

Kiso Suuri Enshu C

Section 4 Exercises

Zehady Abdullah Khan
Student I.D: 09B12702
Email: a_khanzehady@yahoo.com

Excercise 4.1

In[1]:=

```
Clear[f, f1, f2, euler, Heun, RK,
      eulergraph, Heungraph, RKgraph, eug, heg, rug, ext]
```

In[7]:=

```
f[x_, t_] := -x;
f1[x1_, x2_, t_] := x2;
f2[x1_, x2_, t_] := -x1;
```

Euler Method

In[10]:=

```
euler::nonnegativeint = "Nmax must be a non-negative integer";

euler[dt_, Nmax_, a1_, a2_] :=
  Module[{j = 0, n, x1, x2, t}, If[Or[Nmax ≤ 0, Not[IntegerQ[Nmax]]],
    Message[euler::nonnegativeint], x1[0] := a1; x2[0] := a2;
    x1[j_] := (x1[j] = x1[j - 1] + dt f1[x1[j - 1], x2[j - 1], t]);
    x2[j_] := (x2[j] = x2[j - 1] + dt f2[x1[j - 1], x2[j - 1], t]);
    Table[{j dt, x1[j]}, {j, 0, Nmax}]
  ]
];
```

In[12]:=

```
euler[.1, 10, 1, 2]
```

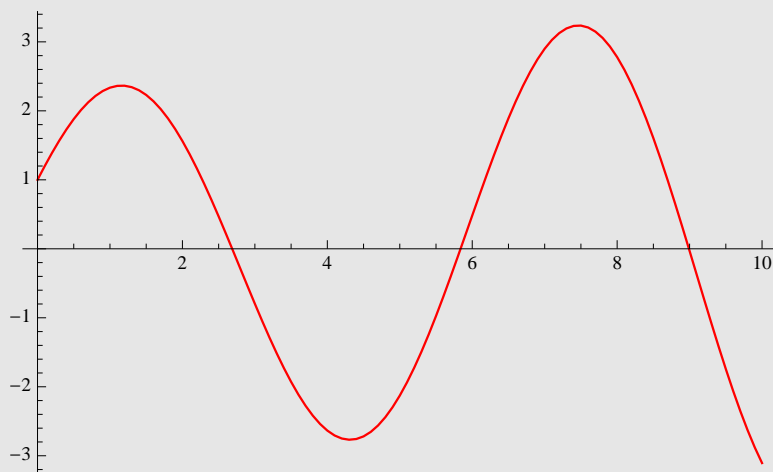
Out[12]=

```
{{0., 1}, {0.1, 1.2}, {0.2, 1.39}, {0.3, 1.568}, {0.4, 1.7321}, {0.5, 1.88052},
 {0.6, 2.01162}, {0.7, 2.12391}, {0.8, 2.21609}, {0.9, 2.28703}, {1., 2.33581}}
```

In[24]:=

```
eulergraph[dt_, Nmax_, a1_, a2_] := ListLinePlot[euler[dt, Nmax, a1, a2],
  PlotStyle → {Thickness[0.003], RGBColor[1, 0, 0]}, PlotRange → All];
eulergraph[.1, 100, 1, 2]
```

Out[25]=



Heun Method

In[16]:=

```
Heun[dt_, Nmax_, a1_, a2_] := Module[
  {j = 0, n, g1, g2, h1, h2, x1, x2, t}, If[Or[Nmax ≤ 0, Not[IntegerQ[Nmax]]],
    Message[Heun::nonnegativeint], x1[0] := a1; x2[0] := a2;
    g1[j_] := (g1[j] = f1[x1[j], x2[j], t]);
    h1[j_] := (h1[j] = f2[x1[j], x2[j], t]);

    g2[j_] := (g2[j] = f1[x1[j] + dt g1[j], x2[j] + dt h1[j], t + dt]);
    h2[j_] := (h2[j] = f2[x1[j] + dt g1[j], x2[j] + dt h1[j], t + dt]);
    x1[j_] := (x1[j] = x1[j - 1] +  $\frac{dt}{2}$  (g1[j - 1] + g2[j - 1]));

    x2[j_] := (x2[j] = x2[j - 1] +  $\frac{dt}{2}$  (h1[j - 1] + h2[j - 1]));
    Table[{j dt, x1[j]}, {j, 0, Nmax}]
  ]
];
```

