 1)^6, {k, 0, n}]
```

Out[33]=

$$\frac{8\pi^6-63\,\text{PolyGamma}[5,\,2+n]}{7560}$$

Mathematica can solve a wide range of ordinary and partial differential equations.

In[34]:=
```
DSolve[y''[x] + y'[x] + x y[x] == 0, y[x], x]
```

Out[34]=

$$\left\{\left\{y[x]\to e^{-x/2}\,\text{AiryAi}\left[-(-1)^{1/3}\left(\frac{1}{4}-x\right)\right]C_1+e^{-x/2}\,\text{AiryBi}\left[-(-1)^{1/3}\left(\frac{1}{4}-x\right)\right]C_2\right\}\right\}$$

This finds the billionth prime number.

In[35]:=
```
Prime[10^9]
```

Out[35]=

```
22 801 763 489
```

Visualization with *Mathematica*

Mathematica makes it easy to create stunning visual images.

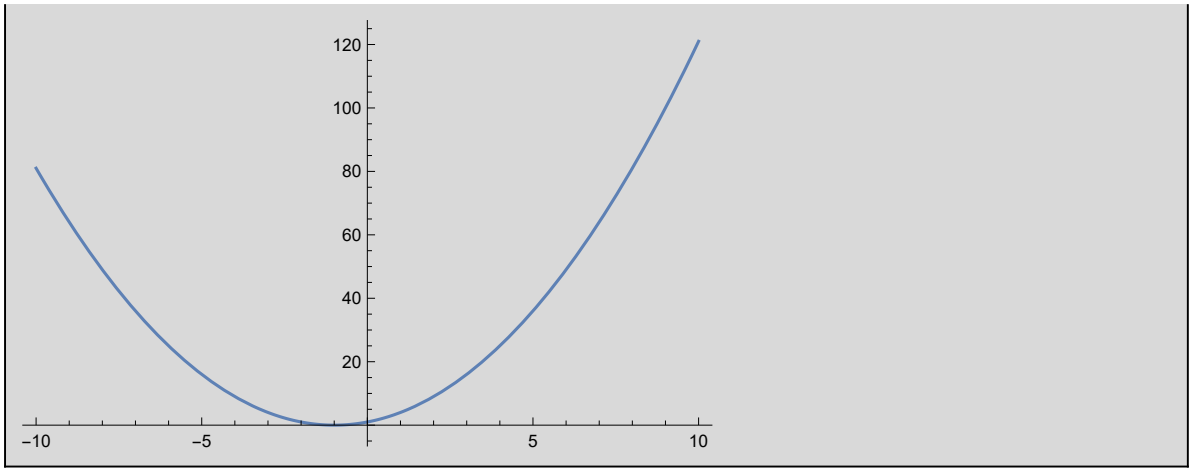
Plots in 2D

Generate a 2 D plot of a polynomial function : (The interval notation of {x, min, max} defines the

domain.)

In[36]:=
```
Plot[x^2 + 2 x + 1, {x, -10, 10}]
```

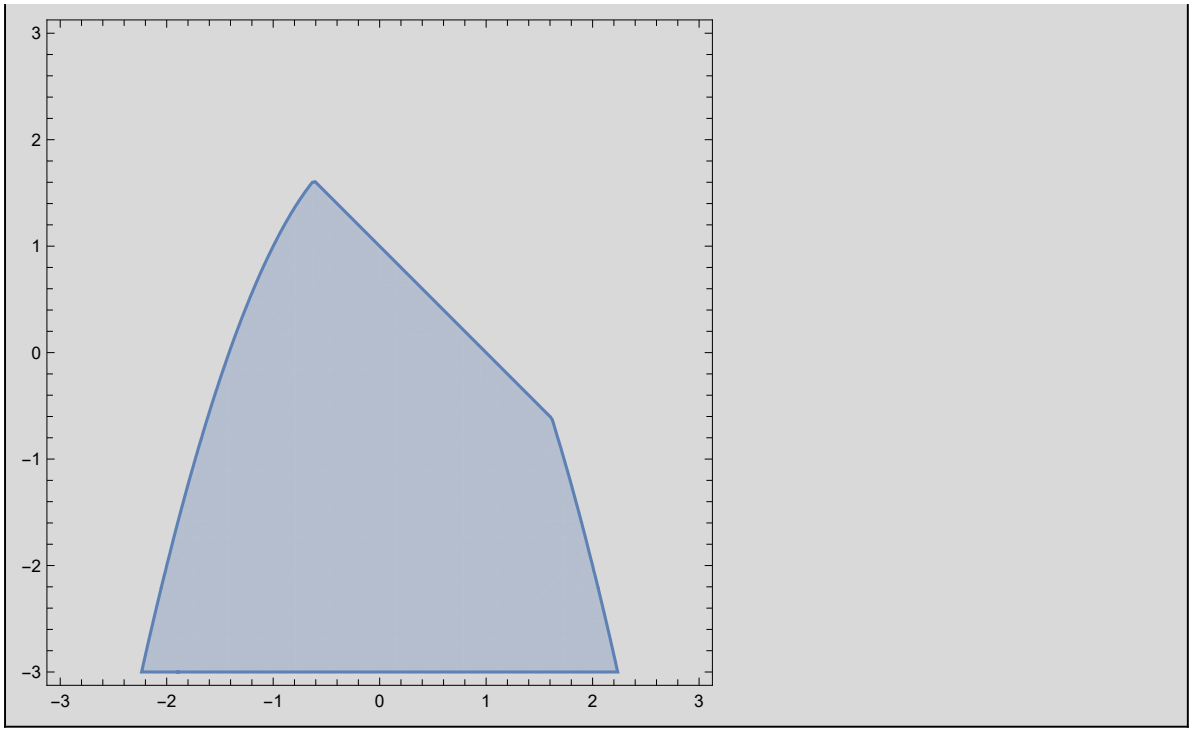
Out[36]=



Or plot a 2 D region for a set of inequalities : (&& is the symbol for And.)

