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

****

Orbits with Runge Kutta: Kepler $V = -k/r$

In[33]:= `Clear["Global`*"]`

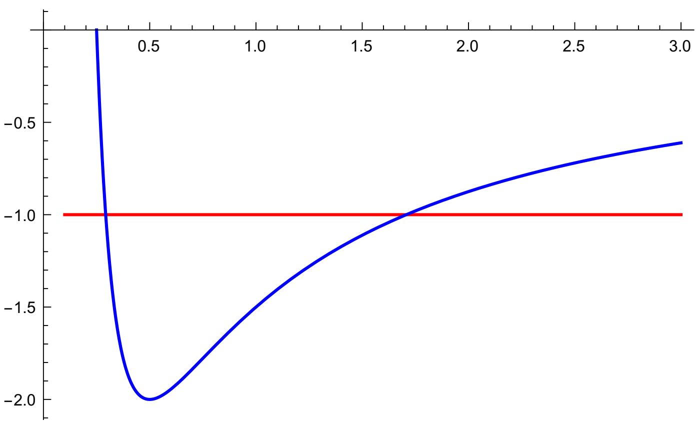
In[34]:= `values={m->1, el->1, k->2,energy->-1,power->1};`

In[35]:= $\text{Veff} = \left(\frac{\text{el}^2}{2\,\text{m}\,\text{r}^2} - \frac{\text{k}}{\text{r}^{\text{power}}}\right)$

Out[35]= $\frac{\text{el}^2}{2\,\text{m}\,\text{r}^2} - \text{k}\,\text{r}^{-\text{power}}$

In[36]:= `Plot[ {energy,Veff} /.values //Evaluate, {r,0.1,3},AxesOrigin->{0,0},PlotStyle->{Red,Blue}]`

Out[36]=



In[37]:= `(* T= Kinetic Energy *)`
`T=energy-Veff`

Out[38]= $\text{energy} - \frac{\text{el}^2}{2\,\text{m}\,\text{r}^2} + \text{k}\,\text{r}^{-\text{power}}$

In[39]:= (* T=Kinetic Energy$==\frac{1}{2}$ m v^2 *)

$\text{eq1} = \text{T} == \frac{1}{2}\,\text{m}\,\text{r}'[\text{t}]^2$

Out[39]= $\text{energy} - \frac{\text{el}^2}{2\,\text{m}\,\text{r}^2} + \text{k}\,\text{r}^{-\text{power}} == \frac{1}{2}\,\text{m}\,\text{r}'[\text{t}]^2$

In[40]:= `sol1 = Solve[eq1, r'[t]] // First`

Out[40]= $\left\{\text{r}'[\text{t}] \to -\frac{\sqrt{2\,\text{energy}\,\text{m} - \frac{\text{el}^2}{\text{r}^2} + 2\,\text{k}\,\text{m}\,\text{r}^{-\text{power}}}}{\text{m}}\right\}$

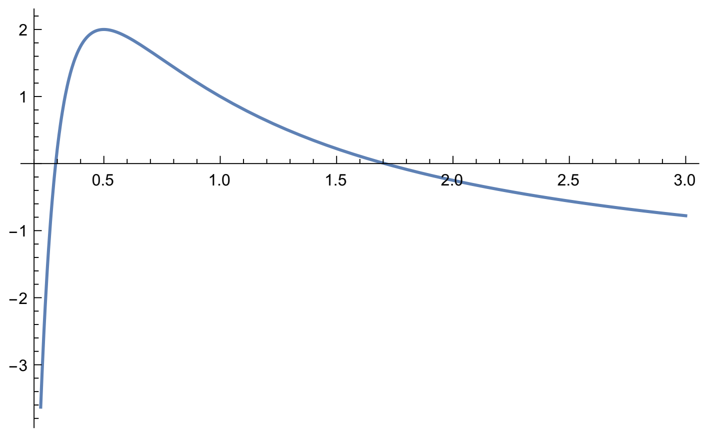
In[41]:=
```
(* This is the square of the velocity. It
 should be positive in the physical region!!! *)
dr2 = r'[t]^2 /. sol1 // Expand
```

Out[41]=

$$\frac{2\,\text{energy}}{m} - \frac{\text{el}^2}{m^2\,r^2} + \frac{2\,k\,r^{-\text{power}}}{m}$$

