Section 6.3: Foundations for RK-4

Andy Long, Spring, 2024

First of all, notice how beautifully simple this is. We're not using any step-size control, and that would make it better -- but this is a pretty good routine for something so simple!

The example is from our text, *Tea Time Numerical Analysis*

> This version of RK-4 with fixed step-size is for general **non**-autonomous f(t,y): that means you can incorporate time into your solutions.

We'll create the routines, and use the "standard example" to test the various routines (comparing to tables in the text):

In[250]:=
```
Clear[x, y, z, t, f]
f[t_, y_] := -y / t + t^2
yExact[t_] := t^3 / 4 + 16 / t
y0 = 20.0;
t0 = 4;
tf = 2;
h = -0.25;
pExact = Plot[yExact[t], {t, t0, tf}];
```

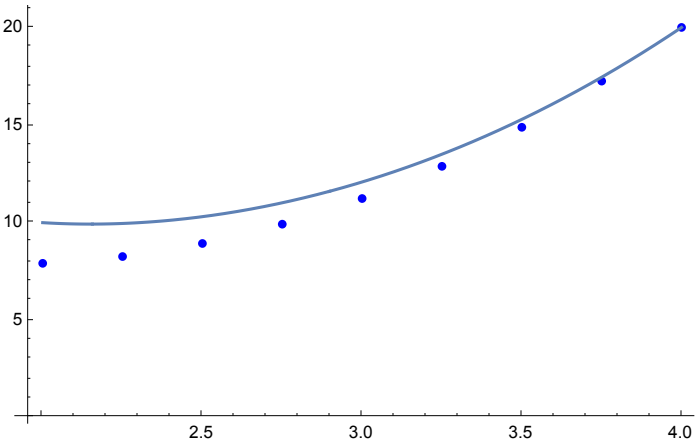
Euler-ode

In[258]:=
```
rkFoundationsEuler[f_, init_, tinit_, tfinal_, h_] :=
 Module[
  {n = Ceiling[(tfinal - tinit) / h],
   halfh = 0.5 h, t = tinit, w = init, wlist = init, g1, g2, g3, g4},
  {wlist} ~Join~
   Table[
    g1 = f[t, w];
    t = t + h; (* update t for the next step *)
    w = w + h g1,
    {n}
   ]
 ]
```

In[259]:=
```
solnEuler = rkFoundationsEuler[f, y0, t0, tf, h]
pEuler = ListPlot[Table[{t0 + (n - 1) * h, solnEuler[[n]]},
    {n, 1, Length[solnEuler]}], PlotStyle → Blue];
Show[pEuler, pExact, Axes → Automatic]
```

Out[259]=
```
{20., 17.25, 14.884375, 12.8850446428571, 11.2355769230769,
 9.921875, 8.93323863636364, 8.2640625, 7.91666666666667}
```

Out[261]=



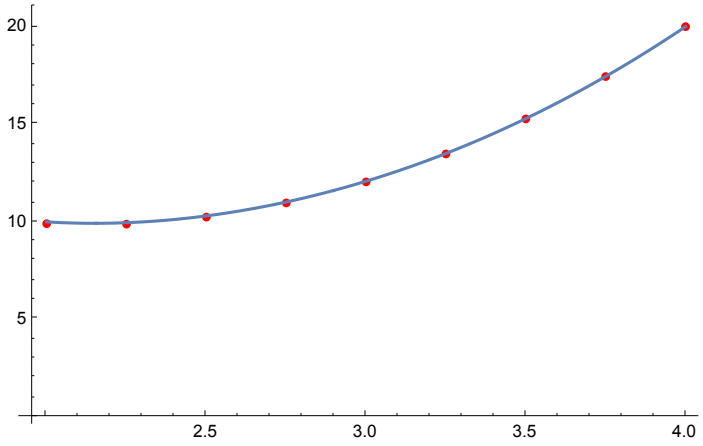
Trap-ode

In[262]:=
```
rkFoundationsTrap[f_, init_, tinit_, tfinal_, h_] :=
 Module[
  {n = Ceiling[(tfinal - tinit) / h],
   halfh = 0.5 h, t = tinit, w = init, wlist = init, g1, g2, g3, g4},
  {wlist} ~Join~
   Table[
    g1 = f[t, w];
    g2 = f[t + h, w + h g1];
    t = t + h; (* update t for the next step *)
    w = w + halfh (g1 + g2),
    {n}
   ]
 ]
```