In[37]:=
```
RegionPlot[Reduce[{x^2 + y < 2 && x + y < 1}], {x, -3, 3}, {y, -3, 3}]
```

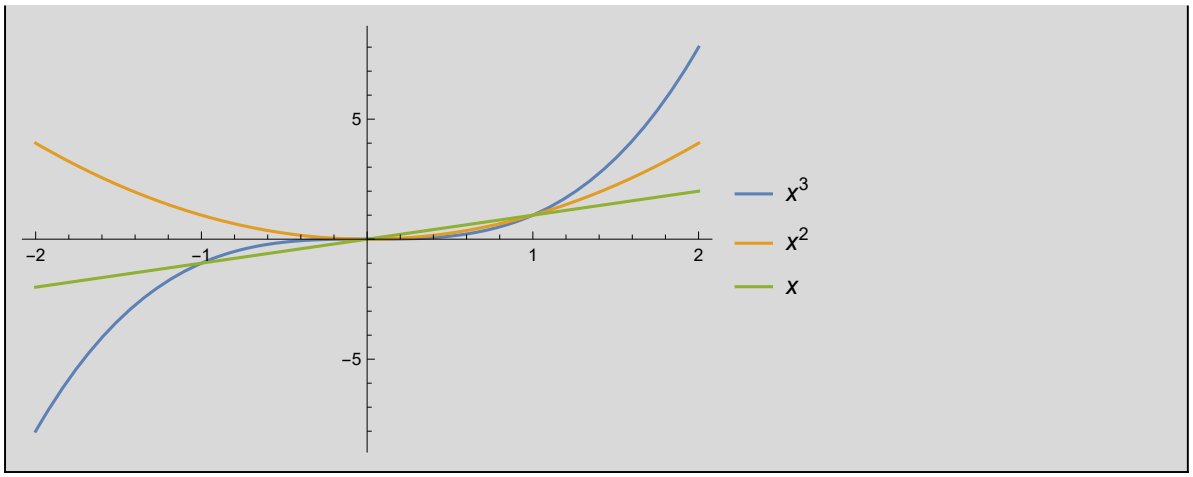
Out[37]=



There are lots of useful options to customize visualizations, like adding legends :

In[38]:=
```
Plot[{x^3, x^2, x}, {x, -2, 2}, PlotLegends → "Expressions"]
```

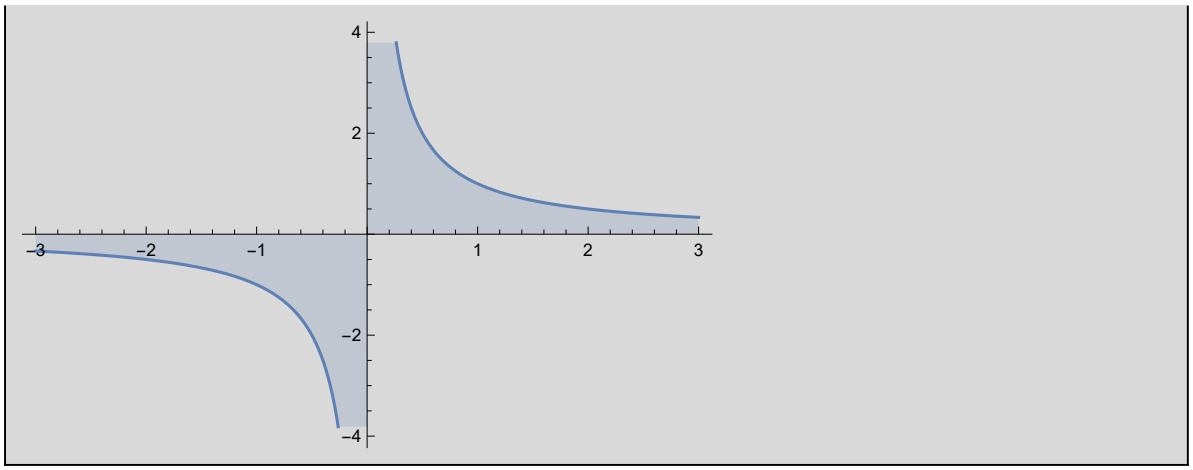
Out[38]=



Or filling a plot to visualize the area under a curve :

In[39]:=