In[17]:=

```
Heun[.1, 10, 1, 2]
```

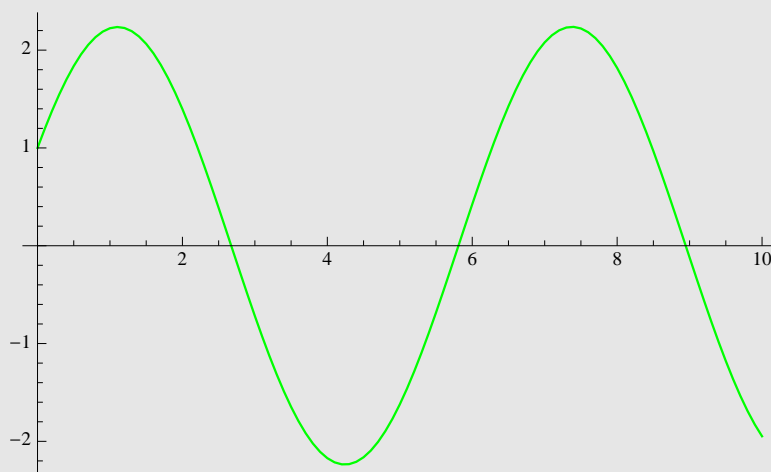
Out[17]=

```
{{0., 1}, {0.1, 1.195}, {0.2, 1.37803}, {0.3, 1.54724},
 {0.4, 1.70095}, {0.5, 1.83761}, {0.6, 1.95585},
 {0.7, 2.05449}, {0.8, 2.13253}, {0.9, 2.1892}, {1., 2.22392}}
```

In[26]:=

```
Heungraph[dt_, Nmax_, a1_, a2_] := ListLinePlot[Heun[dt, Nmax, a1, a2],
  PlotStyle → {Thickness[0.003], RGBColor[0, 1, 0]}, PlotRange → All];
Heungraph[.1, 100, 1, 2]
```

Out[27]=



Runge-Kutta Method

In[20]:=

```

RK::nonnegativeint = "Nmax must be a non-negative integer";

RK[dt_, Nmax_, a1_, a2_] :=
Module[{j = 0, n, g1, g2, g3, g4, h1, h2, h3, h4, x1, x2, t},
  If[Or[Nmax ≤ 0, Not[IntegerQ[Nmax]]], Message[RK::nonnegativeint],
    x1[0] := a1; x2[0] := a2;
    g1[j_] := (g1[j] = f1[x1[j], x2[j], t]);
    h1[j_] := (h1[j] = f2[x1[j], x2[j], t]);

    g2[j_] := (g2[j] = f1[x1[j] + dt / 2 g1[j], x2[j] + dt / 2 h1[j], t + dt / 2]);
    h2[j_] := (h2[j] = f2[x1[j] + dt / 2 g1[j], x2[j] + dt / 2 h1[j], t + dt / 2]);

    g3[j_] := (g3[j] = f1[x1[j] + dt / 2 g2[j], x2[j] + dt / 2 h2[j], t + dt / 2]);
    h3[j_] := (h3[j] = f2[x1[j] + dt / 2 g2[j], x2[j] + dt / 2 h2[j], t + dt / 2]);

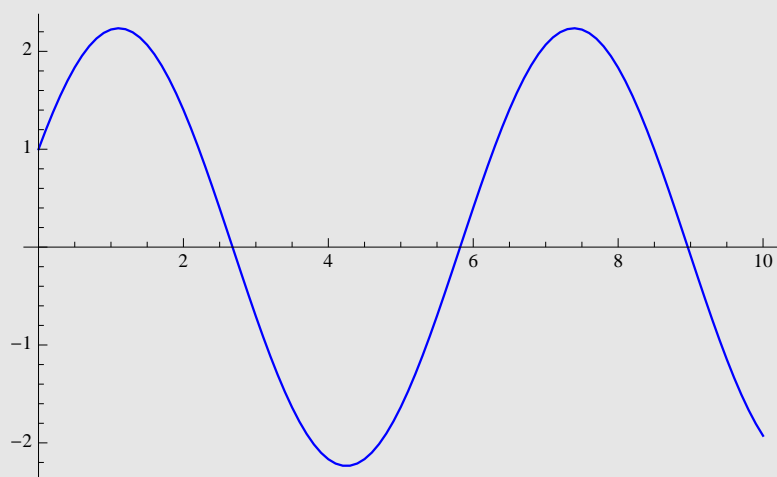
    g4[j_] := (g4[j] = f1[x1[j] + dt g3[j], x2[j] + dt h3[j], t + dt]);
    h4[j_] := (h4[j] = f2[x1[j] + dt g3[j], x2[j] + dt h3[j], t + dt]);
    x1[j_] :=
      (x1[j] = x1[j - 1] +  $\frac{dt}{6}$  (g1[j - 1] + 2 g2[j - 1] + 2 g3[j - 1] + g4[j - 1]));
    x2[j_] := (x2[j] = x2[j - 1] +
       $\frac{dt}{6}$  (h1[j - 1] + 2 h2[j - 1] + 2 h3[j - 1] + h4[j - 1]));
    Table[{j dt, x1[j]}, {j, 0, Nmax}]
  ]
];