In[263]:=
```
solnTrap = rkFoundationsTrap[f, y0, t0, tf, h]
pTrap = ListPlot[
   Table[{t0 + (n - 1) * h, solnTrap[[n]]}, {n, 1, Length[solnTrap]}], PlotStyle → Red];
Show[pTrap, pExact, Axes → Automatic]
```

Out[263]=
```
{20., 17.4421875, 15.2734375, 13.4789663461538,
 12.046875, 10.9694602272727, 10.2453125, 9.8828125, 9.90625}
```

Out[265]=



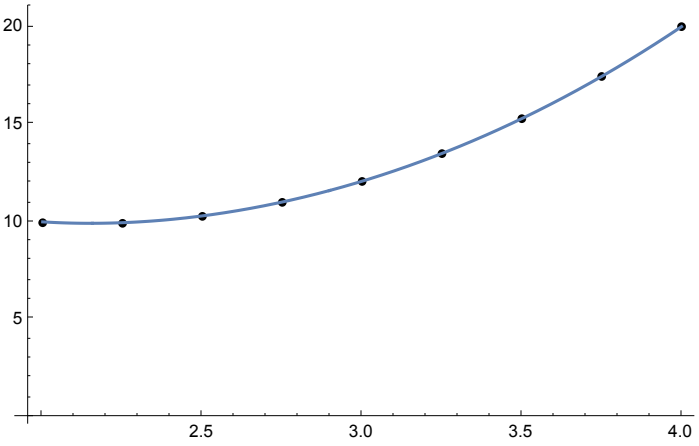
Mid-ode

In[266]:=
```
rkFoundationsMid[f_, init_, tinit_, tfinal_, h_] :=
 Module[
  {n = Ceiling[(tfinal - tinit) / h],
   halfh = 0.5 h, t = tinit, w = init, wlist = init, g1, g2, g3, g4},
  {wlist} ~Join~
   Table[
    g1 = f[t, w];
    g2 = f[t + halfh, w + halfh g1];
    t = t + h; (* update t for the next step *)
    w = w + h g2,
    {n}
   ]
 ]
```

In[267]:=
```
solnMid = rkFoundationsMid[f, y0, t0, tf, h]
pMid = ListPlot[
   Table[{t0 + (n - 1) * h, solnMid[[n]]}, {n, 1, Length[solnMid]}], PlotStyle → Black];
Show[pMid, pExact, Axes → Automatic]
```

Out[267]=
```
{20., 17.4477066532258, 15.2847217394327, 13.4962769456326, 12.0704748572697,
 10.9995826341963, 10.2820931616271, 9.92614194528067, 9.95544531593644}
```

Out[269]=



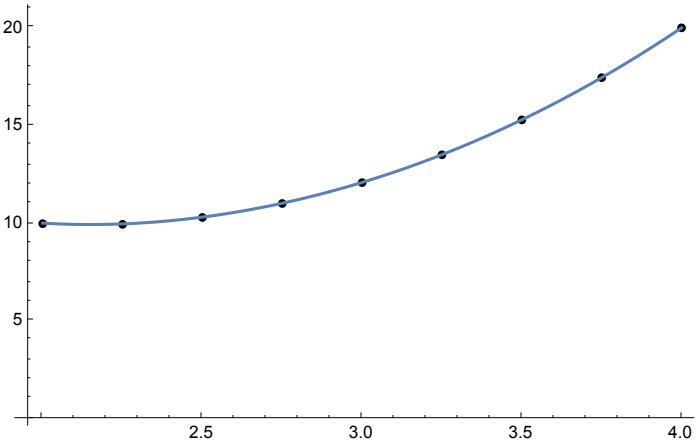
Simp-ode

In[270]:=
```
rkFoundationsSimp[f_, init_, tinit_, tfinal_, h_] :=
 Module[
  {n = Ceiling[(tfinal - tinit) / h],
   halfh = 0.5 h, t = tinit, w = init, wlist = init, g1, g2, g3, g4},
  {wlist} ~Join~
   Table[
    g1 = f[t, w];
    g2 = f[t + halfh, w + halfh g1];
    g3 = f[t + h, w + h g2];
    t = t + h; (* update t for the next step *)
    w = w + h (g1 + 4 g2 + g3) / 6,
    {n}
   ]
 ]
```

