## Midterm Project

### Numerical Error Comparison

a) Exact solutions:

b) Euler’s Method

Red lines are the graph of exact solution, and the dots in different colors are numerical solutions.

Mathematica Code:

//Set the colors in plotting:

ColorTable={

RGBColor[0.992157, 0.337255, 0.282353],

RGBColor[0.968627, 0.596078, 0.0901961],

RGBColor[0.921569, 0.835294, 0.0392157],

RGBColor[0.662745, 0.85098, 0.0235294],

RGBColor[0.0392157, 0.870588, 0.101961],

RGBColor[0.054902, 0.921569, 0.745098],

RGBColor[0.0588235, 0.717647, 0.882353],

RGBColor[0.317647, 0.52549, 0.956863],

RGBColor[0.443137, 0.368627, 0.956863]

}

//Solve the equations:

EulerSol[x1\_,x2\_,y0\_,n\_]:=

Module[{X,Y,dx,j,x01=x1,x02=x2,y00=y0,m=n,dy},

dx=(x02-x01)/m;X=Y=Table[0,{m+1}];

X[[1]]=x01;Y[[1]]=y00;

For[j=1,j<=m,j++,

Y[[j+1]]=Y[[j]]+dx\*N[f[X[[j]],Y[[j]]]];

X[[j+1]]=X[[1]]+dx\*j];

Return[{Transpose[{X,Y}],ListPlot[Transpose[{X,Y}]]}];]

//Plot the graphs:

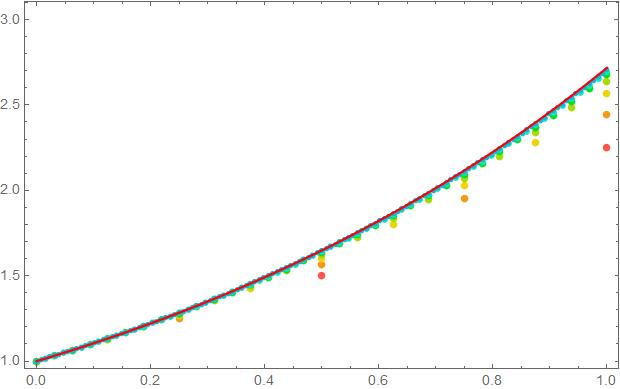
DSolveValue[{y'[x]==F3[x,y[x]],y[0]==1},y[x],x]

Show[Append[Table[ListPlot[EulerSol[0.0,1.0,1.0,2^k,F3],

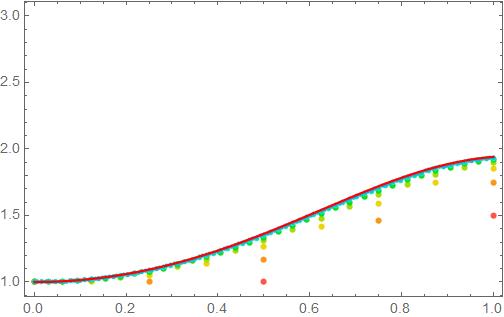
PlotStyle->ColorTable[[k]]],{k,1,7}],

Plot[-(2/(-2+Sin[2 x])),{x,0,1},PlotStyle->Red]],PlotRange->{1.0,3.0},Frame->True];

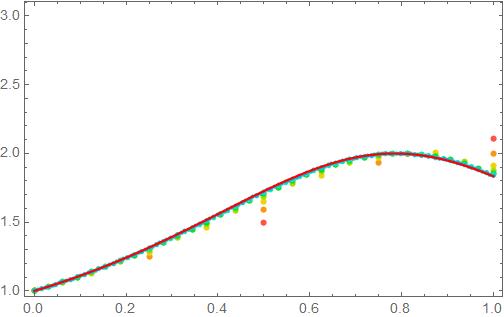
For , the result of Euler method is plotted below:



For , the result of Euler method is plotted below:



For , the result of Euler method is plotted below:



c)Runge-Kutta Method:

Mathematica Codes for order-2 and order-4:

//Order 2 Runge-Kutta

RungeKutta2O[a0\_,b0\_,\[Alpha]0\_,m0\_,fx\_]:=

Module[{a=a0,b=b0,\[Alpha]=\[Alpha]0,j,m=m0,h,Y,T,k1,k2,x,y,f=fx},

h=(b-a)/m;Y=T=Table[0,{m+1}];

T[[1]]=a;Y[[1]]=\[Alpha];

For[j=1,j<=m,j++,

k1=h\*Evaluate[f[T[[j]],Y[[j]]]];

k2=h\*Evaluate[f[T[[j]]+h,Y[[j]]+k1]];

Y[[j+1]]=Y[[j]]+(k1+k2)/2;

T[[j+1]]=a+h \*j;

];

Return[Transpose[{T,Y}]];]

//Order 4 Runge-Kutta

RungeKutta[a0\_,b0\_,\[Alpha]0\_,m0\_,fx\_]:=

Module[{a=a0,b=b0,\[Alpha]=\[Alpha]0,j,m=m0,h,Y,T,k1,k2,k3,k4,x,y,f=fx},

h=(b-a)/m;Y=T=Table[0,{m+1}];