```

In[28]:=

```
RKgraph[dt_, Nmax_, a1_, a2_] := ListLinePlot[RK[dt, Nmax, a1, a2],  
  PlotStyle → {Thickness[0.003], RGBColor[0, 0, 1]}, PlotRange → All];  
RKgraph[  
  .1,  
  100,  
  1,  
  2]
```

Out[29]:=



Analytic

In[140]:=

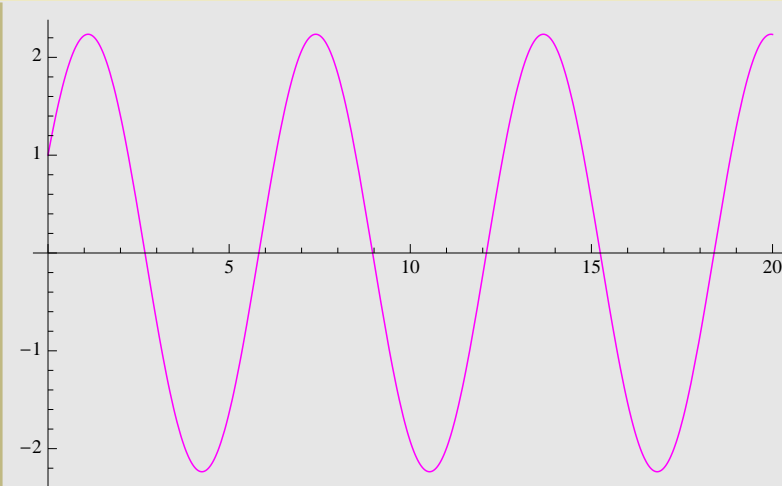
```
asol[a1_, a2_] :=
  DSolve[{x''[t] == f[x[t], t], x[0] == a1, x'[0] == a2}, x[t], t];

asol[a1, a2]
agraph[a1_, a2_, tlast_] :=
  Plot[Evaluate[x[t] /. asol[a1, a2]], {t, 0, tlast},
    PlotStyle -> {Thickness[0.002], RGBColor[1, 0, 1]}, PlotRange -> All]
exactgraph[dt_, Nmax_, a1_, a2_] :=
  Plot[a1 Cos[t] + a2 Sin[t], {t, 0, dt * Nmax}, PlotStyle -> RGBColor[1, 1, 0]];
agraph[
  1,
  2,
  20]
```