In[271]:=
```
solnSimp = rkFoundationsSimp[f, y0, t0, tf, h]
pSimp = ListPlot[
   Table[{t0 + (n - 1) * h, solnSimp[[n]]}, {n, 1, Length[solnSimp]}], PlotStyle → Black];
Show[pSimp, pExact, Axes → Automatic]
```

Out[271]=
```
{20., 17.4480636760753, 15.285569429317, 13.4978073808953, 12.0729705698583,
 11.0034706904965, 10.2880403490027, 9.93523067353151, 9.96951825736932}
```

Out[273]=



In[274]:=
```
solnExact = Table[yExact[t0 + k h], {k, 0, Length[solnEuler] - 1}]
TableForm[TableForm[Transpose[{solnEuler, solnTrap, solnMid, solnSimp, solnExact}],
  TableHeadings → {None, {"Euler-ode", "Trap-ode", "Mid-ode", "Simp-ode", "Exact"}}]]
```

Out[274]=
```
{20., 17.4502604166667, 15.2901785714286, 13.5051081730769,
 12.0833333333333, 11.0174005681818, 10.30625, 9.95876736111111, 10.}
```

Out[275]//TableForm=

Euler-ode	Trap-ode	Mid-ode	Simp-ode	Exact
20.	20.	20.	20.	20.
17.25	17.4421875	17.4477066532258	17.4480636760753	17.45
14.884375	15.2734375	15.2847217394327	15.285569429317	15.29
12.8850446428571	13.4789663461538	13.4962769456326	13.4978073808953	13.50
11.2355769230769	12.046875	12.0704748572697	12.0729705698583	12.08
9.921875	10.9694602272727	10.9995826341963	11.0034706904965	11.01
8.93323863636364	10.2453125	10.2820931616271	10.2880403490027	10.30
8.2640625	9.8828125	9.92614194528067	9.93523067353151	9.958
7.91666666666667	9.90625	9.95544531593644	9.96951825736932	10.

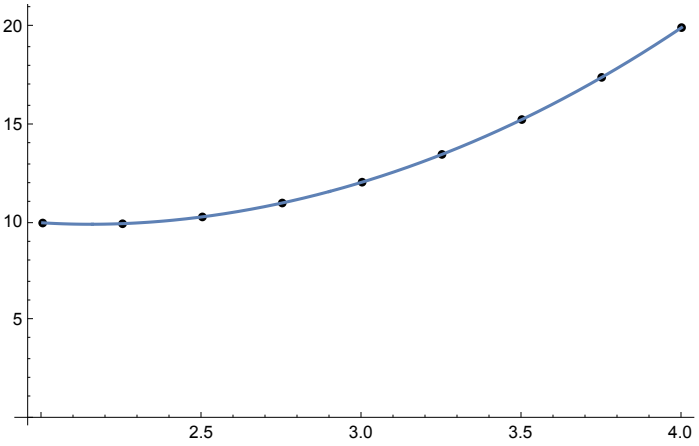
Open and ClOpen methods

In[276]:=
```
rkFoundationsOpen[f_, init_, tinit_, tfinal_, h_] :=
 Module[
  {n = Ceiling[(tfinal - tinit) / h], halfh = h / 2.0, thirdh = h / 3.0,
   twothirdh = 2 * h / 3.0, t = tinit, w = init, wlist = init, g1, g2, g3, g4},
  {wlist} ~Join~
   Table[
    g1 = f[t, w];
    g2 = f[t + thirdh, w + thirdh g1];
    g3 = f[t + twothirdh, w + twothirdh g2];
    t = t + h; (* update t for the next step *)
    w = w + halfh (g2 + g3),
    {n}
   ]
 ]
```