```
Plot[1 / x, {x, -3, 3}, Filling → Axis]
```

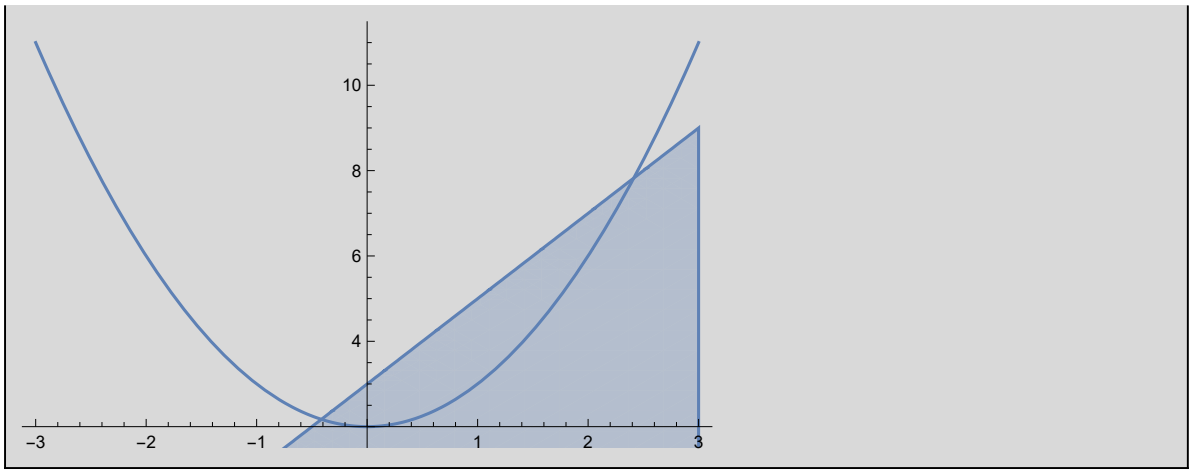
Out[39]=



Combine different plot types with Show :

In[40]:=
```
Show[{Plot[x^2 + 2, {x, -3, 3}], RegionPlot[2 x > y - 3, {x, -3, 3}, {y, 0, 9}]}]
```

Out[40]=

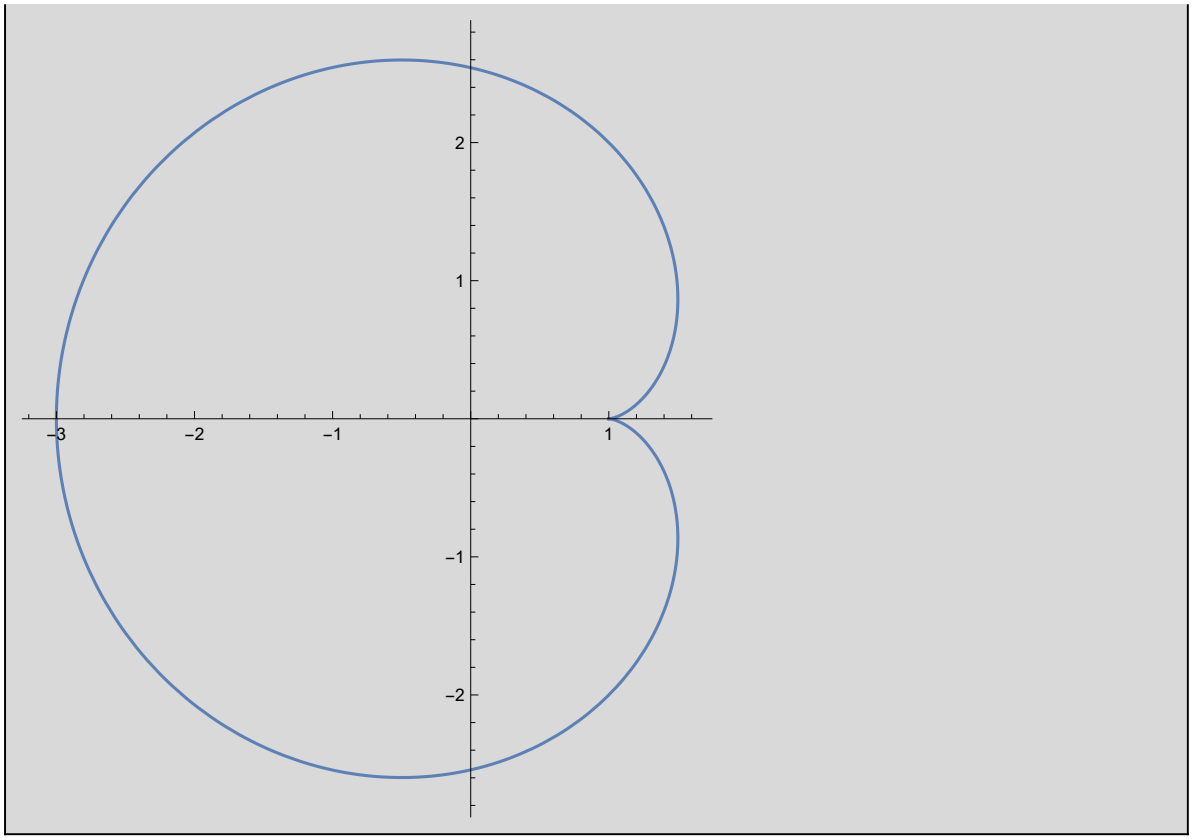


More Plots in 2 D

Use ParametricPlot to plot a set of parametric equations :

In[41]:=
```
