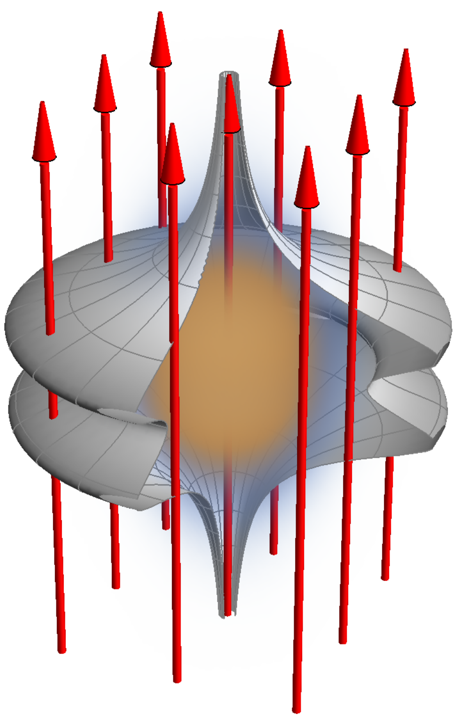
Introduction to Mathematica programming and scientific visualization

By Renan Cabrera



Philosophy

Unified Structure

Everything is an expression composed of a Head and a list of elements

```
a + b
```

```
a + b
```

```
FullForm[a + b]
```

```
Plus[a, b]
```

```
FullForm[a * b]
```

```
Times[a, b]
```

```
a + b + c // FullForm
```

```
Plus[a, b, c]
```

One of the most important expressions in Mathematica is the List

```
list = {1, 2, 3, 4};
```

```
list // FullForm
```

```
List[1, 2, 3, 4]
```

Lists can be nested to represent matrices

```
{list, list, list} // MatrixForm
```

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

Raw numbers are the exception

```
1 // FullForm
```

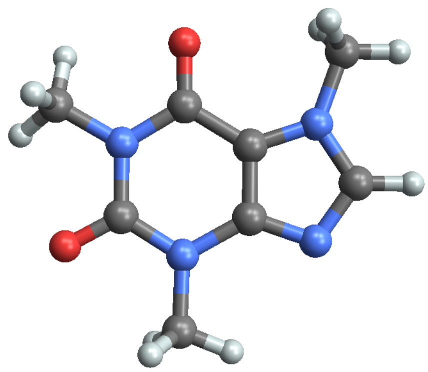
```
1
```

Preventing from prompt echo: use semicolon

```
a + b;
```

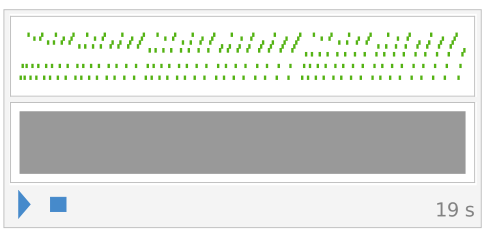
Everything is an expression, so high degree of personalization of graphics

```
ChemicalData["Caffeine", "MoleculePlot"]
```

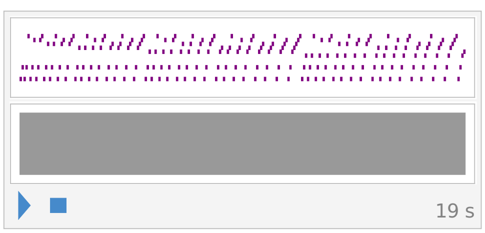


In similar way we can change the instrument of a midi sound file

```
Interpreter["Sound"][File["ExampleData/scaleprogression.mid"]]
```




```
Interpreter["Sound"][File["ExampleData/scaleprogression.mid"]] /. {"Harp" → "Piano"}
```



Functions, Functional Programming, and Rules

Symbolic Computation

Calculus

Derivatives

```
D[Sin[x], x]
```

Cos[x]

Integrals

```
Integrate[Sin[x], x]
```

-Cos[x]

```
Integrate[ Exp[-a x^2], {x, -∞, ∞}, Assumptions → {a > 0}]
```

$$\frac{\sqrt{\pi}}{\sqrt{a}}$$

```
Integrate[BesselJ[2, x], x]
```

$$\frac{1}{24} x^3 \text{HypergeometricPFQ}\left[\left\{\frac{3}{2}\right\}, \left\{\frac{5}{2}, 3\right\}, -\frac{x^2}{4}\right]$$

Differential equations and differential operators

Differential equations. Method 1

```
sol1 = DSolve[ f''[x] + x^2 f[x] == 0 , f[x], x]
```

$$\left\{\left\{f[x] \to C[2] \text{ ParabolicCylinderD}\left[-\frac{1}{2}, (-1+i)\, x\right] + C[1] \text{ ParabolicCylinderD}\left[-\frac{1}{2}, (1+i)\, x\right]\right\}\right\}$$

Advantage: Direct manipulation

```
sol1 // FunctionExpand
```

$$\left\{\left\{f[x] \to 2^{1/4}\, e^{\frac{i x^2}{2}}\, C[2] \text{ HermiteH}\left[-\frac{1}{2}, -\frac{(1-i)\, x}{\sqrt{2}}\right] + 2^{1/4}\, e^{-\frac{i x^2}{2}}\, C[1] \text{ HermiteH}\left[-\frac{1}{2}, \frac{(1+i)\, x}{\sqrt{2}}\right]\right\}\right\}$$

```
f[x] /. sol1
```

$$\left\{C[2] \text{ ParabolicCylinderD}\left[-\frac{1}{2}, (-1+i)\, x\right] + C[1] \text{ ParabolicCylinderD}\left[-\frac{1}{2}, (1+i)\, x\right]\right\}$$