Out[141]=

```
{{x[t] -> a1 Cos[t] + a2 Sin[t]}}
```

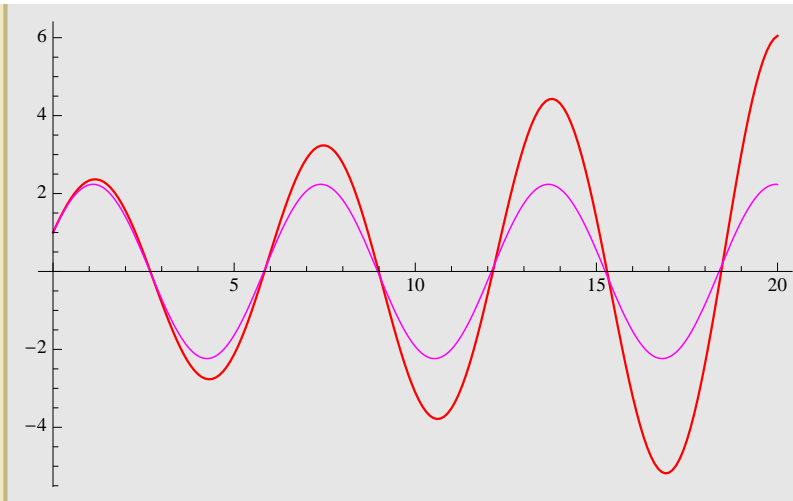
Out[144]=



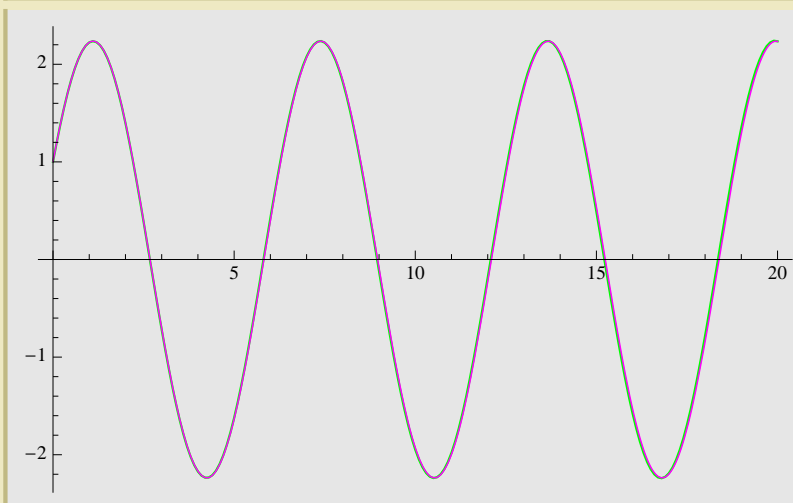
Comparison of Approximation for 3 methods

```
eug = eulergraph[.1, 200, 1, 2];
heg = Heungraph[.1, 200, 1, 2];
rug = RKgraph[.1, 200, 1, 2];
ext = agraph[1, 2, 20];
Show[eug, ext]
Show[heg, ext]
Show[rug, ext]
Show[eug, heg, rug, ext]
```