ParametricPlot[{2 Cos[t] - Cos[2 t], 2 Sin[t] - Sin[2 t]}, {t, 0, 2 Pi}]
```

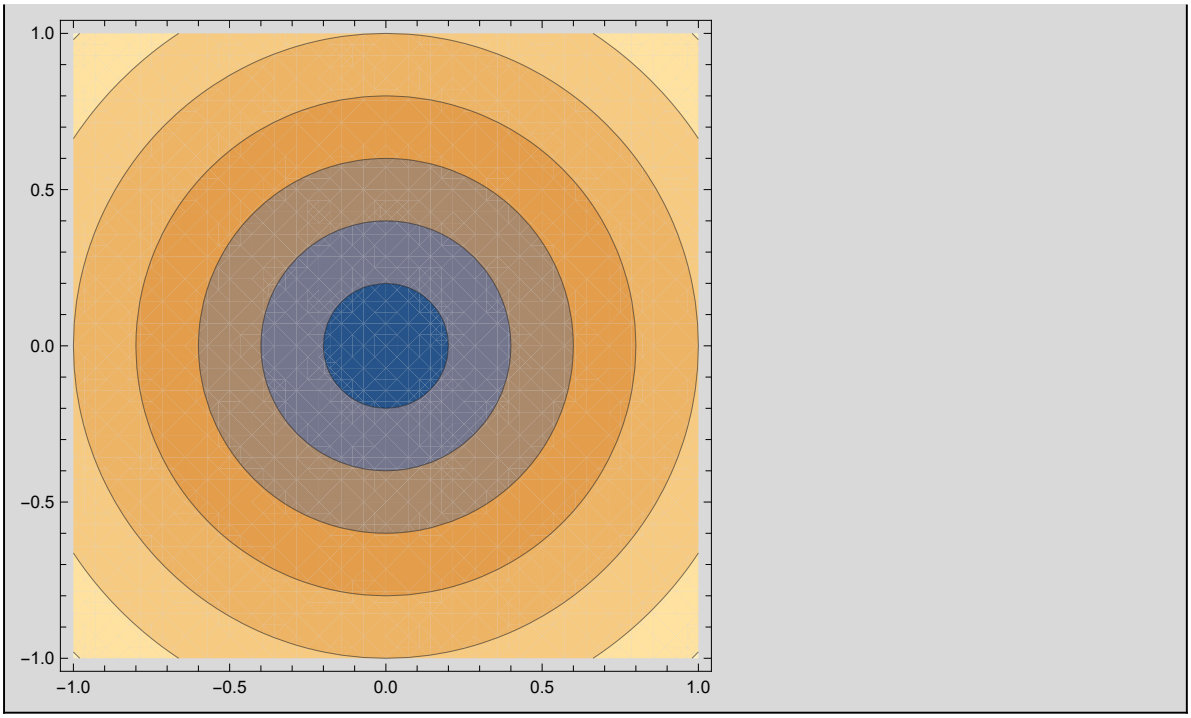
Out[41]=



Make a ContourPlot of a function in multiple variables :

In[42]:=
```
ContourPlot[Sqrt[x^2 + y^2], {x, -1, 1}, {y, -1, 1}]
```

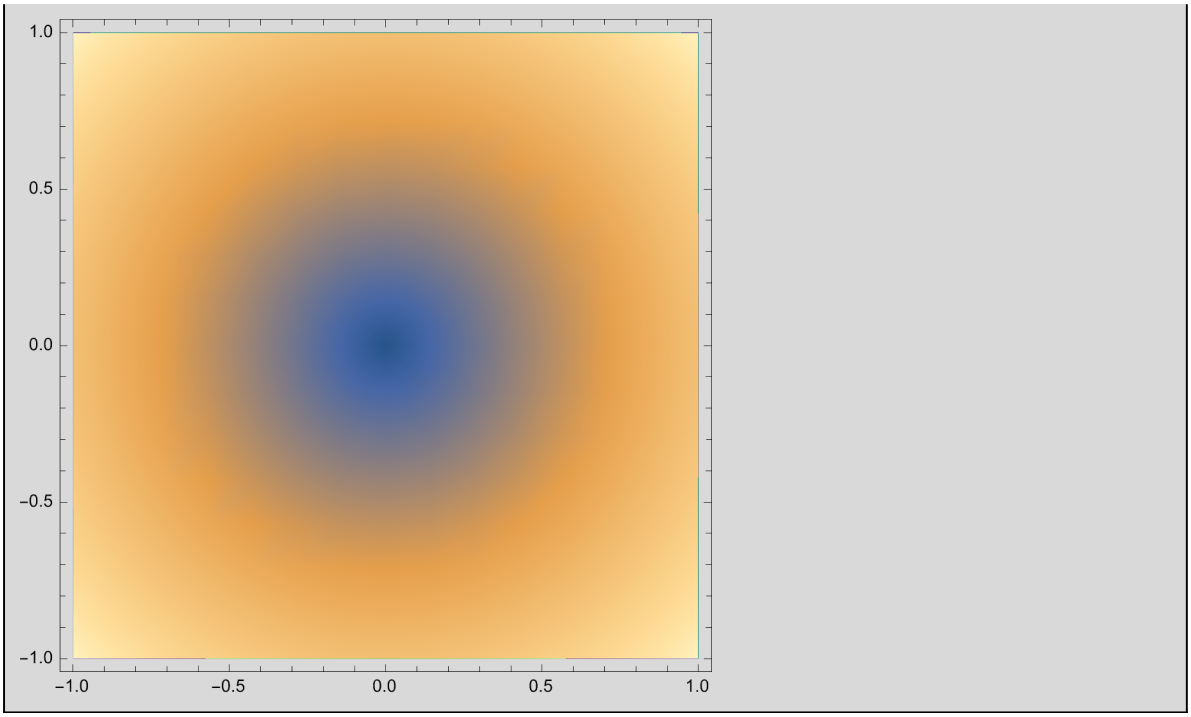
Out[42]=



Use DensityPlot for a continuous version :

In[43]:=
```
DensityPlot[Sqrt[x^2 + y^2], {x, -1, 1}, {y, -1, 1}]
```

Out[43]=