Disadvantage: Substitution in differential equations

```
f'[x] /. sol1
```

$\{f'[x]\}$

Differential equations. Method 2

```
sol2 = DSolve[ f''[x] + x^2 f[x] == 0 , f, x]
```

$$\left\{\left\{f \to \text{Function}\left[\{x\}, C[2] \text{ ParabolicCylinderD}\left[-\frac{1}{2}, (-1+i)\, x\right] + C[1] \text{ ParabolicCylinderD}\left[-\frac{1}{2}, (1+i)\, x\right]\right]\right\}\right\}$$

Disadvantage: Not directly manipulable

```
sol2 // FunctionExpand
```

$$\left\{\left\{f \to \text{Function}\left[\{x\},\; C[2]\,\text{ParabolicCylinderD}\left[-\frac{1}{2},\,(-1+i)\,x\right] + C[1]\,\text{ParabolicCylinderD}\left[-\frac{1}{2},\,(1+i)\,x\right]\right]\right\}\right\}$$

Advantage: Substitution in differential equations

```
f[x] + a f'[x] /. sol2
```

$$\Big\{C[2]\,\text{ParabolicCylinderD}\left[-\frac{1}{2},\,(-1+i)\,x\right] + C[1]\,\text{ParabolicCylinderD}\left[-\frac{1}{2},\,(1+i)\,x\right] + a\,\Big((-1+i)\,C[2]\,\Big(\left(-\frac{1}{2}+\frac{i}{2}\right)x\,\text{ParabolicCylinderD}\left[-\frac{1}{2},\,(-1+i)\,x\right] - \text{ParabolicCylinderD}\left[\frac{1}{2},\,(-1+i)\,x\right]\Big) + (1+i)\,C[1]\,\Big(\left(\frac{1}{2}+\frac{i}{2}\right)x\,\text{ParabolicCylinderD}\left[-\frac{1}{2},\,(1+i)\,x\right] - \text{ParabolicCylinderD}\left[\frac{1}{2},\,(1+i)\,x\right]\Big)\Big)\Big\}$$

Differential Operators 1

```
ExtractOperator[ψ_][w_] := Module[{k, pre, DD},
  pre =
   Evaluate[w /. {ψ[x] → #, ψ'[x] → DD[#, x], Derivative[k_][ψ][x] :→ DD[#, {x, k}] }] &;
  pre /. {DD → D}
 ]
```

```
Commutator[F_, G_] := Module[{ψ, pre},
   pre = F[G[ψ[x]]] - G[F[ψ[x]]];
   ExtractOperator[ψ]@Simplify[pre]
  ];
```

Defining the position and momentum operator

```
xOperator = (x # &)
```

$x\,\#1\;\&$

```
pOperator = ((-I ħ D[#, x]) &)
```

$-i\,\hbar\,\partial_x \#1\;\&$

```
Commutator[xOperator, pOperator[pOperator[#]] & ]
```

$2\,\hbar^2\,\partial_x \#1\;\&$

```
Commutator[pOperator, F[x] # & ]
```

$-i\,\hbar\,\#1\,F'[x]\;\&$

```
Commutator[ pOperator@pOperator[#] &, F[x] # & ]
```

$-\hbar^2\,(2\,\partial_x \#1\,F'[x] + \#1\,F''[x])\;\&$

Differential Operators 2: Landau Levels

```
Clear@ExtractOperator

Clear@Commutator

Clear@A

ExtractOperator[ψ_][w_] := Module[{k, pre, DD},
  pre = Evaluate[w /. {
       ψ[x, y, z] → #,
       Derivative[kx_, ky_, kz_][ψ][x, y, z] :→ DD[#, {x, kx}, {y, ky}, {z, kz}] }] &;
  pre /. {DD → D}
 ]

Commutator[F_, G_] := Module[{ψ, pre},
   pre = F[G[ψ[x, y, z]]] - G[F[ψ[x, y, z]]];
   ExtractOperator[ψ]@Simplify[pre]
  ];

xOperator = (x # &);
yOperator = (y # &);
zOperator = (z # &);

pxOperator = ((-I ħ D[#, x]) &);
pyOperator = ((-I ħ D[#, y]) &);
pzOperator = ((-I ħ D[#, z]) &);

Commutator[xOperator, pxOperator]
```

ⅈ ħ #1 &

```
Commutator[xOperator, pOperator[pOperator[#]] & ]
```

$2\,\hbar^2\,\partial_{\{x,1\},\{y,0\},\{z,0\}}\#1\ \&$

$$\text{uxOperator} = \left(\frac{1}{m}\left(\text{pxOperator}[\#] - \frac{1}{c}\,\text{A1}[x, y, z]\,\#\right)\ \&\right);$$
$$\text{uyOperator} = \left(\frac{1}{m}\left(\text{pyOperator}[\#] - \frac{1}{c}\,\text{A2}[x, y, z]\,\#\right)\ \&\right);$$
$$\text{uzOperator} = \left(\frac{1}{m}\left(\text{pzOperator}[\#] - \frac{1}{c}\,\text{A3}[x, y, z]\,\#\right)\ \&\right);$$

$$\text{hamiltonianOperator} = \left(\frac{m}{2}\,\text{uxOperator@uxOperator}[\#] + \frac{m}{2}\,\text{uyOperator@uyOperator}[\#] + \frac{m}{2}\,\text{uzOperator@uzOperator}[\#] + V[x, y, z]\,\#\ \&\right)$$

$$\frac{1}{2}\,m\,\text{uxOperator}[\text{uxOperator}[\#1]] + \frac{1}{2}\,m\,\text{uyOperator}[\text{uyOperator}[\#1]] + \frac{1}{2}\,m\,\text{uzOperator}[\text{uzOperator}[\#1]] + V[x, y, z]\,\#1\ \&$$

```
hamiltonianOperator[ψ[x, y, z]] // Simplify
```