T[[1]]=a;Y[[1]]=\[Alpha];

For[j=1,j<=m,j++,

k1=h\*Evaluate[f[T[[j]],Y[[j]]]];

k2=h\*Evaluate[f[T[[j]]+h/2,Y[[j]]+k1/2]];

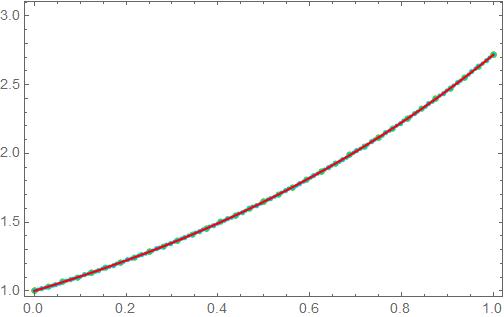
k3=h\*Evaluate[f[T[[j]]+h/2,Y[[j]]+k2/2]];

k4=h\*Evaluate[f[T[[j]]+h/2,Y[[j]]+k3]];

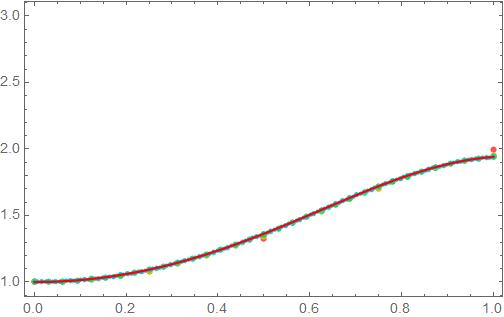
Y[[j+1]]=Y[[j]]+(k1+2k2+2k3+k4)/6;T[[j+1]]=a+h \*j;];

Return[Transpose[{T,Y}]];]

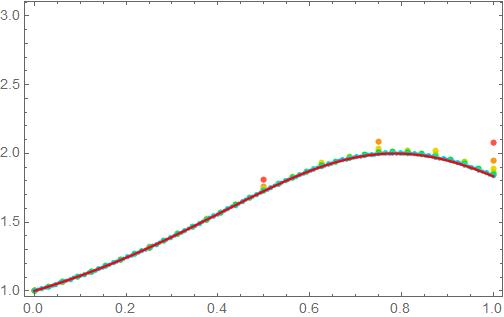
For , the result of order-4 Runge-Kutta method is plotted below:



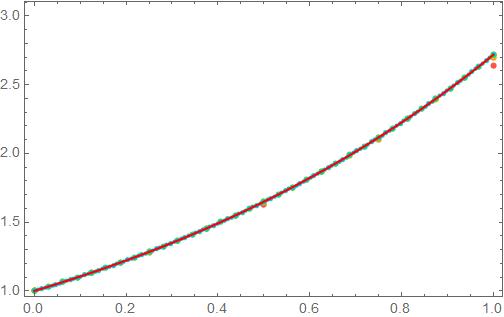
For , the result of order-4 Runge-Kutta method is plotted below:



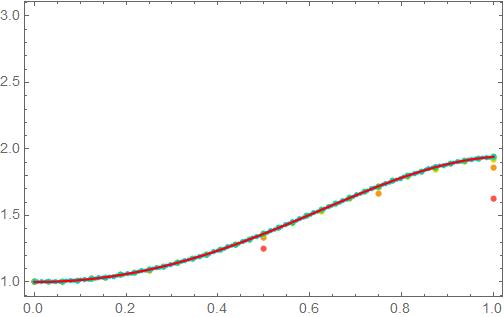
For , the result of order-4 Runge-Kutta method is plotted below:

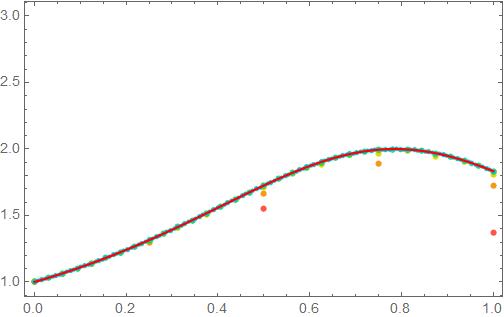


For , the result of order-2 Runge-Kutta method is plotted below:



For , the result of order-2 Runge-Kutta method is plotted below:

 For , the result of order-2 Runge-Kutta method is plotted below:



d)In regard of the legibility of graph(not too dense), compare the solutions from three methods at 1/8 step-length and the exact solution.

In the section d), all the Euler solutions are Yellow, while the 2-Order Runge-Kutta solutions are Green and the 4-Order Runge-Kutta solutions are Purple.

