MATHEMATICA POTPOURRI

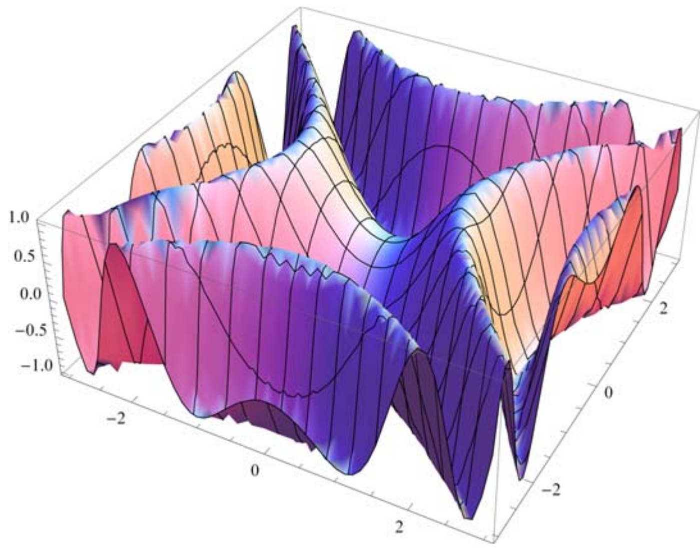
Now that you have a strong background in the fundamentals of Mathematica, here is a listing of some interesting aspects of Mathematica that might prove useful to you. Please check the doc center for more details and examples on each one.

More advanced plotting routines :

1) To plot in 3 dimensions in Cartesian coordinates :

In[235]:=
```
Plot3D[Sin[x^2 - y^2], {x, -3, 3}, {y, -3, 3}]
```

Out[235]=

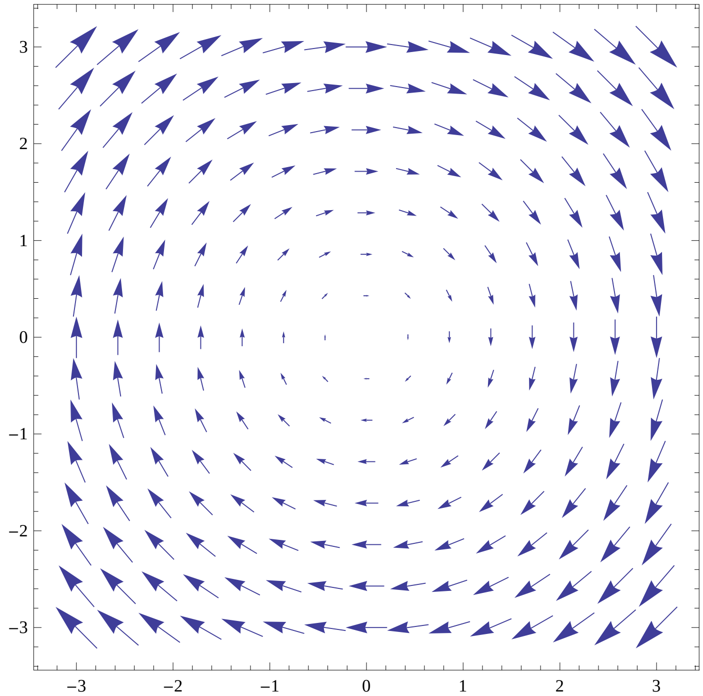


2) To plot a vector field :

To plot the 2 D vector field {y, -x} :

In[236]:= **VectorPlot**[{**y**, -**x**}, {**x**, -**3**, **3**}, {**y**, -**3**, **3**}]

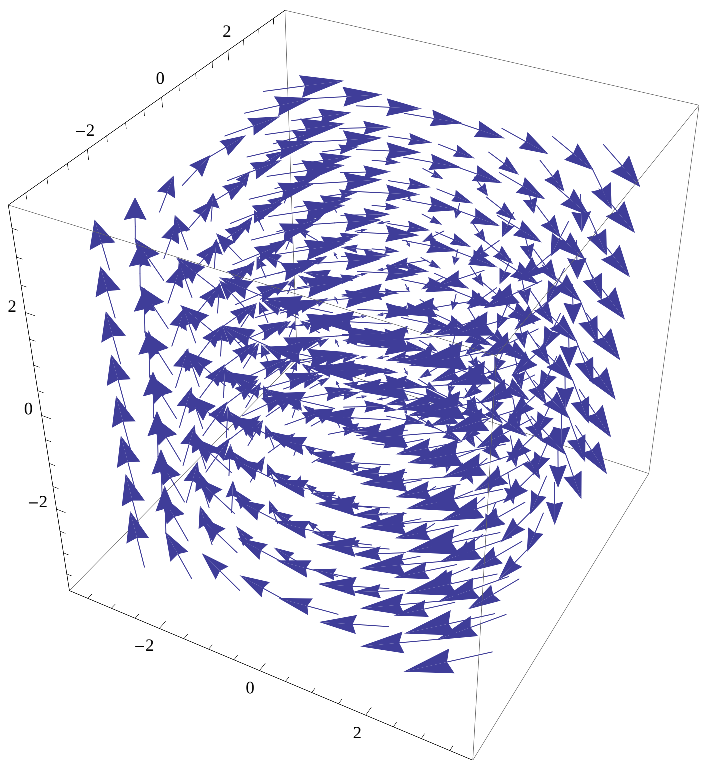
Out[236]=



To make this a 3 D field :