$$\frac{1}{2 c^2 m}\Big(A1[x,y,z]^2\,\psi[x,y,z]+A2[x,y,z]^2\,\psi[x,y,z]+A3[x,y,z]^2\,\psi[x,y,z]+2c^2 m\,V[x,y,z]\,\psi[x,y,z]+i\,c\,\hbar\,\psi[x,y,z]\,A3^{(0,0,1)}[x,y,z]+2 i\,c\,\hbar\,A3[x,y,z]\,\psi^{(0,0,1)}[x,y,z]-c^2\hbar^2\,\psi^{(0,0,2)}[x,y,z]+i\,c\,\hbar\,\psi[x,y,z]\,A2^{(0,1,0)}[x,y,z]+2 i\,c\,\hbar\,A2[x,y,z]\,\psi^{(0,1,0)}[x,y,z]-c^2\hbar^2\,\psi^{(0,2,0)}[x,y,z]+i\,c\,\hbar\,\psi[x,y,z]\,A1^{(1,0,0)}[x,y,z]+2 i\,c\,\hbar\,A1[x,y,z]\,\psi^{(1,0,0)}[x,y,z]-c^2\hbar^2\,\psi^{(2,0,0)}[x,y,z]\Big)$$

The vector potential of a homogeneous magnetic field

```
AHomogeneous = Cross[{0, 0, B3}, {x, y, z}] / 2
```

$$\left\{-\frac{B3\,y}{2},\ \frac{B3\,x}{2},\ 0\right\}$$

```
Curl[AHomogeneous, {x, y, z}]
```

{0, 0, B3}

Applying this vector potential to the Hamiltonian

```
hamiltonianOperator[ψ[x, y]] /. {
    V[x, y, z] → 0,
    A1 → Function[{x, y, z}, -B * y],
    A2 → Function[{x, y, z}, 0],
    A3 → Function[{x, y, z}, 0] } // Simplify
```

```
hamiltonianY = Exp[-I k x] (% /. {ψ → Function[{x, y}, Exp[I k x] ϕ[y]] }) // Simplify
```

$$\frac{1}{2 c^2 m}\left(B^2 y^2\,\psi[x,y]-c\,\hbar\left(c\,\hbar\,\psi^{(0,2)}[x,y]+2 i\,B\,y\,\psi^{(1,0)}[x,y]+c\,\hbar\,\psi^{(2,0)}[x,y]\right)\right)$$

$$\frac{(B\,y+c\,k\,\hbar)^2\,\phi[y]-c^2\,\hbar^2\,\phi''[y]}{2 c^2 m}$$

we obtain the Landau levels

```
DSolve[hamiltonianY == ϵ ϕ[y], ϕ[y], y] // FunctionExpand // Simplify
```

$$\left\{\left\{\phi[y]\to 2^{\frac{1}{4}-\frac{c\,m\,\epsilon}{2 B\hbar}}\,e^{-\frac{(B y+c k \hbar)^2}{2 B c \hbar}}\left(2^{\frac{c\,m\,\epsilon}{B\hbar}}\,e^{\frac{(B y+c k\hbar)^2}{B c\hbar}}\,C[2]\,\text{HermiteH}\left[-\frac{1}{2}-\frac{c\,m\,\epsilon}{B\,\hbar},\ \frac{i\,(B\,y+c\,k\,\hbar)}{\sqrt{B}\,\sqrt{c}\,\sqrt{\hbar}}\right]+C[1]\,\text{HermiteH}\left[-\frac{1}{2}+\frac{c\,m\,\epsilon}{B\,\hbar},\ \frac{B\,y+c\,k\,\hbar}{\sqrt{B}\,\sqrt{c}\,\sqrt{\hbar}}\right]\right)\right\}\right\}$$

with the proper energy quantization

$$\text{Last@Solve}\left[-\frac{1}{2}+\frac{c\,m\,\epsilon}{B\,\hbar}==n,\ \epsilon\right]\ /.\ \{B\to m\,c\,\omega_c\}$$

$$\left\{\epsilon\to\frac{1}{2}\,(1+2\,n)\,\hbar\,\omega_c\right\}$$

Differential Operators 3: Curvilinear coordinates

```
xyTorθRule = {x → r Cos[θ], y → r Sin[θ]}
```

$\{x \to r \, \mathrm{Cos}[\theta], y \to r \, \mathrm{Sin}[\theta]\}$

```
Map[Equal @@ # &, xyTorθRule];
rθToxyRule = Simplify[Last@Solve[%, {r, θ}] /. {C[1] → 0}]
```

$\left\{r \to \sqrt{x^2 + y^2}, \theta \to \mathrm{ArcTan}\left[\frac{x}{\sqrt{x^2 + y^2}}, \frac{y}{\sqrt{x^2 + y^2}}\right]\right\}$

```
ChainRuleOperator[ψ_][W_] := Module[{rule, pre, DD},
  rule = {
    Derivative[1, 0][ψ][x, y] :→
     (D[r /. rθToxyRule, x] DD[#, r] + D[θ /. rθToxyRule, x] DD[#, θ]),
    Derivative[0, 1][ψ][x, y] :→ (D[r /. rθToxyRule, y] DD[#, r] +
       D[θ /. rθToxyRule, y] DD[#, θ])};

  pre = Simplify[(W /. rule) /. xyTorθRule, Assumptions → {r > 0}];
  (Evaluate[pre] &) /. {DD → D}
 ]
```

```
D[ψ[x, y], x]
gradxOp = ChainRuleOperator[ψ][%]
```

$\psi^{(1,0)}[x, y]$

$\mathrm{Cos}[\theta] \, \partial_r \#1 - \frac{\partial_\theta \#1 \, \mathrm{Sin}[\theta]}{r} \; \&$

```
D[ψ[x, y], y]
gradyOp = ChainRuleOperator[ψ][%]
```

$\psi^{(0,1)}[x, y]$

$\frac{\mathrm{Cos}[\theta] \, \partial_\theta \#1}{r} + \partial_r \#1 \, \mathrm{Sin}[\theta] \; \&$