Mathematica code for the plot:

Show[

Plot[

(Evaluate[DSolveValue[{y'[x]==F3[x,y[x]],y[0]==1},y[x],x]]/.x->t)&[t],{t,0,1},PlotStyle->Red],

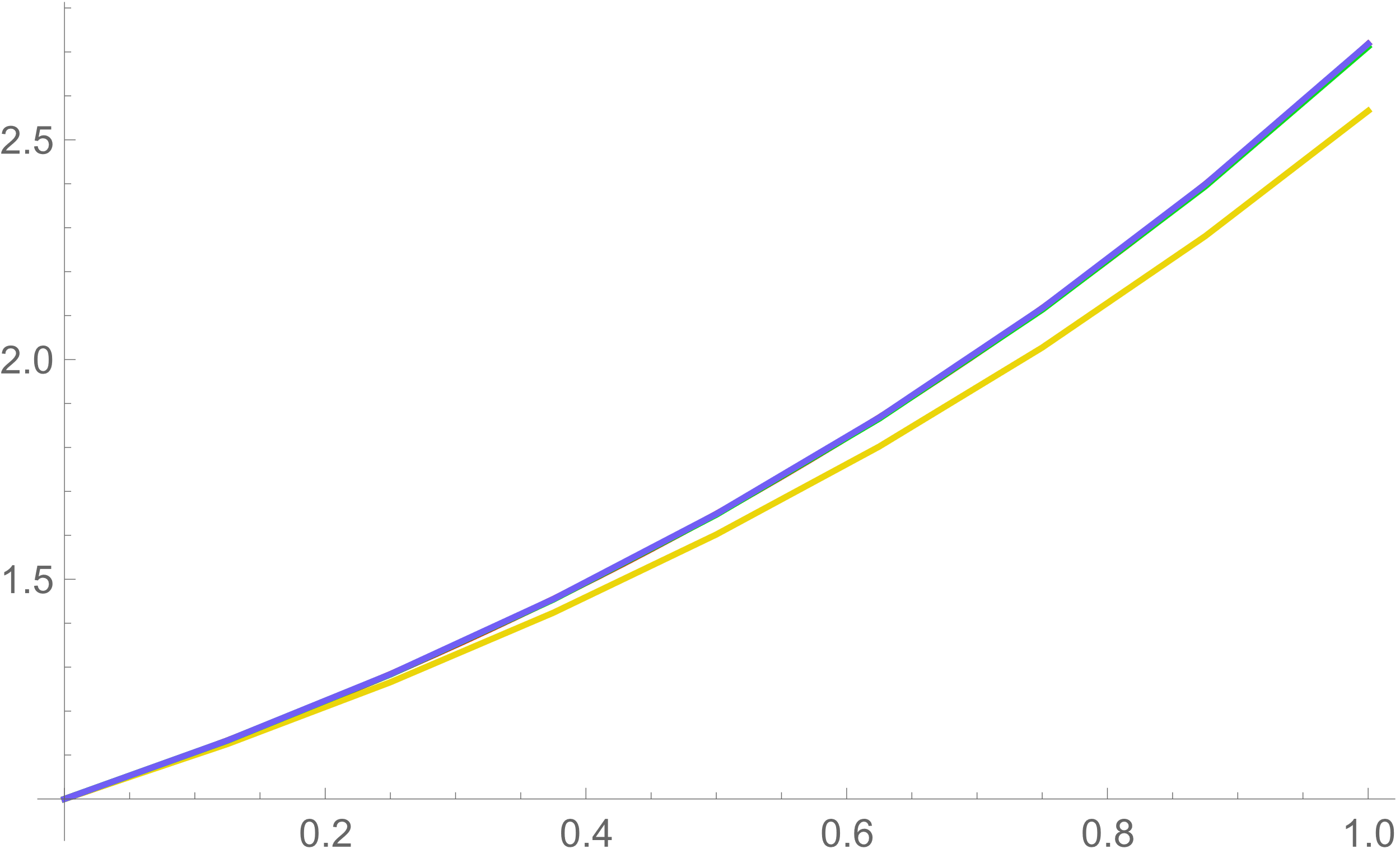
ListLinePlot[EulerSol[0.0,1.0,1.0,8,F3],PlotStyle->ColorTable[[3]]],