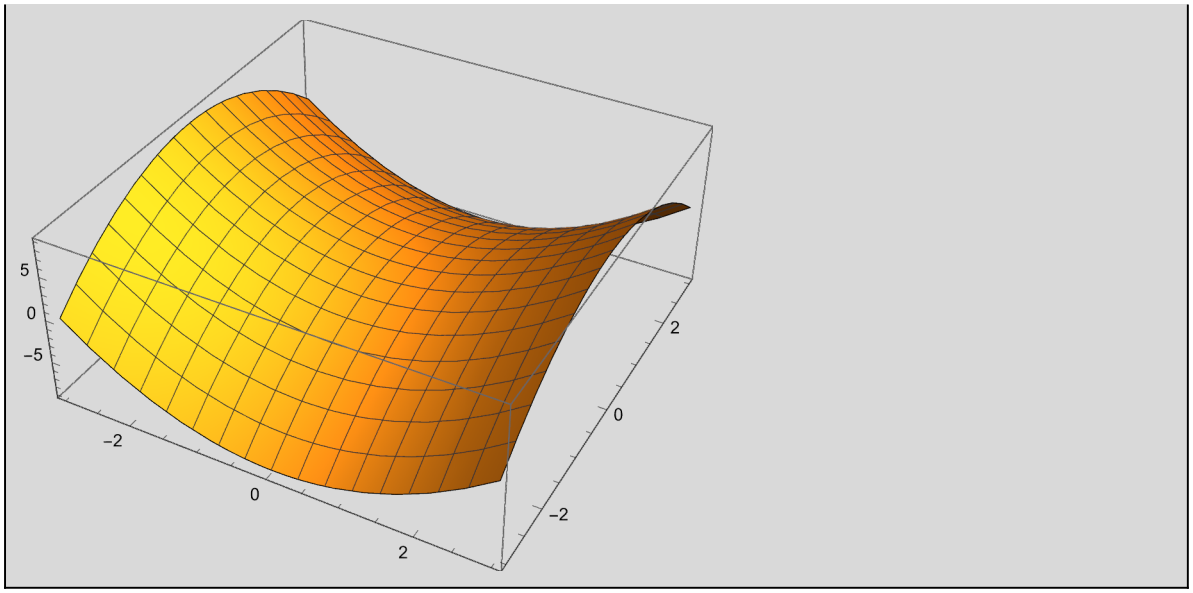


Plots in 3 D

Plot3D will plot a 3 D Cartesian curve or surface :

In[44]:=
```
Plot3D[x^2 - y^2, {x, -3, 3}, {y, -3, 3}]
```

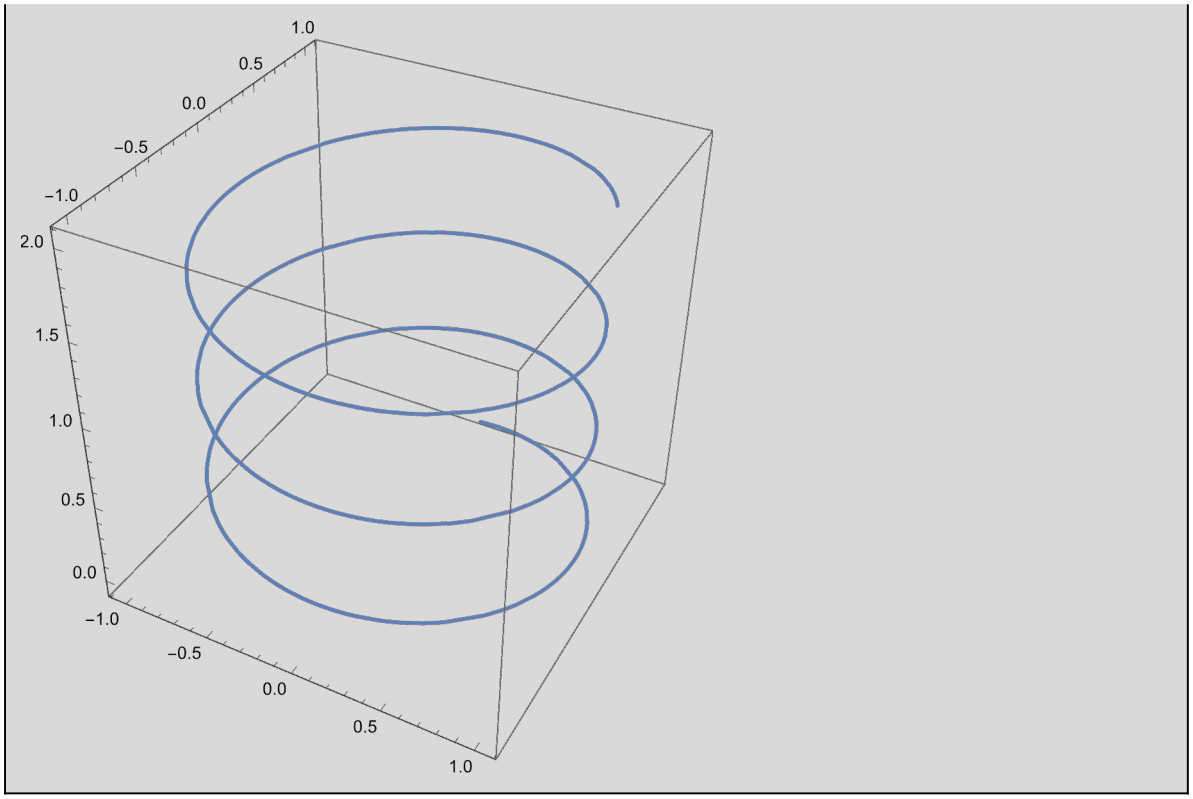
Out[44]=



Use ParametricPlot3D to plot a 3 D space curve :

In[45]:=
```
ParametricPlot3D[{Sin[u], Cos[u], u / 10}, {u, 0, 20}]
```

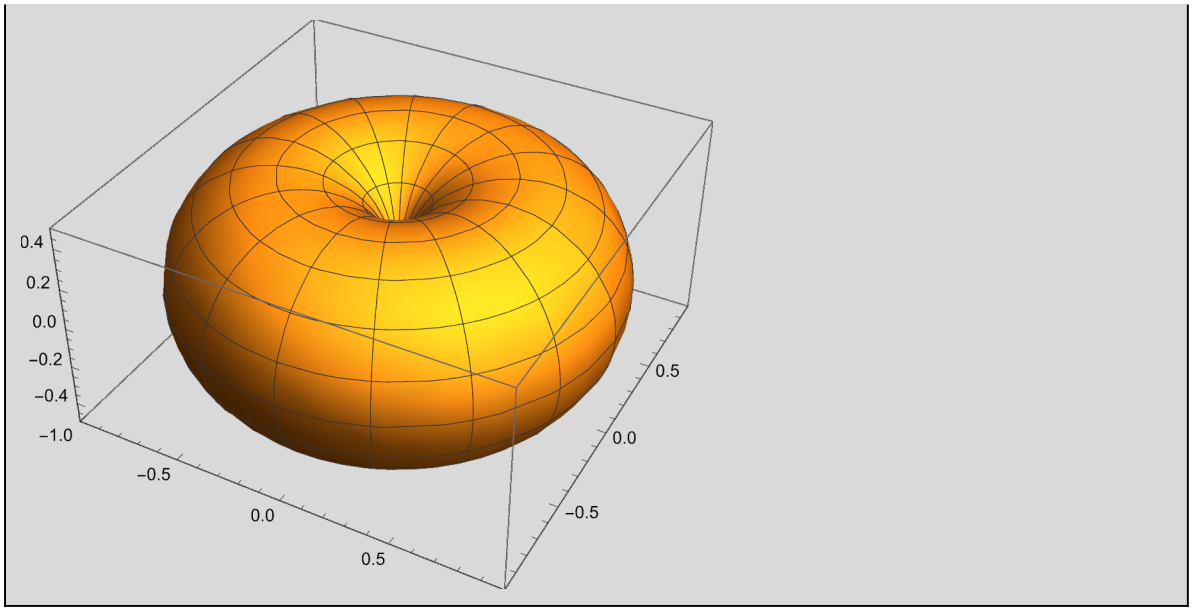
Out[45]=



For plotting in spherical coordinates, use SphericalPlot3D :