Gradient

```
{D[ψ[x, y], x], D[ψ[x, y], y]}
grad = ChainRuleOperator[ψ][%]
```

$\{\psi^{(1,0)}[x, y], \psi^{(0,1)}[x, y]\}$

$\left\{\mathrm{Cos}[\theta] \, \partial_r \#1 - \frac{\partial_\theta \#1 \, \mathrm{Sin}[\theta]}{r}, \frac{\mathrm{Cos}[\theta] \, \partial_\theta \#1}{r} + \partial_r \#1 \, \mathrm{Sin}[\theta]\right\} \; \&$

Laplacian

```
gradxOp@gradxOp[ψ[r, θ]] + gradyOp@gradyOp[ψ[r, θ]] // Simplify // Expand
```

$$\frac{\psi^{(0,2)}[r,\theta]}{r^2}+\frac{\psi^{(1,0)}[r,\theta]}{r}+\psi^{(2,0)}[r,\theta]$$

Numerics

Arbitrary precision

The numerical evaluation is achieved by the function N with the option of arbitrary numerical precision

```
Sin[π / 8]
N[%, 24]
```

$$\mathrm{Sin}\left[\frac{\pi}{8}\right]$$

```
0.382683432365089771728460
```

It is possible to specify the precision using the ` symbol

```
1.`32 / 3.`32
```

```
0.33333333333333333333333333333333
```

Complex numbers basics

```
Conjugate[1 + I]
```

1 - ⅈ

```
Conjugate[a]
```

```
Conjugate[a]
```

```
Simplify[
 Conjugate[a]
 , Assumptions → {Element[a, Reals]}]
```

```
a
```

Numerical Optimization

Random matrices

Scientific Visualization

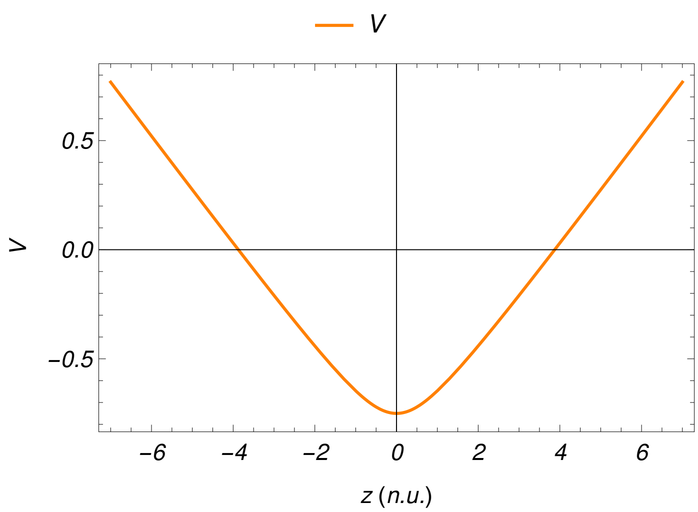
Basics

Before everything, let us only maintain only one step behind in the history of evaluation in order to improve performance

```
$HistoryLength = 1
```

```
1
```

```
potentialPlot = Plot[
   Sqrt[z^2 + 1] / 4 - 1 , {z, -7, 7},
  BaseStyle → {FontSize → 12, FontSlant -> Italic},
  PlotRange → All, FrameLabel → {"z  (n.u.)", "V"},
  PlotLegends → Placed[{Style["V", Italic]}, Top],
  Frame → True, PlotStyle → {Orange}]
```



The set of options that characterize this figure is found with AbsoluteOptions

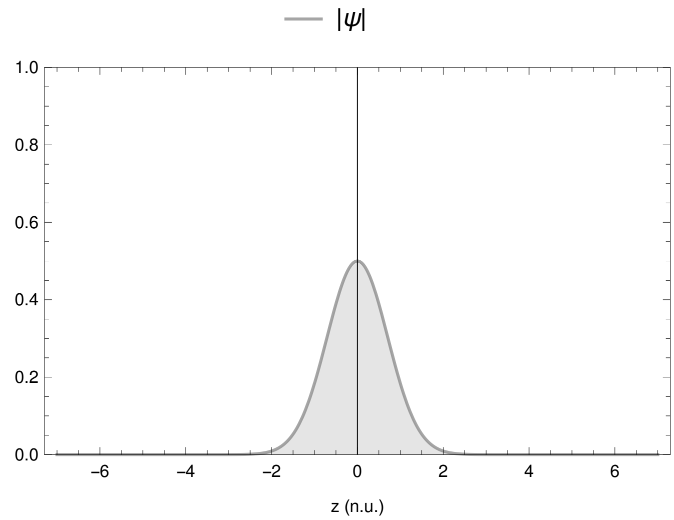
```
AbsoluteOptions[potentialPlot];
```

```
Ticks: {Automatic, Automatic} is not a valid tick specification.
Ticks: {Automatic, Automatic} is not a valid tick specification.
Ticks: {Automatic, Automatic} is not a valid tick specification.
General: Further output of Ticks::ticks will be suppressed during this calculation.
```

The internal Mathematica representation of the plot is found as

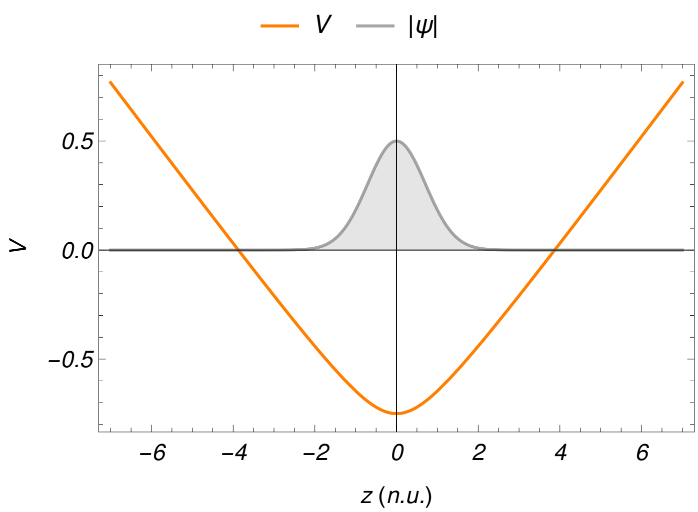
```
InputForm[potentialPlot];
```

```
statePlot = Plot[ Exp[-z^2] / 2, {z, -7, 7},
  PlotRange → {0, 1},
  FrameLabel → {"z (n.u.)", ""},
  Filling → Bottom, Frame → True,
  PlotLegends → Placed[{"|ψ|"}, Top],
  PlotStyle → {Gray, Opacity[0.7]}]
```