ListLinePlot[RungeKutta2O[0.0,1.0,1.0,8,F3],PlotStyle->ColorTable[[5]]],

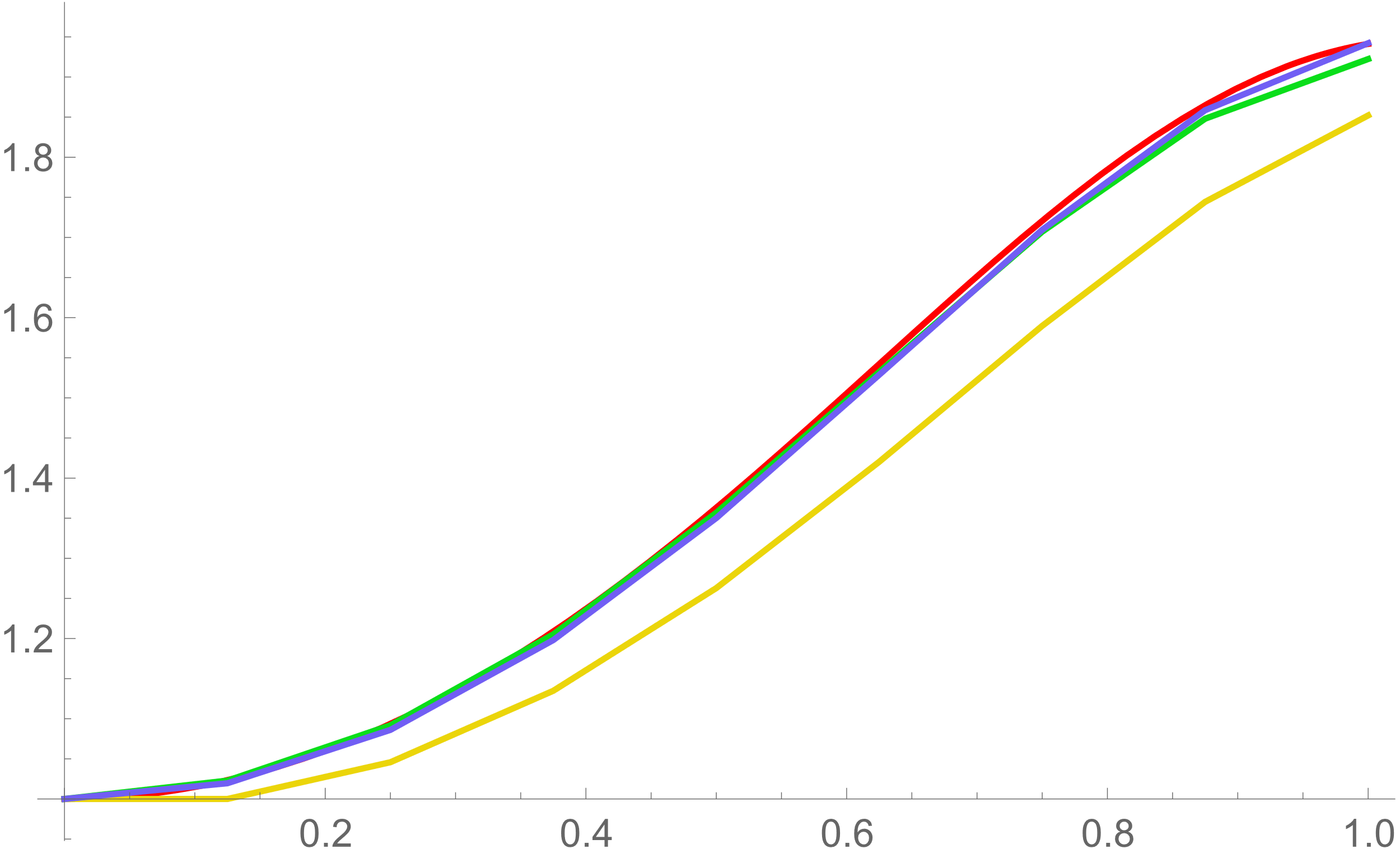
ListLinePlot[RungeKutta[0.0,1.0,1.0,8,F3],PlotStyle->ColorTable[[9]]]

]