In[46]:=
```
SphericalPlot3D[Sin[θ], {θ, 0, Pi}, {ϕ, 0, 2 Pi}]
```

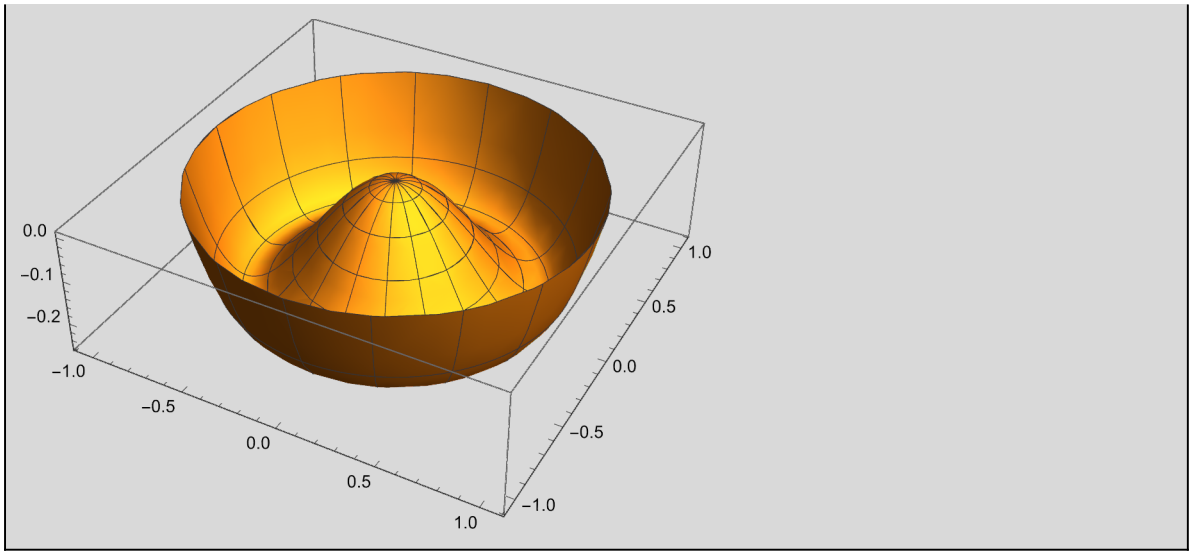
Out[46]=



RevolutionPlot3D constructs the surface formed by revolving an expression around an axis :

In[47]:=
```
RevolutionPlot3D[x^4 - x^2, {x, -1, 1}]
```

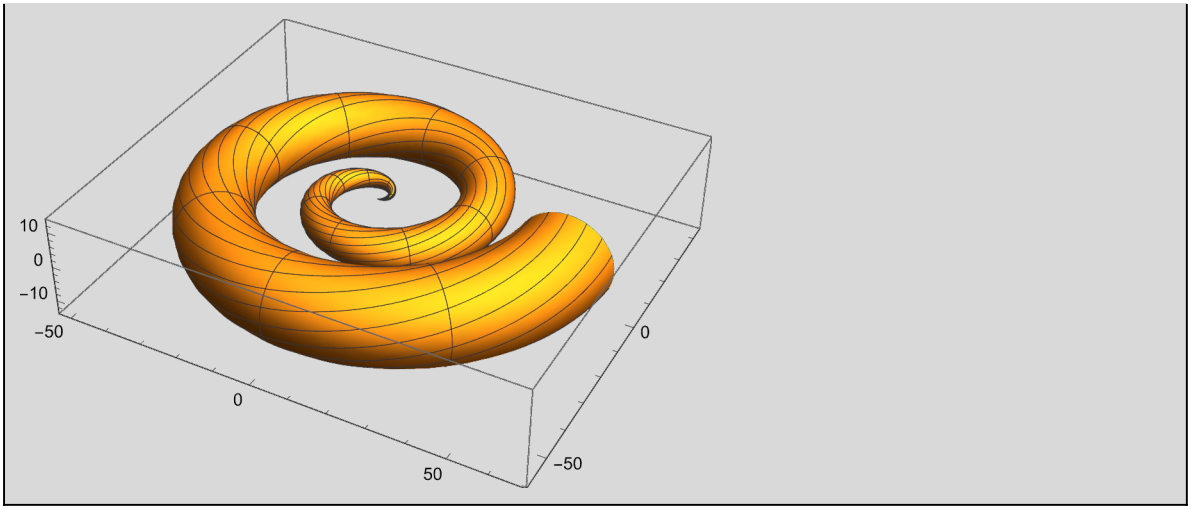
Out[47]=



This creates a 3D parametric plot with automatic choices for most options.