Two plots can be superimposed with Show

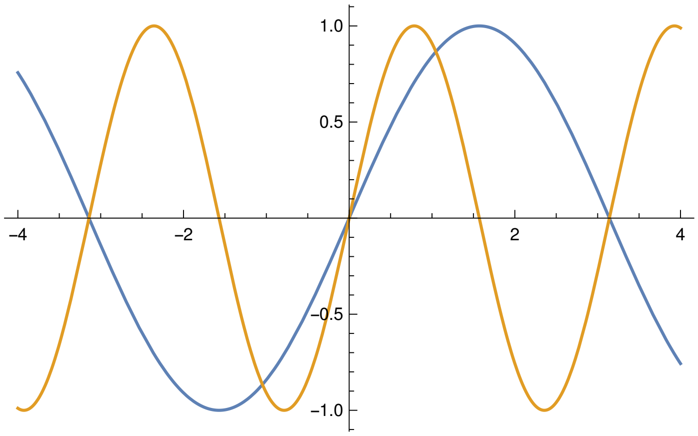
```
Show[ potentialPlot , statePlot]
```



This way to superimpose plots is intelligent in the sense that the validity of each plot is respected. The options are merged with the options of the first plot ruling out over the second one.

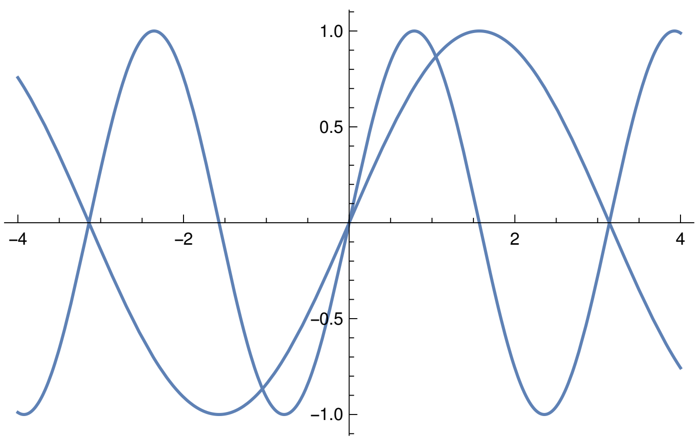
Multiple curves can be plotted

```
Plot[{Sin[x], Sin[2 x]}, {x, -4, 4}]
```



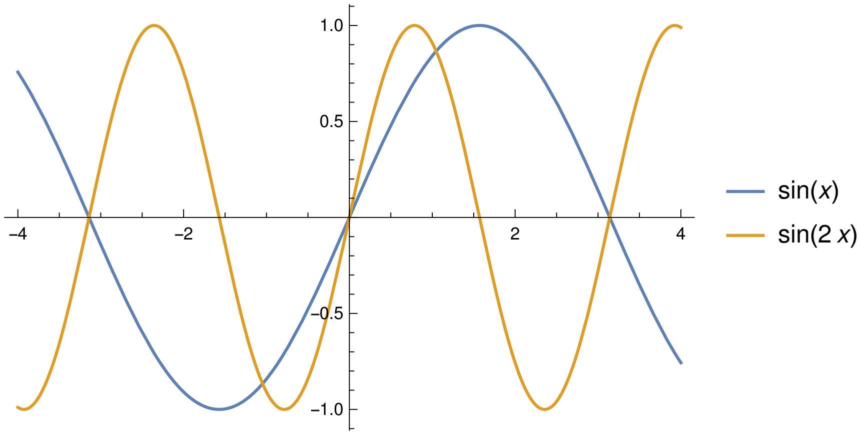
The following plot should be the same, but it is not

```
Plot[
 Table[Sin[n x], {n, 1, 2}],
 {x, -4, 4}, PlotLegends → "Expressions"]
```



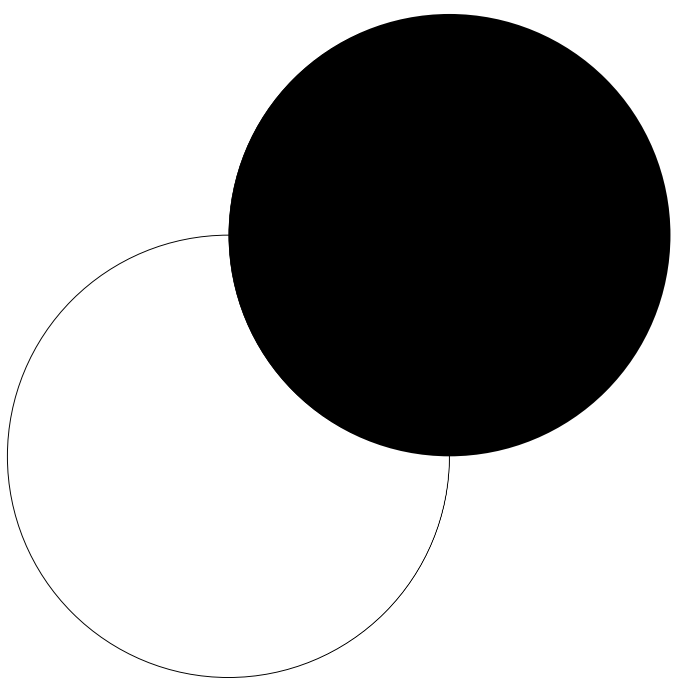
The correct behavior is obtained by applying an extra Evaluate function that forces the evaluation before executing the plotting.
This is an important key point to remember specially for demanding plots that otherwise would demand too much evaluation time.

```
Plot[
 Evaluate[
  Table[Sin[n x], {n, 1, 2}]
 ],
 {x, -4, 4}, PlotLegends → "Expressions"]
```