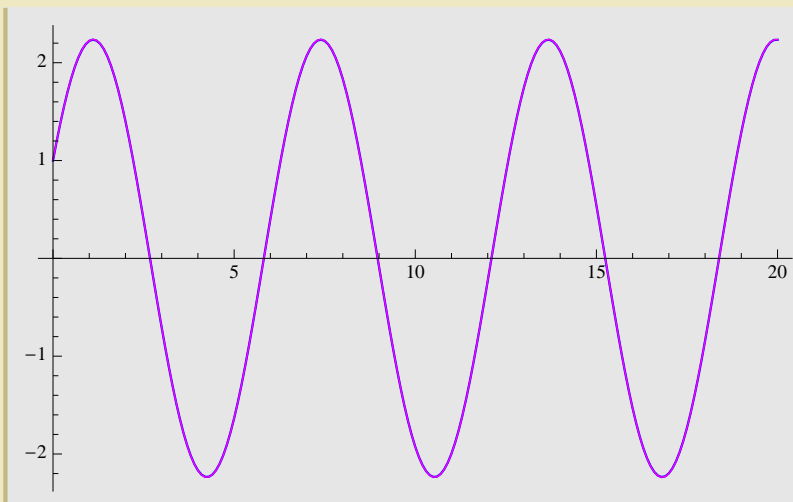
Out[157]=



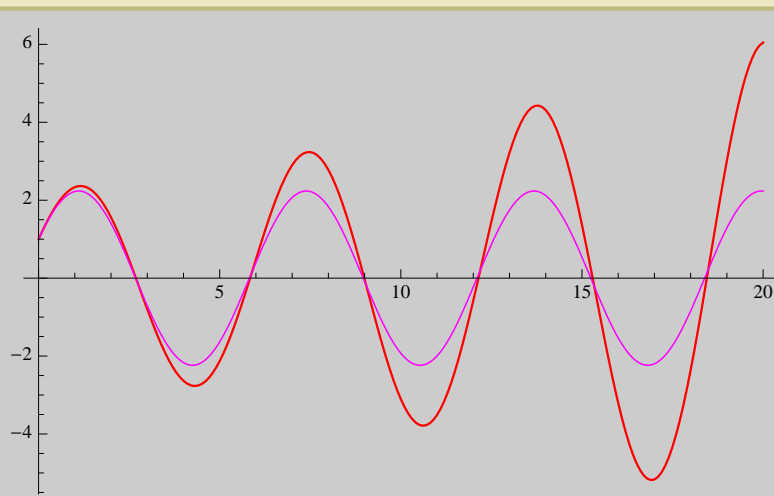
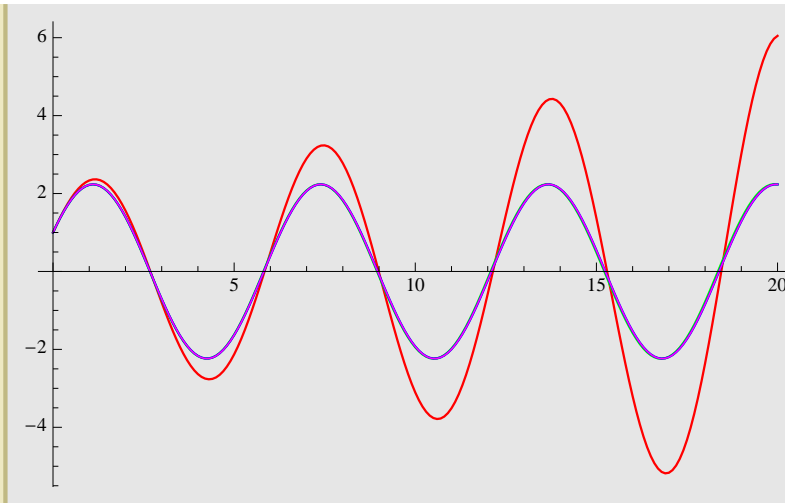
Out[158]=