In[48]:=
```
ParametricPlot3D[{u Cos[u] (4 + Cos[v + u]), u Sin[u] (4 + Cos[v + u]), u Sin[v + u]},
 {u, 0, 4 Pi}, {v, 0, 2 Pi}, PlotPoints → {60, 12}]
```

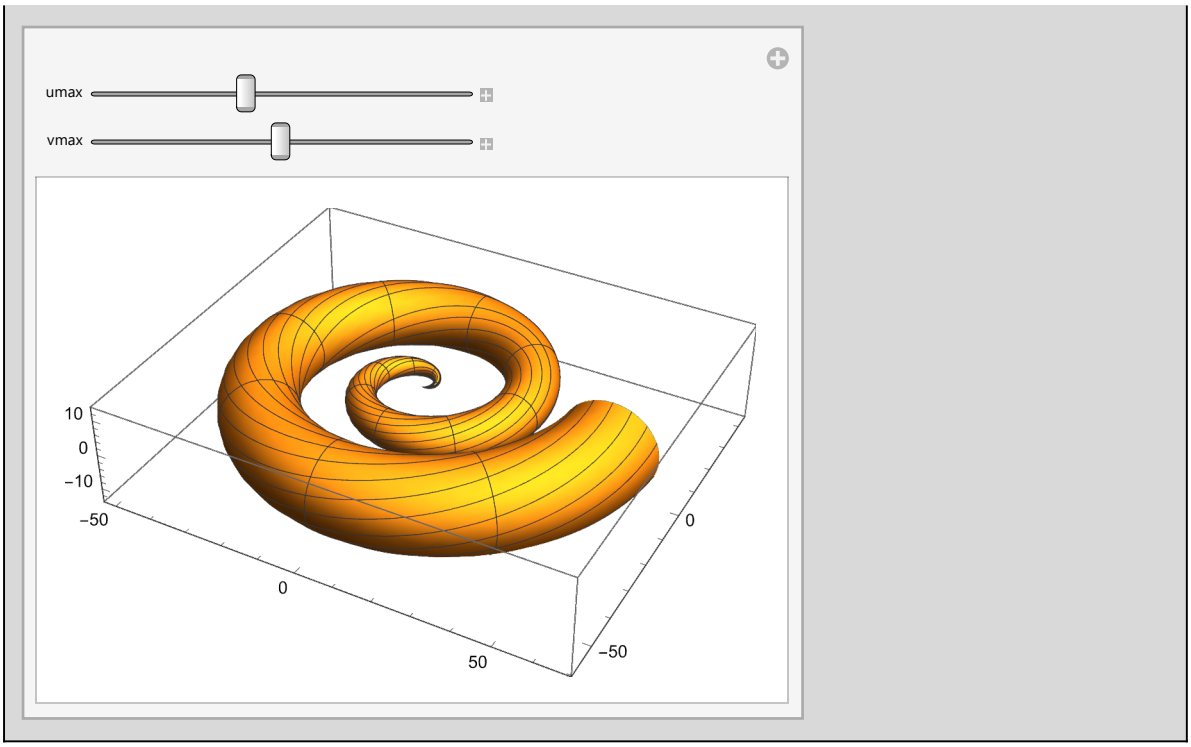
Out[48]=



This shows how Manipulate can be used to visualize a 3D plot

In[49]:=
```
Manipulate[
 ParametricPlot3D[{u Cos[u] (4 + Cos[v + u]), u Sin[u] (4 + Cos[v + u]), u Sin[v + u]},
  {u, 0, umax}, {v, 0, vmax}, PlotPoints → {60, 12}],
 {{umax, 4 π}, 0, 10 π},
 {{vmax, 2 π}, 0, 4 π}]
```

Out[49]=



Mathematical Notation

Mathematica notebooks fully support standard mathematical notation—for both output and input.

Mathematica combines the compactness of mathematical notation with the precision of a computer language.

Here is an integral entered using only ordinary keyboard characters.

In[50]:=
```
Integrate[Log[1 + x] / Sqrt[x], x]
```

Out[50]=

$$4\,\text{ArcTan}\left[\sqrt{x}\right] + 2\sqrt{x}\,\left(-2 + \text{Log}[1 + x]\right)$$

Here is the same integral entered in 2D form with special characters. You can enter this form using a palette or directly from the keyboard.

In[51]:=
$$\int \frac{\text{Log}[1+\xi]}{\sqrt{\xi}}\,d\xi$$

Out[51]=

$$4\,\text{ArcTan}\left[\sqrt{\xi}\right] + 2\sqrt{\xi}\,\left(-2 + \text{Log}[1 + \xi]\right)$$

This shows the keys you need to type to get the input above. The symbol ⎋ stands for the ESC key.

```
⎋int⎋ Log[1 + ⎋x⎋] CTRL[ / ] CTRL[2] ⎋x⎋ CTRL[ ] CTRL[ ] ⎋dd⎋ ⎋x⎋
```

In[52]:=
$$\int \frac{\text{Log}[1+\xi]}{\sqrt{\xi}}\,d\xi$$

Out[52]=

$$4\,\text{ArcTan}\left[\sqrt{\xi}\right] + 2\sqrt{\xi}\,\left(-2 + \text{Log}[1 + \xi]\right)$$

Mathematica's StandardForm is precise and unambiguous. TraditionalForm requires heuristics for interpretation.

Mathematica can generate output in traditional textbook form.

In[53]:=
```
TraditionalForm[%]
```

Out[53]//TraditionalForm=

$$2\sqrt{\xi}\,(\log(\xi+1)-2)+4\tan^{-1}\left(\sqrt{\xi}\right)$$

Mathematica produces top-quality output for formulas of any size or complexity.

In[54]:=
```
Sum[φ^μ Exp[π μ/4]/(μ!^2 (μ^2 + κ) (μ^2 - λ)), {μ, 0, ∞}] // TraditionalForm
```

Out[54]//TraditionalForm=

$$\frac{1}{2\,\kappa\,\lambda}\left(-{}_3F_4\left(-i\,\sqrt{\kappa},\,i\,\sqrt{\kappa},\,-\sqrt{\lambda};\,1,\,1-\sqrt{-\kappa},\,\sqrt{-\kappa}+1,\,1-\sqrt{\lambda};\,e^{\pi/4}\,\varphi\right)-\right.$$
$$\left.{}_3F_4\left(-i\,\sqrt{\kappa},\,i\,\sqrt{\kappa},\,\sqrt{\lambda};\,1,\,1-\sqrt{-\kappa},\,\sqrt{-\kappa}+1,\,\sqrt{\lambda}+1;\,e^{\pi/4}\,\varphi\right)\right)$$

Formulas can be converted to TEX input:

In[55]:=
```
% // TeXForm
```

Out[55]//TeXForm=

```
\frac{-\, _3F_4\left(-i \sqrt{\kappa },i \sqrt{\kappa },-\sqrt{\lambda };1,1-\sqrt{-\
   }+1,1-\sqrt{\lambda };e^{\pi /4} \varphi \right)-\, _3F_4\left(-i \sqrt{\kappa },i
   };1,1-\sqrt{-\kappa },\sqrt{-\kappa }+1,\sqrt{\lambda }+1;e^{\pi /4} \varphi \right
```

Lists and Functional Programming

Functional Programming is an important programming paradigm since it is almost trivial to parallelize a program written in functional form.

Since in Mathematica everything is a list, so functional programming comes very naturally.