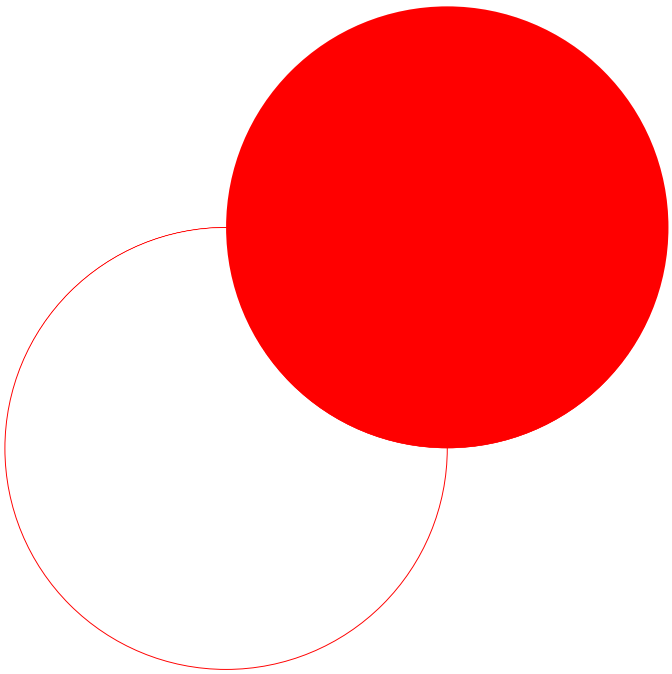


Objects

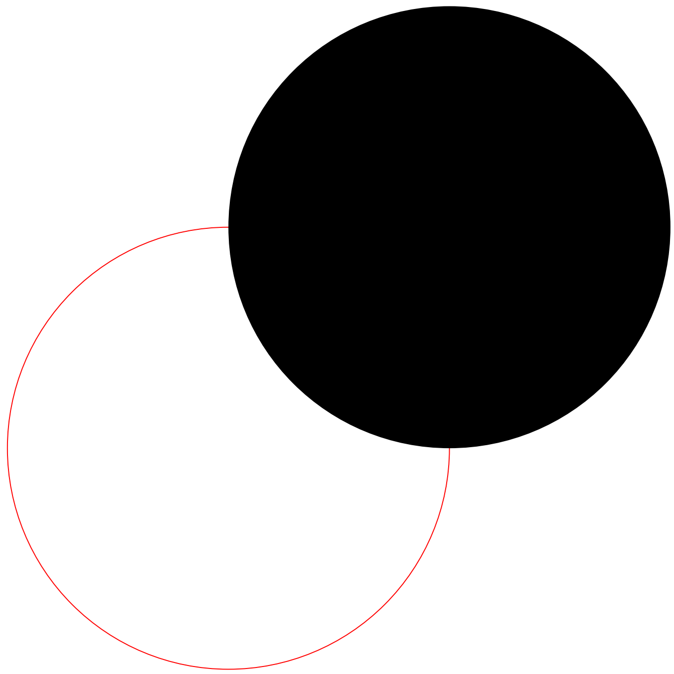
```
Graphics[
 {Circle[{0, 0}, 1], Disk[{1, 1}, 1]}
]
```



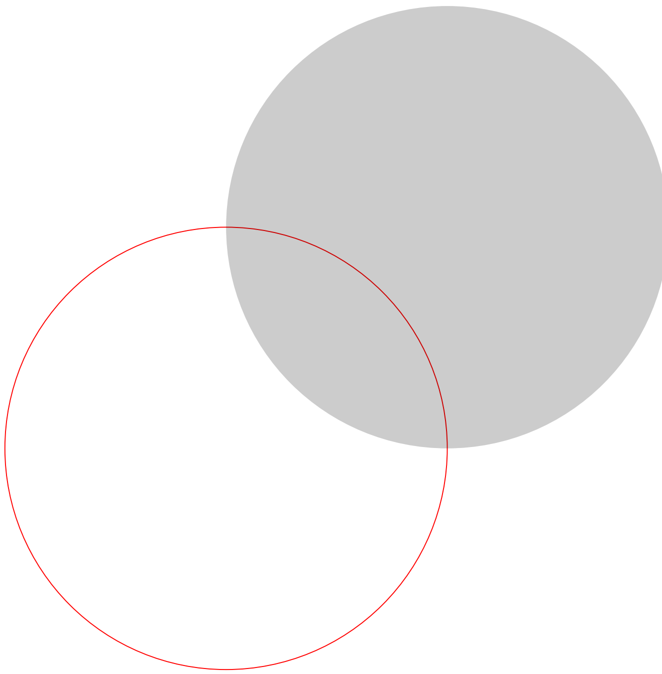
```
Graphics[
 {
  Red,
  Circle[{0, 0}, 1], Disk[{1, 1}, 1]
 }
]
```



```
Graphics[
 {
  {Red, Circle[{0, 0}, 1]},
  Disk[{1, 1}, 1]
 }
]
```