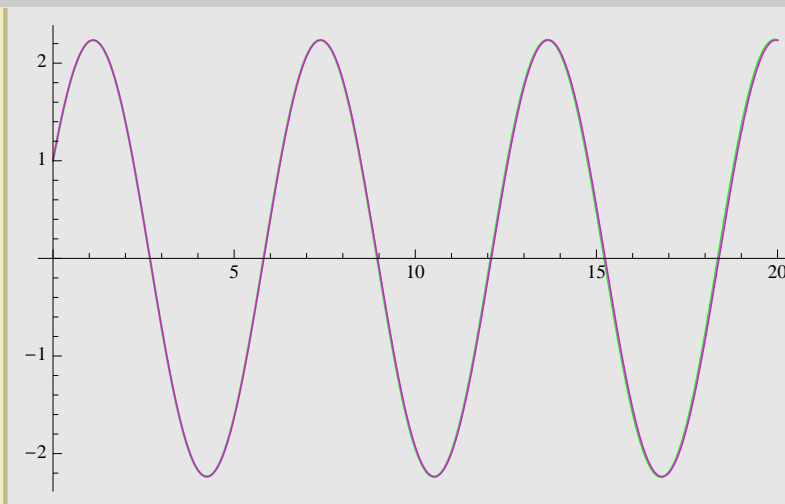
Out[159]=



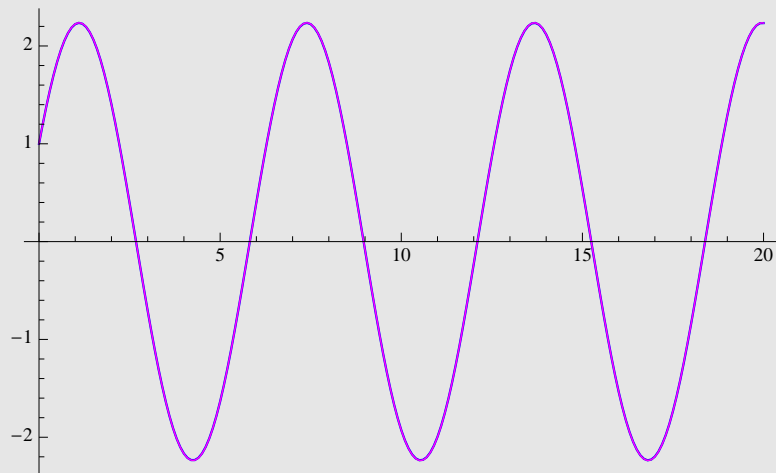
Out[160]=



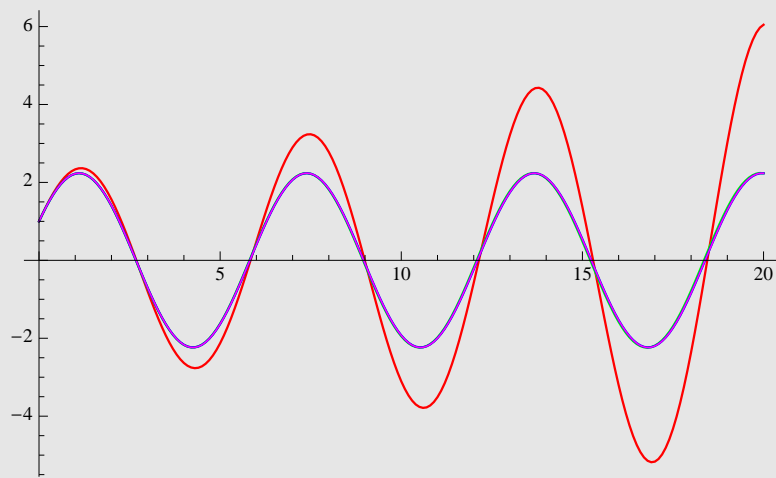
Out[150]=



Out[151]=



Out[152]=

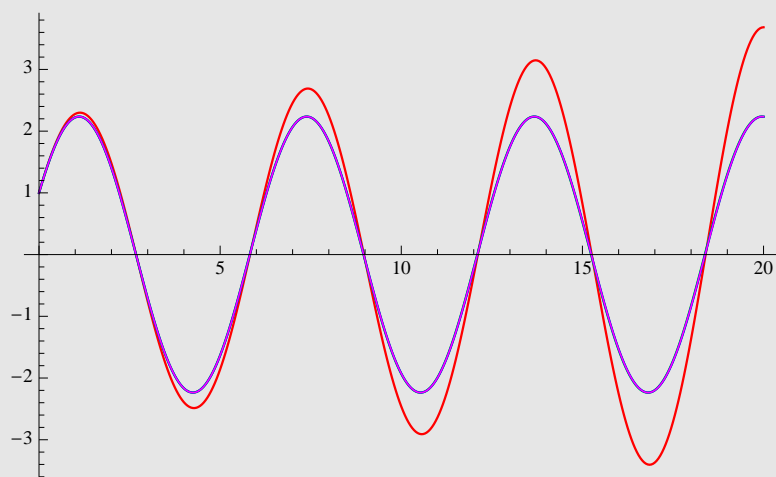


Red : Euler Green : Heun, light Green : Runge Kutta Yellow : analytic

In[177]:=

```
eug = eulergraph[.05, 400, 1, 2];
heg = Heungraph[.05, 400, 1, 2];
rug = RKgraph[.05, 400, 1, 2];
ext = agraph[1, 2, 20];
Show[eug, heg, rug, ext]
```

Out[181]=

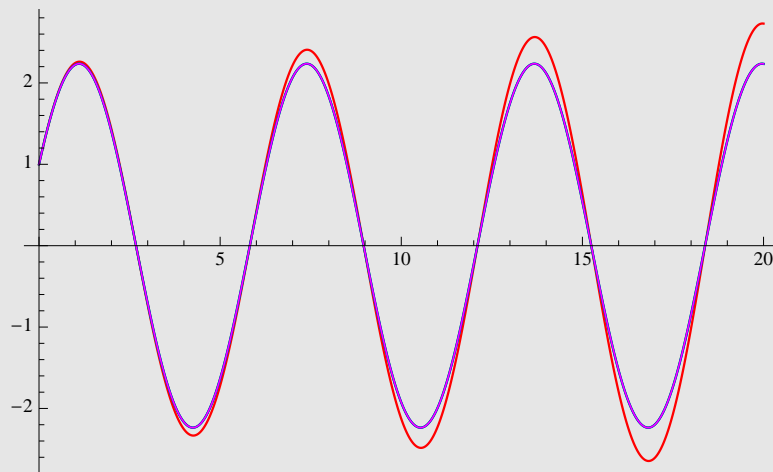


Red : Euler, Green : Heun, light Green : Runge Kutta, Yellow : analytic

In[195]:=

```
eug = eulergraph[.02, 1000, 1, 2];
heg = Heungraph[.02, 1000, 1, 2];
rug = RKgraph[.02, 1000, 1, 2];
ext = agraph[1, 2, 20];
Show[eug, heg, rug, ext]
```