In[277]:=
```
solnOpen = rkFoundationsOpen[f, y0, t0, tf, h]
pOpen = ListPlot[
   Table[{t0 + (n - 1) * h, solnOpen[[n]]}, {n, 1, Length[solnOpen]}], PlotStyle → Black];
Show[pOpen, pExact, Axes → Automatic]
```

Out[277]=
```
{20., 17.4499852245863, 15.2895310746786, 13.5039492734378, 12.0814589164305,
 11.0145025034306, 10.3018472523169, 9.95207808145981, 9.98968933938873}
```

Out[279]=

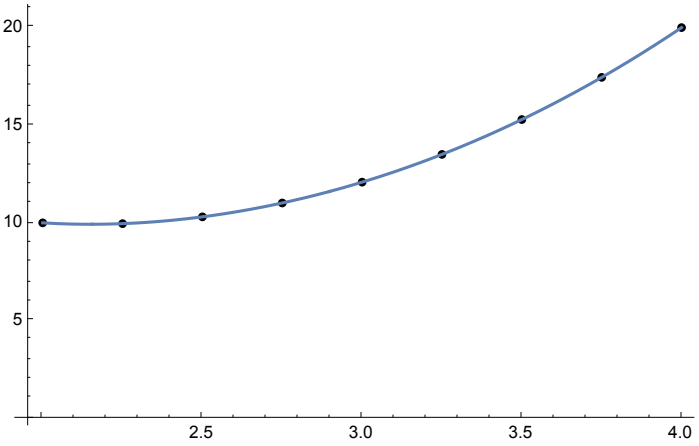


In[280]:=
```
rkFoundationsClosed[f_, init_, tinit_, tfinal_, h_] :=
 Module[
  {n = Ceiling[(tfinal - tinit) / h], fourthh = h / 4.0, thirdh = h / 3.0,
   twothirdh = 2 * h / 3.0, t = tinit, w = init, wlist = init, g1, g2, g3, g4},
  {wlist} ~Join~
   Table[
    g1 = f[t, w];
    g2 = f[t + thirdh, w + thirdh g1];
    g3 = f[t + twothirdh, w + twothirdh g2];
    t = t + h; (* update t for the next step *)
    w = w + fourthh (g1 + 3 g3),
    {n}
   ]
 ]
```

In[281]:=
```
solnClosed = rkFoundationsClosed[f, y0, t0, tf, h]
pClosed = ListPlot[Table[{t0 + (n - 1) * h, solnClosed[[n]]},
    {n, 1, Length[solnClosed]}], PlotStyle → Black];
Show[pClosed, pExact, Axes → Automatic]
```

Out[281]=
```
{20., 17.4502160904255, 15.2900786328072, 13.5049365701129, 12.0830664234219,
 11.0170021004248, 10.3056617392686, 9.95789049712821, 9.99865558531637}
```

Out[283]=



RK4-ode

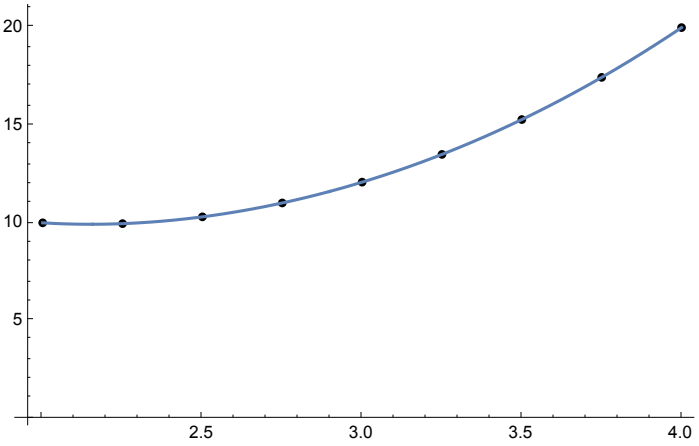
Let's add in Runge-Kutta-4: it's our target, and it certainly looks like these other methods!