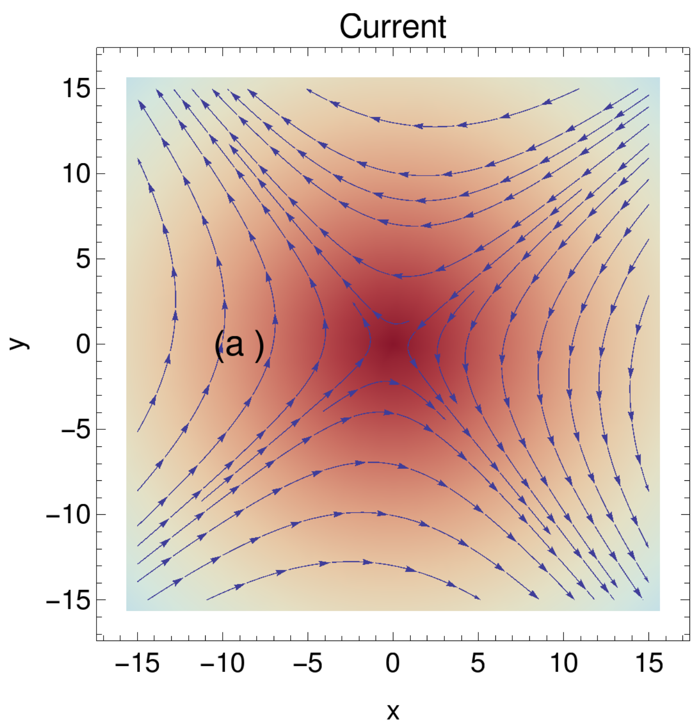
```
Graphics[
 {
  {Red, Circle[{0, 0}, 1]},
  {Opacity[0.2], Disk[{1, 1}, 1]}
 }
]
```



Stream plots

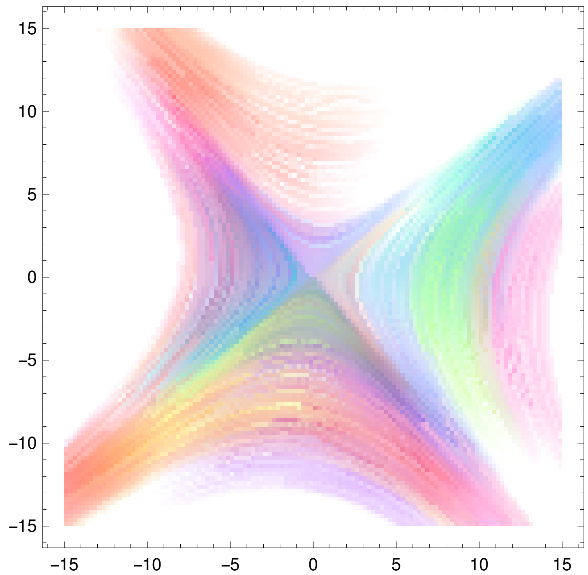
```
currentJ[{r_, B_, c_, m_, ħ_}, {t_, x_, y_}] =
  {-((r ω^2 (-c (-2 c (B + c m ω) + ω^2 ħ) Sin[t ω] + 2 B r ω^2 (y Cos[2 t ω] - x Sin[2 t ω]))) /
       (8 c^2 π Sqrt[(c - r ω) (c + r ω)])),
    (r ω^2 (c (2 c (B + c m ω) - ω^2 ħ) Cos[t ω] - 2 B r ω^2 (x Cos[2 t ω] + y Sin[2 t ω]))) /
     (8 c^2 π Sqrt[(c - r ω) (c + r ω)])} /. {ω → (c^2 m - Sqrt[c^4 m^2 + 2 B c ħ])/ħ} // Simplify;
```

```
StreamDensityPlot[
 currentJ[{2, 0.1, 1, 1, 1}, {1, x, y}]
 , {x, -15., 15.}, {y, -15, 15},
 StreamStyle → {Orange}, BaseStyle → {FontSize -> 14}, PlotLegends → Automatic,
 StreamPoints → 24,
 PlotLabel → "Current",
 FrameLabel → {"x", "y"},
 Epilog → { Inset[ Style["(a )", FontSize → 18], {-9, 0}] },
 ColorFunctionScaling → False,
 ColorFunction → (ColorData["ThermometerColors"][1 - 10 000 #5] &)]
```



```
im = Image[Graphics[
    {Opacity[0.5], PointSize[0.1],
     Table[{Hue[RandomReal[]], Disk[{RandomReal[], RandomReal[]}, 0.1]}, {20} ]}
   ], ImageSize → 300];
```

```
LineIntegralConvolutionPlot[
 {currentJ[{2, 0.1, 1, 1, 1}, {1, x, y}], im}
 , {x, -15., 15.}, {y, -15, 15}, RasterSize → 120,
 LineIntegralConvolutionScale → 8,
 ImageSize → 300]
```