Out[199]=

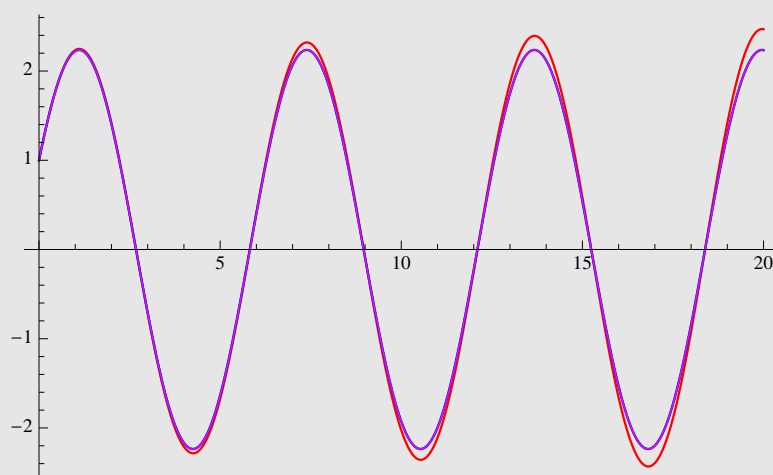


Red : Euler, Green : Heun, light Green : Runge Kutta, Yellow : analytic

In[200]:=

```
eug = eulergraph[.01, 2000, 1, 2];
heg = Heungraph[.01, 2000, 1, 2];
rug = RKgraph[.01, 2000, 1, 2];
ext = agraph[1, 2, 20];
Show[eug, heg, rug, ext]
```

Out[204]=

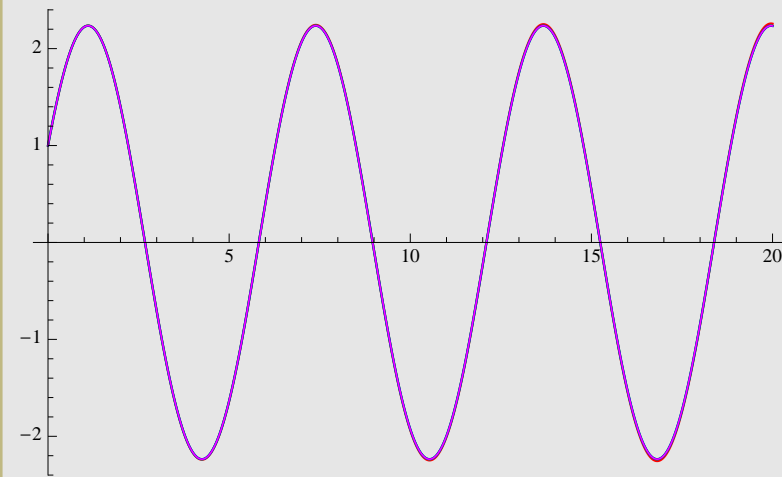


Red : Euler, Green : Heun, light Green : Runge Kutta, Yellow : analytic

In[205]:=

```
eug = eulergraph[.001, 20 000, 1, 2];
heg = Heungraph[.001, 20 000, 1, 2];
rug = RKgraph[.001, 20 000, 1, 2];
ext = agraph[1, 2, 20];
Show[eug, heg, rug, ext]
```

Out[209]=



Red : Euler **Green : Heun**, **light Green : Runge Kutta** **Yellow : analytic**

The approximation of the euler method is worst. Even for large $N_{\max} = 2000$ ($dt = 0.01$), the graph in euler method doesn't superpose on the graph of analytical solution. Finally I have succeeded to have perfect approximation for Euler Method taking $N_{\max} = 20000$. Comparatively, Heun Method is faster. For small N_{\max} like 200, it has very good approximation. And Runge-Kutta Method is the fastest. The graph of both Heun and Runge Kutta superposed with each other so well that we can't really differentiate them.