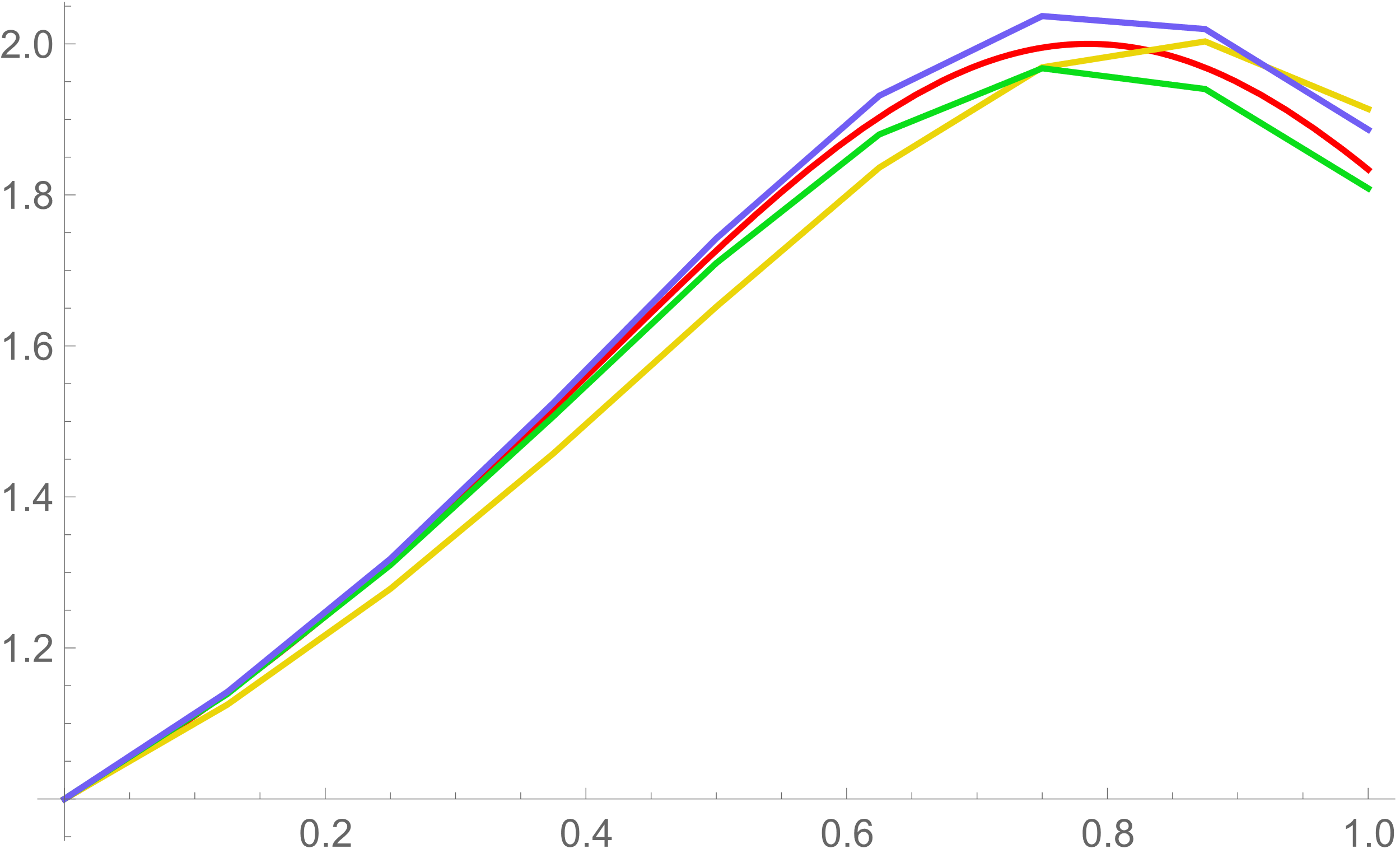
For ,



For ,



For ,



e)&f) Record the maximum error by Mathematica, and plot the log-log plot at the same time. In the log-log plot, the X-scale is 1/step, or the division.

The codes are shown below:

//ErrorPlot[y’[x,y],MethodSign]

MethodSign=1⇒Euler Method;

MethodSign=2⇒2-order Runge-Kutta method;

MethodSign=3⇒4-Order Runge-Kutta method;

ErrorPlot[F\_]:=Module[{T,V,op,f,sol,j,k,d},

op={};

V={};

S={EulerSol,RungeKutta2O,RungeKutta};

f=DSolveValue[{y'[x]==F[x,y[x]],y[0]==1},y[x],x];

sol[t\_]:=f/.x->t;

For[k=1,k<=7,k++,

d=1/2^k;

T=S[[Met]][0.0,1.0,1.0,2^k,F];

For[j=1,j<=Length[T],j++,

AppendTo[V,sol[(j-1)\*d]-T[[j]][[2]]]

];

AppendTo[op,{2^k,Max[V]}];

V=Drop[V,2^k];

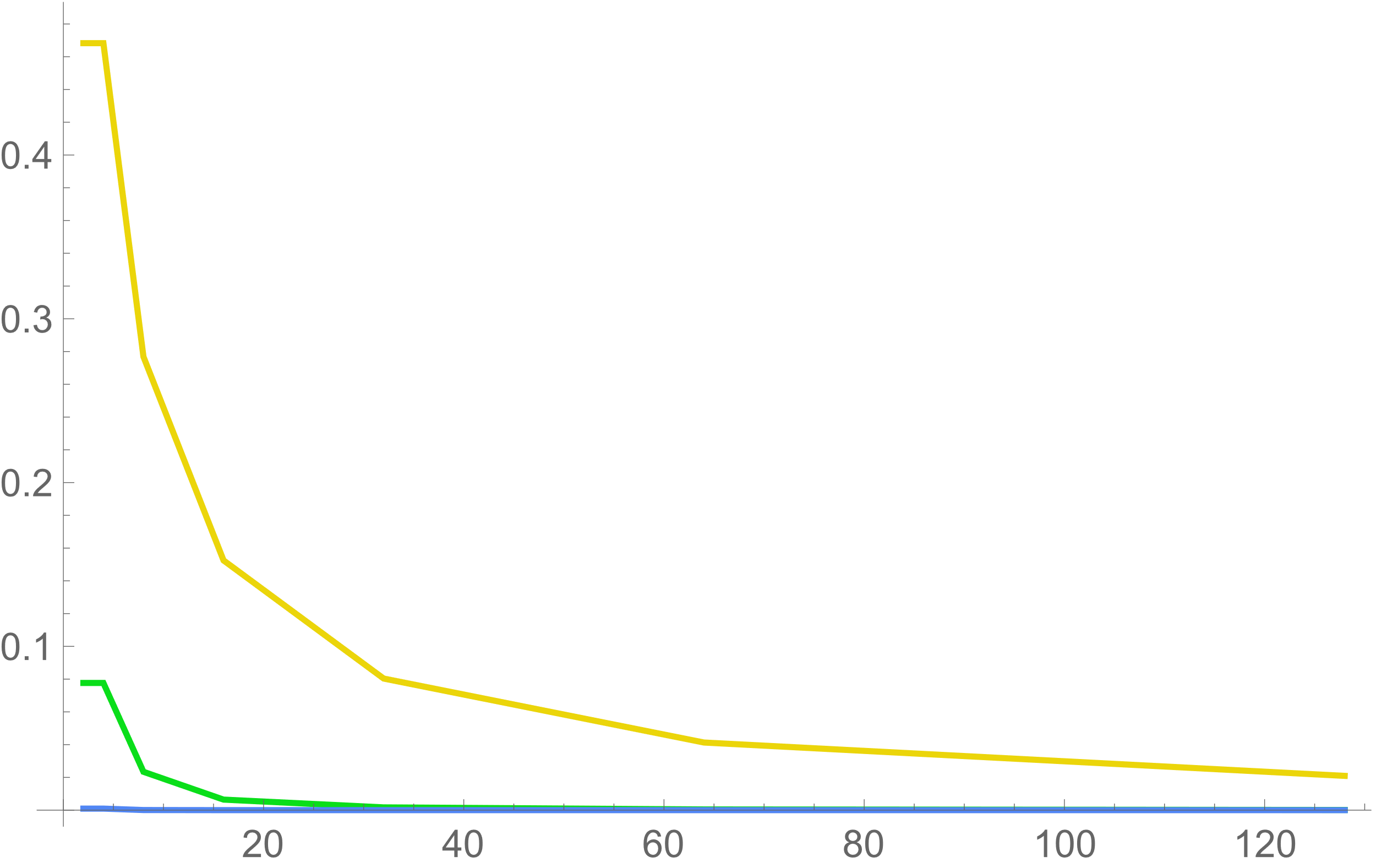
Print[V];