In[237]:= **VectorPlot3D**[{**y**, -**x**, **0**}, {**x**, -**3**, **3**}, {**y**, -**3**, **3**}, {**z**, -**3**, **3**}]

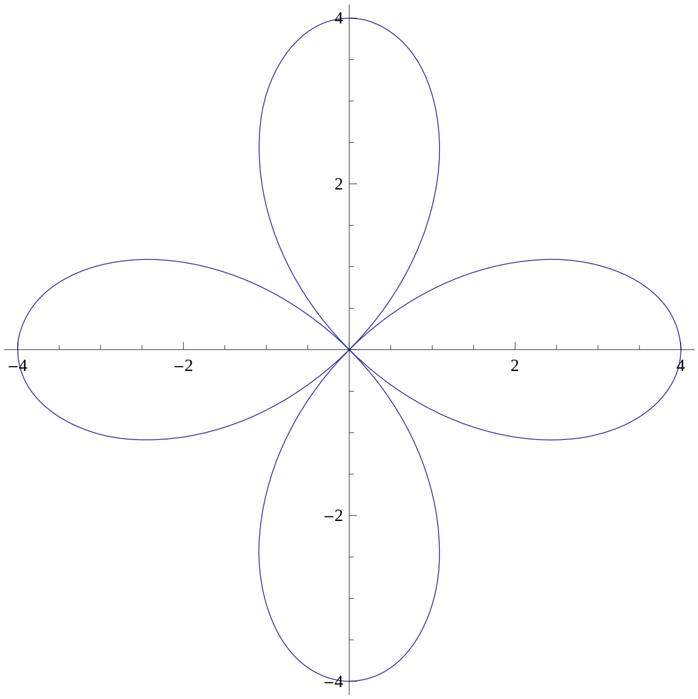
Out[237]=



3) Plotting in polar coordinates :

In[262]:=
```
PolarPlot[2^2 Cos[2 θ], {θ, 0, 2 π}]
```

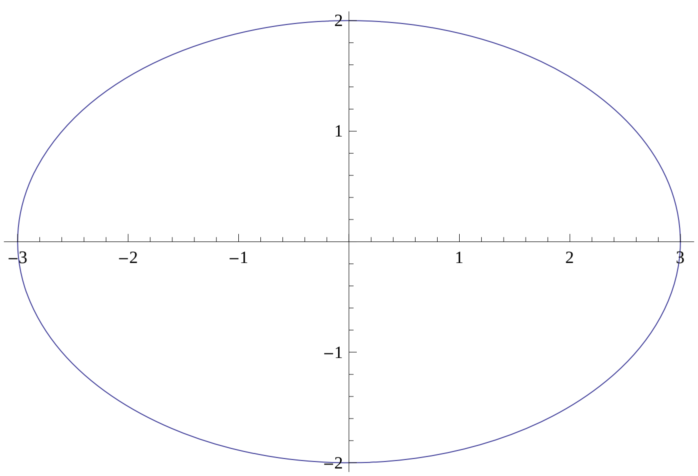
Out[262]=



4) Parametric plots (2 and 3 D)

In[264]:=
```
ParametricPlot[{3 Cos[t], 2 Sin[t]}, {t, 0, 2 π}]
```

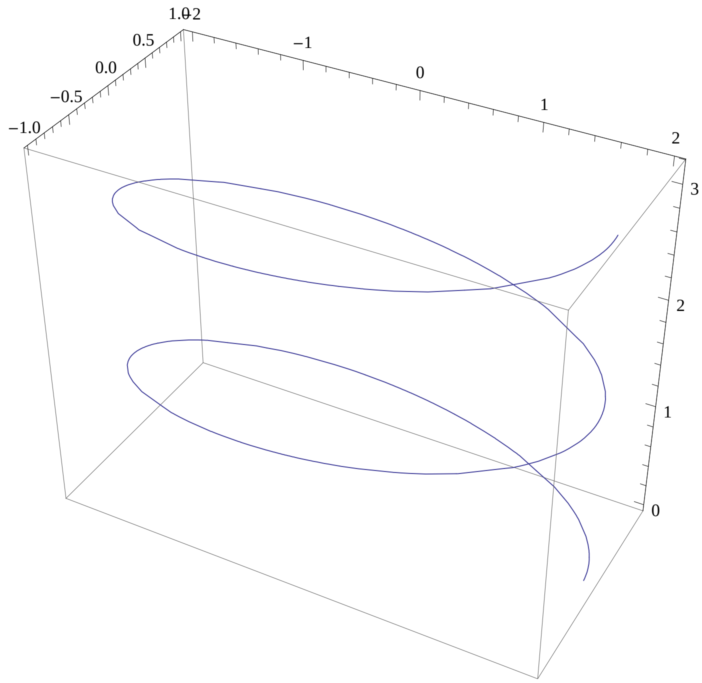
Out[264]=



In[265]:=

```
ParametricPlot3D[{2 Cos[t], Sin[t], t / 4}, {t, 0, 4 π}]
```

Out[265]=



5) Solving differential equations analytically :