In[42]:=
```
Plot[dr2 /.values,{r,0.2,3}]
```

Out[42]=

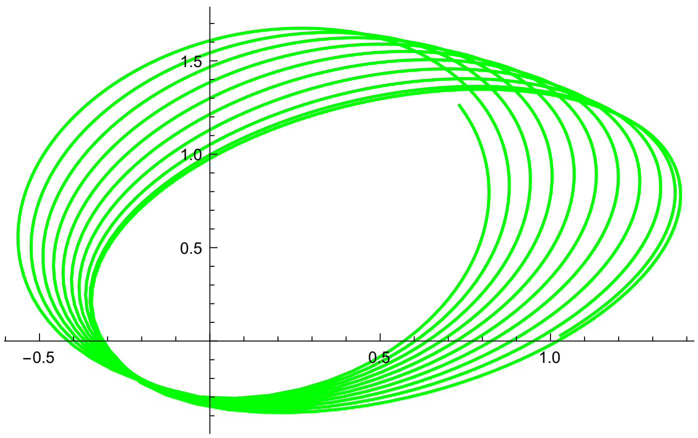


Naive 1 - step method

In[91]:=

```
r0 = 1; (* Make sure r0 is in the physical region where T>0 *)
theta0 = 0;
t0 = 0;
timeStep = 0.03;
rSign = +1; (* We'll start moving in the positive direction *)
xarray = {};
print = True; (* For debugging  *)
print = False; (* For debugging  *)

Do[
 dr20 = dr2 //. values //. r → r0; (* Slope SQUARED at r0  *)
 If[dr20 < 0, rSign = -rSign]; (* Change direction *)
 dr0 = rSign Sqrt[Abs[dr20]]; (* Slope not-SQUARED with sign included *)
 r1 = r0 + dr0 * timeStep;
 (* Step to endpoint [no 1/2 factor] ala Runge-Kutta 2-step method *)

 dtheta0 = el / (m r0^2) //. values; (* Advance theta *)
 theta1 = theta0 + dtheta0 * timeStep;

 t0 = t0 + timeStep; (* Advance time, and r point *)
 r0 = r1;
 theta0 = theta1;

 (* Collect points in xarray *)
 xarray = Append[xarray, {r1 Cos[theta1], r1 Sin[theta1]}];

 (* For debugging *)
 If[print, Print[t0, "   ", r0, "   ", dr0, "   ", theta0, "   ", rSign];],
 {i, 1, 1500}]

p1 = ListPlot[xarray, Joined → True, PlotStyle → Green]
```

Out[97]=



Runge - Kutta 2 - step method

In[105]:=

```
r0 = 1; (* Make sure r0 is in the physical region where T>0 *)
theta0 = 0;
t0 = 0;
timeStep = 0.03;
rSign = +1;  (* We'll start moving in the positive direction *)
xarray = {};
print = True; (* For debugging  *)
print = False; (* For debugging  *)

Do[
 dr20 = dr2 //. values //. r → r0; (* Slope SQUARED at r0  *)
 If[dr20 < 0, rSign = -rSign]; (* Change direction *)
 dr0 = rSign Sqrt[Abs[dr20]]; (* Slope not-SQUARED with sign included *)
 r0mid = r0 + dr0 * timeStep / 2; (* Step to midpoint ala Runge-Kutta 2-step method *)

 dr20 = dr2 //. values //. r → r0mid; (* Slope SQUARED at Midpoint r0mid  *)
 If[dr20 < 0, rSign = -rSign]; (* Change direction *)
 dr0 = rSign Sqrt[Abs[dr20]]; (* Slope not-SQUARED with sign included *)
 r1 = r0 + dr0 * timeStep;
 (* Step to endpoint [no 1/2 factor] ala Runge-Kutta 2-step method *)

 dtheta0 = el / (m r0^2) //. values; (* Advance theta *)
 theta1 = theta0 + dtheta0 * timeStep;

 t0 = t0 + timeStep; (* Advance time, and r point *)
 r0 = r1;
 theta0 = theta1;

 (* Collect points in xarray *)
 xarray = Append[xarray, {r1 Cos[theta1], r1 Sin[theta1]}];

 (* For debugging *)
 If[print, Print[t0, "   ", r0, "   ", dr0, "   ", theta0, "   ", rSign];],
 {i, 1, 1500}]

p2 = ListPlot[xarray, Joined → True, PlotStyle → Blue]
```

Out[111]=