];

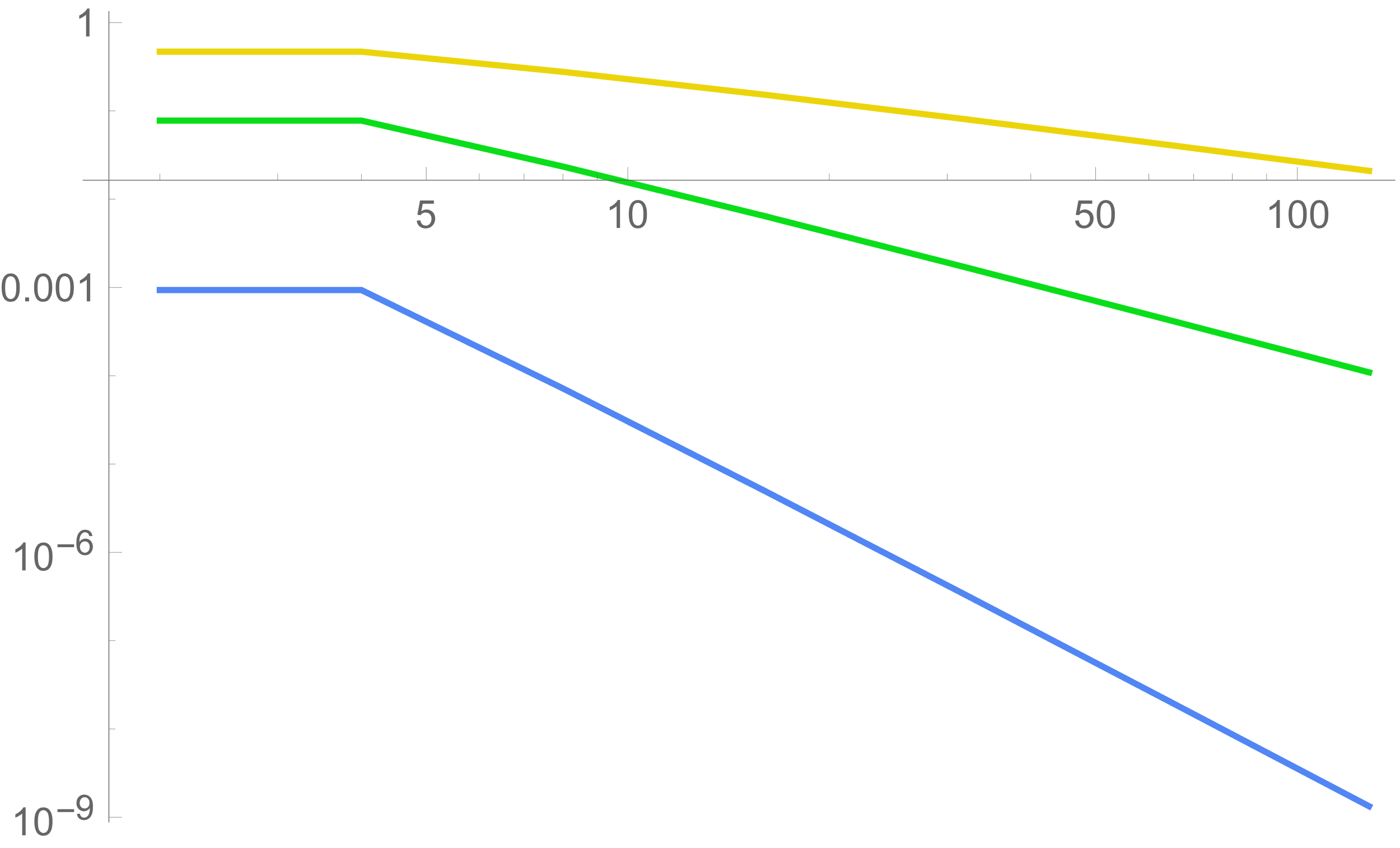
Return[Transpose[op]];]