Let' s consider the differential equation describing damped oscillatory motion :

$$m\,\ddot{x}(t) + c\,\dot{x} + k\,x = 0$$

where m is mass, c is the damping constant and k is the spring constant :

In[238]:=

```
Clear[m, x, c, k]
DSolve[m x''[t] + c x'[t] + x[t] == 0, x[t], t]
```

Out[239]= $\left\{\left\{x[t] \to e^{\frac{\left(-c-\sqrt{c^2-4m}\right)t}{2m}} C[1] + e^{\frac{\left(-c+\sqrt{c^2-4m}\right)t}{2m}} C[2]\right\}\right\}$

To solve with values and boundary conditions :

```
Clear[m, x, c, k]
```

In[246]:=

```
m = 1; c = 0.2; k = 10;
DSolve[{m x''[t] + c x'[t] + k x[t] == 0, x[0] == 0, x'[0] == 10}, x[t], t]
```

Out[247]= $\{\{x[t] \to 3.16386\, e^{-0.1\,t} \mathrm{Sin}[3.1607\, t]\}\}$

Solve this equation numerically :

In[254]:=

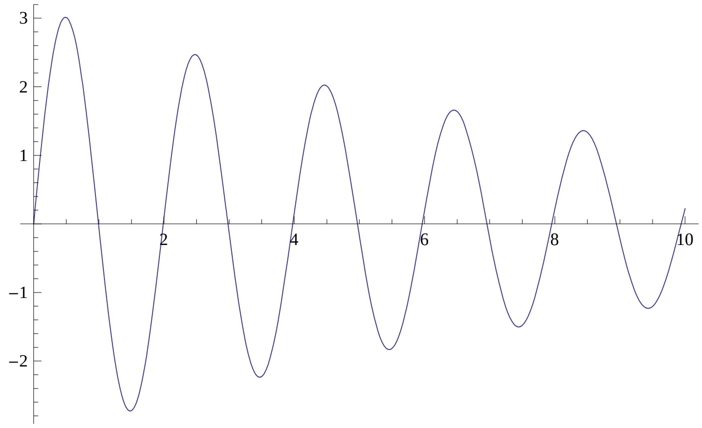
```
Clear[m, c, x, k]
m = 1; c = 0.2; k = 10;
s = NDSolve[{m x''[t] + c x'[t] + k x[t] == 0, x[0] == 0, x'[0] == 10.}, x, {t, 0, 10}]
```

Out[256]= {{x → InterpolatingFunction[{{0., 10.}}, <>]}}

Plot the solution :

In[259]:=
```
Plot[Evaluate[x[t] /. s], {t, 0, 10}, PlotRange → All]
```

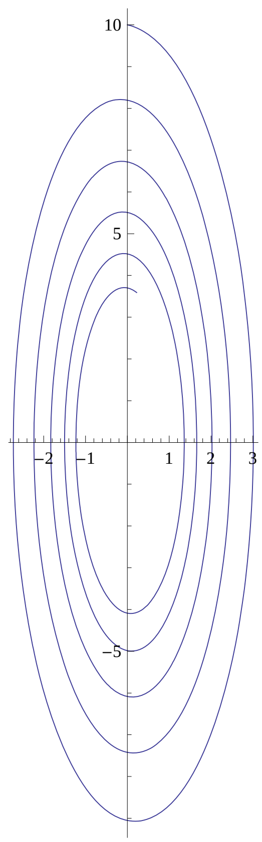
Out[259]=



Plot the Phase Diagram (v (t) vs. x (t)) :

In[261]:=
```
ParametricPlot[{x[t], x'[t]} /. s, {t, 0, 10}]
```

Out[261]=



6) Solve recursion relations like :

$$a_n = a_{n-1} + a_{n-2}$$

In[273]:=
```
Clear[a]
RSolve[a[n] == a[n - 1] + a[n - 2], a[n], n]
```

Out[274]=
```
{{a[n] → C[1] Fibonacci[n] + C[2] LucasL[n]}}
```

With boundary conditions :

In[275]:=
```
RSolve[{a[n] == a[n - 1] + a[n - 2], a[1] == 1, a[2] == 1}, a[n], n]
```

Out[275]=
```
{{a[n] → Fibonacci[n]}}
```