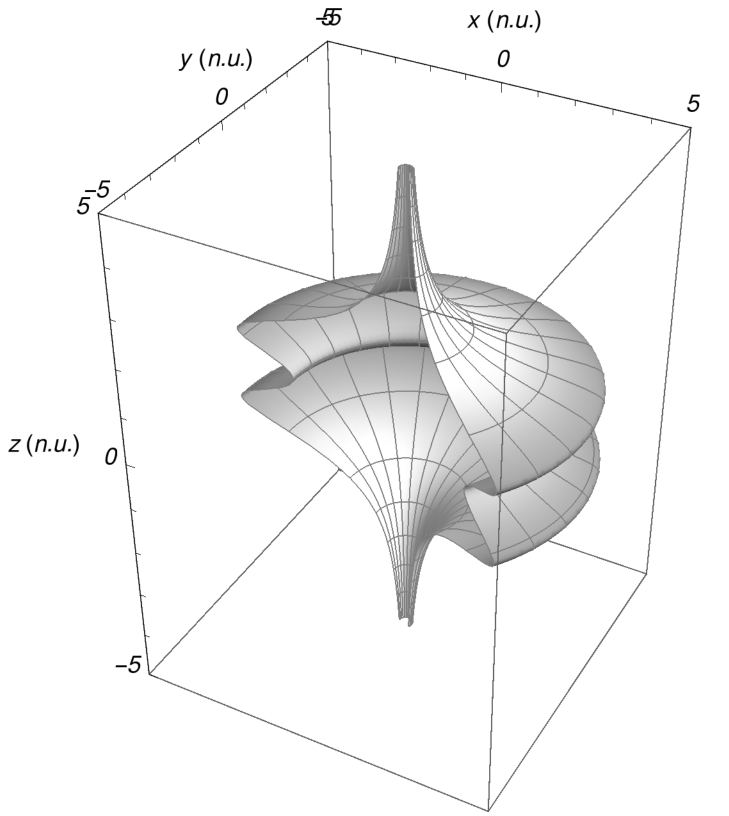


3D

Drawing a 3 D surface as a revolution in the z axis

$$\text{potential}[z\_] = \frac{-2 + \frac{3 z^6}{(1+z^4)^{7/4}} + \frac{2 z^6}{(1+z^4)^{3/2}} - \frac{3 z^2}{(1+z^4)^{3/4}}}{2 \sqrt{1 - \frac{z^6}{(1+z^4)^{3/2}}}};$$

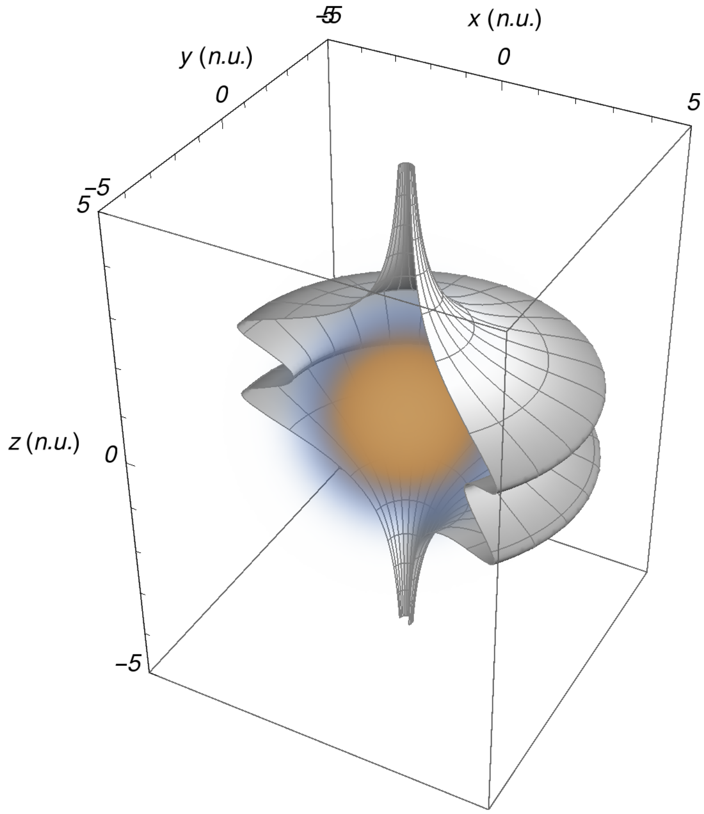
```
potentialPlot = RevolutionPlot3D[
  {3.3 potential[z], z}, {z, -5, 5},
  PlotRange → {{-5, 5}, {-5, 5}, {-5, 5}},
  MeshStyle → { Gray}, PlotPoints → {60, 60},
  ColorFunction → (RGBColor[1 - #6 / 2, 1 - #6 / 2, 1 - #6 / 2] &), BoxRatios → {5, 5, 7},
  BaseStyle → {FontSize -> 12, FontSlant → Italic},
  RegionFunction → Function[{x, y, z, t, θ, r}, Evaluate[2.5 < θ < 2 Pi]],
  AxesLabel → {"x (n.u.)", "y (n.u.)", "z (n.u.)"}]
```



Superimposing on a volume rendering of a Gaussian

```
statePlot = DensityPlot3D[
   Evaluate[(Exp[- ( (x)^2 + (y)^2 + 2 z^2) / 4])]
  , {x, -5, 5}, {y, -5, 5}, {z, -5, 5}, OpacityFunction → (#^(1.7) &)];
```

```
Show[potentialPlot, statePlot]
```



Another example, superimposing a vector field with the volume rendering of a Gaussian

```
Show[
  DensityPlot3D[
   Evaluate[(Exp[- ( (x - 2)^2 + (y)^2 + 2 z^2)/4 ])]
   , {x, -5, 5}, {y, -5, 5}, {z, -2, 2},
   BoxRatios → {5, 5, 2},
   OpacityFunction → (#^(1.7) &)]
  ,
  SliceVectorPlot3D[
   {x, -y, z}, {z == 0}, {x, -5, 5}, {y, -5, 5}, {z, -2, 2}, Lighting → "Neutral"]
 ];
```

Drawing two planes :
(a) Top plane with a density plot
(b) Bottom plane with a vector field

```
state = Rasterize[DensityPlot[ Exp[- (x - 2) ^2 - y^2], {x, -5, 5}, {y, -5, 5} ,
    PlotRange → {0, 1}, PlotPoints → 30,
    FrameTicks → {None, None},
    ColorFunction → (GrayLevel[1 - #] &)]];

stateAlpha = SetAlphaChannel[state, 0.6];

Show[
 Graphics3D[
  {Texture[ stateAlpha ], Polygon[{{-5, -5, 5}, {5, -5, 5}, {5, 5, 5}, {-5, 5, 5}},
    VertexTextureCoordinates → {{0, 0}, {1, 0}, {1, 1}, {0, 1}}] }
  , Lighting → "Neutral"]
 ,
 SliceVectorPlot3D[
  {x, -y, z}, {z == 0}, {x, -5, 5}, {y, -5, 5}, {z, -2, 2}, Lighting → "Neutral"]
]
```