In[56]:=
```
{3, 4, 5, 7 / 8, x, y, x^2 + 3 y^3, {a, b, c},  }
```

Out[56]=

{3, 4, 5, $\frac{7}{8}$, x, y, $x^2 + 3\,y^3$, {a, b, c},  }

Parts of lists are indexed starting at 1, and can be extracted using[[...]]

In[57]:=
```
{a, b, c, d}[[3]]
```

Out[57]=

```
c
```

Negative indices count from the end :

In[58]:=
```
{a, b, c, d, e, f}[[-3]]
```

Out[58]=

```
d
```

Many operations immediately "thread" over lists :

In[59]:=
```
{1, 2, 3} + 2
```
Out[59]=
```
{3, 4, 5}
```

In[60]:=
```
{a, b, c} + {x, y, z}
```
Out[60]=
```
{a + x, b + y, c + z}
```

Refer to "spans" in lists using ;;

In[61]:=
```
{a, b, c, d, e, f}[[2 ;; 4]]
```
Out[61]=
```
{b, c, d}
```

Iterators

Make a table of the first 10 squares :

In[62]:=
```
Table[x^2, {x, 10}]
```
Out[62]=
```
{1, 4, 9, 16, 25, 36, 49, 64, 81, 100}
```

Many functions in the Wolfram Language use the standard "iterator specification" : min, max, step :

In[63]:=
```
Table[f[x], {x, 4, 20, 2}]
```
Out[63]=
```
{f[4], f[6], f[8], f[10], f[12], f[14], f[16], f[18], f[20]}
```

An alternative iterator specification just gives an explicit list of values :

In[64]:=
```
Table[f[x], {x, {5, 10, 20, 10, 5}}]
```
Out[64]=
```
{f[5], f[10], f[20], f[10], f[5]}
```

This makes a nested table :

In[65]:=
```
Table[i / j, {i, 4}, {j, 2}] // MatrixForm
```
Out[65]//MatrixForm=

$$\begin{pmatrix} 1 & \frac{1}{2} \\ 2 & 1 \\ 3 & \frac{3}{2} \\ 4 & 2 \end{pmatrix}$$

Assignments

Values can be assigned using =

In[66]:=
```
x = 7
```

Out[66]=
```
7
```

This is "immediate" assignment.

An alternative is delayed assignment, where the value is recomputed every time it is needed :

In[67]:=
```
t := Now
```

In[68]:=
```
t
```

Out[68]=
```
Tue 6 Jun 2023 14:28:55 GMT+2
```

In[69]:=
```
t
```

Out[69]=
```
Tue 6 Jun 2023 14:28:56 GMT+2
```

Clear the assignments :

In[70]:=
```
x =.
t =.
```

Use Module to localize variables :

In[72]:=
```
Module[{a = 1}, a + 8]
```

Out[72]=
```
9
```

In[73]:=
```
a
```

Out[73]=
```
a
```

Function Definitions

In the Wolfram Language, function definitions are just assignments that give transformation rules for patterns.

Define a function of two arguments named x and y :

In[74]:=
```
f[x_, y_] := x + y
```

Use the definition :

In[75]:=
```
f[4, a]
```

Out[75]=
```
4 + a
```

Clear the definition :

In[76]:=
```
Clear[f]
```

Pure Functions

The Wolfram Language allows what it calls pure functions, indicated by ending with &
Their first argument is indicated by #

(These are also known as anonymous functions, lambda expressions, etc.)

Make a pure function for adding 1 :

In[77]:=
```
(# + 1) &
```

Out[77]=
```
#1 + 1 &
```

If a pure function is given as the head of an expression, the function is applied to the arguments :

In[78]:=
```
(# + 1) &[50]
```

Out[78]=
```
51
```

Here is a function of several arguments :

In[79]:=
```
{#2, 1 + #1, #1 + #2} &[a, b]
```

Out[79]=
```
{b, 1 + a, a + b}
```

This is an alternative way to specify the function :

In[80]:=
```
Function[{x, y}, {y, 1 + x, x + y}][a, b]
```

Out[80]=
```
{b, 1 + a, a + b}
```

Lots of built - in functions commonly use pure functions :

In[81]:=
```
Select[{1, 4, 6, 8, 10, 15}, # > 7 &]
```

Out[81]=
```
{8, 10, 15}
```

Applying Functions

It' s very common to want to "map" a function over multiple expressions :

In[82]:=
```
Map[f, {a, b, c, d}]
```

Out[82]=
```
{f[a], f[b], f[c], f[d]}
```

/@ ("slash at") is a short notation for Map :

In[83]:=
```
f /@ {a, b, c, d}
```

Out[83]=
```
{f[a], f[b], f[c], f[d]}
```

This uses a pure function :

In[84]:=
```
{#, #} & /@ {a, b, c, d}
```

Out[84]=
```
{{a, a}, {b, b}, {c, c}, {d, d}}
```

Apply applies a function to multiple arguments :

In[85]:=
```
Apply[f, {a, b, c, d}]
```

Out[85]=
```
f[a, b, c, d]
```

Expressions have "levels"—corresponding to the number of indices needed to extract a part.Functions like **Map** can operate at specific levels.

Map defaults to operate at level 1 :

In[86]:=
```
Map[f, {{a, b}, {c, d}}]
```

Out[86]=
```
{f[{a, b}], f[{c, d}]}
```

This operates only at level 2 :

In[87]:=
```
Map[f, {{a, b}, {c, d}}, {2}]
```

Out[87]=
```
{{f[a], f[b]}, {f[c], f[d]}}
```

@@ is equivalent to **Apply**, operating by default at level 0 :

In[88]:=
```
f @@ {{a, b}, {c, d}}
```

Out[88]=
```
f[{a, b}, {c, d}]
```

@@@ means "apply at level 1" :

In[89]:=
```
f @@@ {{a, b}, {c, d}}
```
Out[89]=
```
{f[a, b], f[c, d]}
```

@means ordinary function application :

In[90]:=
```
f@x
```
Out[90]=
```
f[x]
```