In[284]:=
```
rk4[f_, init_, tinit_, tfinal_, h_] :=
 Module[
  {n = Ceiling[(tfinal - tinit) / h],
   halfh = 0.5 h, t = tinit, w = init, wlist = init, g1, g2, g3, g4},
  {wlist} ~Join~
   Table[
    g1 = f[t, w];
    g2 = f[t + halfh, w + halfh g1];
    g3 = f[t + halfh, w + halfh g2];
    g4 = f[t + h, w + h g3];
    t = t + h; (* update t for the next step *)
    w = w + h (g1 + 2. g2 + 2. g3 + g4) / 6.
    ,
    {n}
   ]
 ]
```

In[285]:=
```
solnRK4 = rk4[f, y0, t0, tf, h]
pRK4 = ListPlot[
   Table[{t0 + (n - 1) * h, solnRK4[[n]]}, {n, 1, Length[solnRK4]}], PlotStyle → Black];
Show[pRK4, pExact, Axes → Automatic]
```

Out[285]=
```
{20., 17.4502604166667, 15.2901785714286, 13.5051081730769,
 12.0833333333333, 11.0174005681818, 10.30625, 9.95876736111111, 10.}
```

Out[287]=



In[288]:=
```
TableForm[TableForm[
  Transpose[{solnSimp, solnOpen, solnClosed, solnRK4, solnExact}], TableHeadings →
   {None, {"Simp-ode", "Open-ode", "ClOpen-ode", "RK4-ode", "Exact"}}]]
```

Out[288]//TableForm=

Simp-ode	Open-ode	ClOpen-ode	RK4-ode	Exact
20.	20.	20.	20.	20.
17.4480636760753	17.4499852245863	17.4502160904255	17.4502604166667	17.45
15.285569429317	15.2895310746786	15.2900786328072	15.2901785714286	15.29
13.4978073808953	13.5039492734378	13.5049365701129	13.5051081730769	13.50
12.0729705698583	12.0814589164305	12.0830664234219	12.0833333333333	12.08
11.0034706904965	11.0145025034306	11.0170021004248	11.0174005681818	11.01
10.2880403490027	10.3018472523169	10.3056617392686	10.30625	10.30
9.93523067353151	9.95207808145981	9.95789049712821	9.95876736111111	9.958
9.96951825736932	9.98968933938873	9.99865558531637	10.	10.

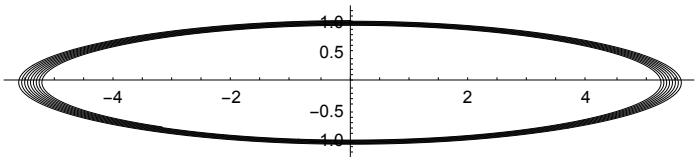
It’s easy to operate on systems:

Here’s the pendulum problem:

In[289]:=
```
Clear[x, y, z, f, c, m, g, l]
(* We're thinking of f as a vector-valued function,
obtained by reducing the second order equation for the pendulum to a
  first order system. The arguments a time t and a location (u,v): *)
f[t_, {u_, v_}] := {
  -c / m u - g / l Sin[v],
  u
 }
g = 9.81; (* m/s2 *)
c = 1.1;
l = .31; (* m *)
m = 70; (* kg *)
theta0 = Pi / 3;
thetaprime0 = 0;

(* We just give an initial value that is also a list, of the proper size: *)
rk4[f, {thetaprime0, theta0}, 0, 10, 0.0025];
Show[Graphics[{Line[%]}], Axes → Automatic]
```

Out[295]=



Here is the lovely Lorenz System, exhibiting chaos:

In[296]:=
```
Clear[x, y, z, f]
f[t_, {x_, y_, z_}] := {
  -3. (x - y),
  -x z + 26.5 x - y,
  x y - z
 }
rk4[f, {0., 1., 0.}, 0, 50., 0.0004];
Show[Graphics3D[{Line[%]}], Axes → Automatic]
```

Out[299]=