For ,

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Euler’s Method | | | | | | | |
| Division Size | 2 | 4 | 8 | 16 | 32 | 64 | 128 |
| Maximum Error | 0.468282 | 0.468282 | 0.276876 | 0.154297 | 0.080353 | 0.041292 | 0.020937 |
| 2-order Runge-Kutta | | | | | | | |
| Maximum Error | 0.077657 | 0.077657 | 0.023426 | 0.006440 | 0.001688 | 0.000043 | 0.000011 |
| 4-order Runge-Kutta | | | | | | | |
| Maximum Error | 0.00094 | 0.00094 | 7.189×10-5 | 4.984×10-6 | 3.281×10-7 | 2.105×10-8 | 1.333×10-9 |

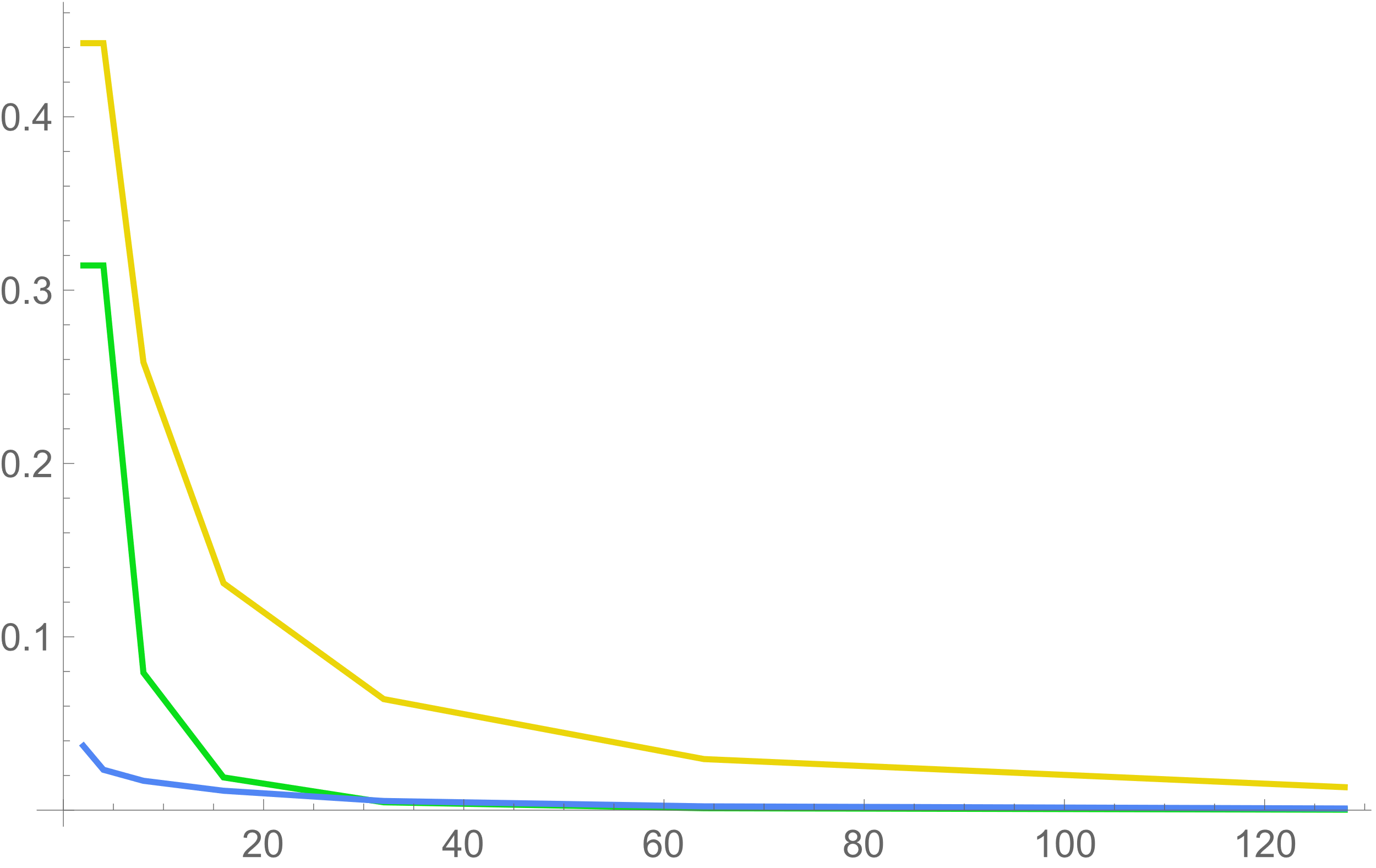


Log-Log plot:

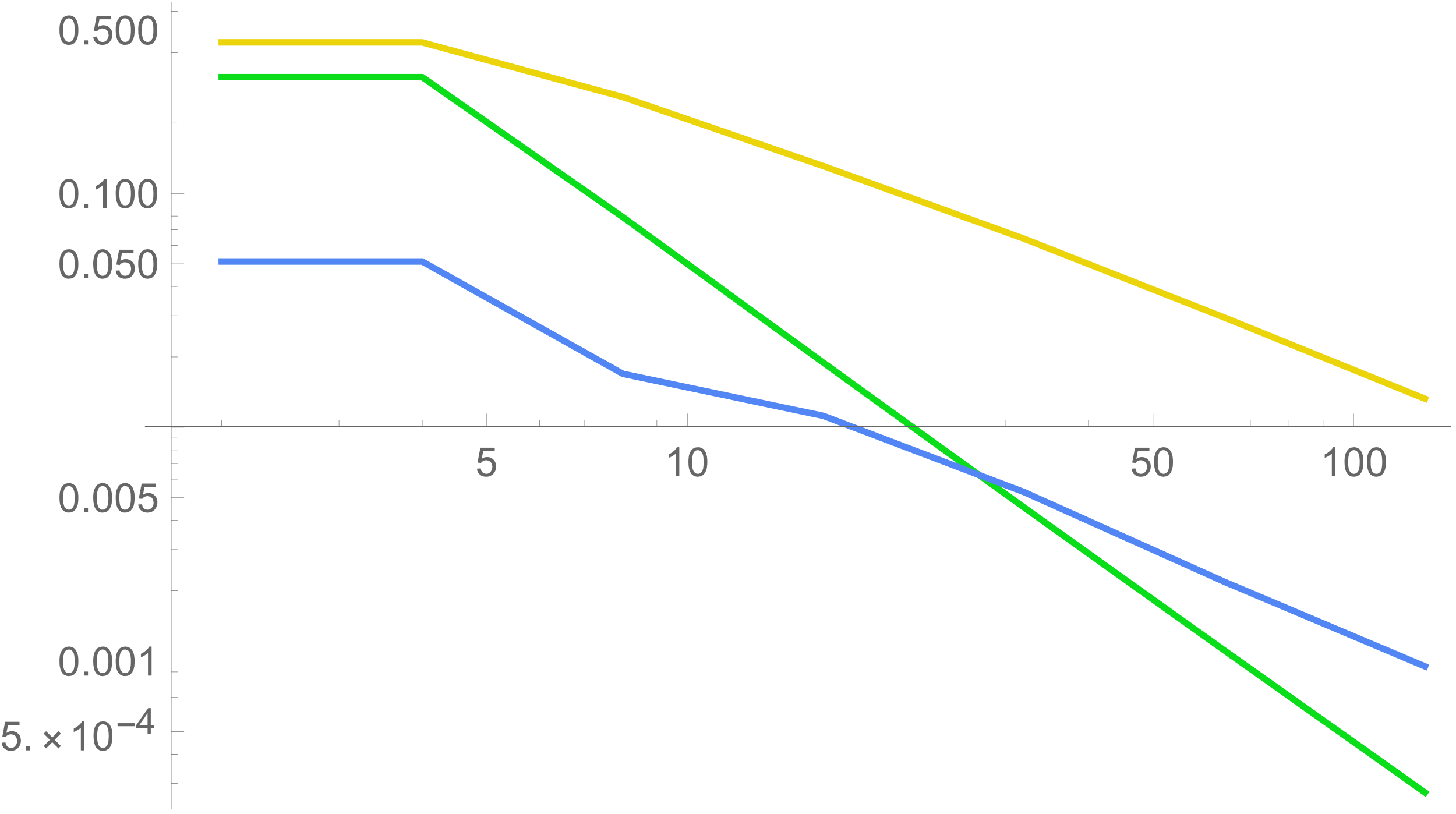


For ,

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Euler’s Method | | | | | | | |
| Division Size | 2 | 4 | 8 | 16 | 32 | 64 | 128 |
| Maximum Error | 0.4425 | 0.4425 | 0.258409 | 0.130944 | 0.064066 | 0.029464 | 0.013251 |
| 2-order Runge-Kutta | | | | | | | |
| Maximum Error | 0.314251 | 0.0314251 | 0.079260 | 0.018894 | 0.004548 | 0.001111 | 0.000274 |
| 4-order Runge-Kutta | | | | | | | |
| Maximum Error | 0.037055 | 0.023305 | 0.016934 | 0.011193 | 0.005288 | 0.002185 | 0.000950 |

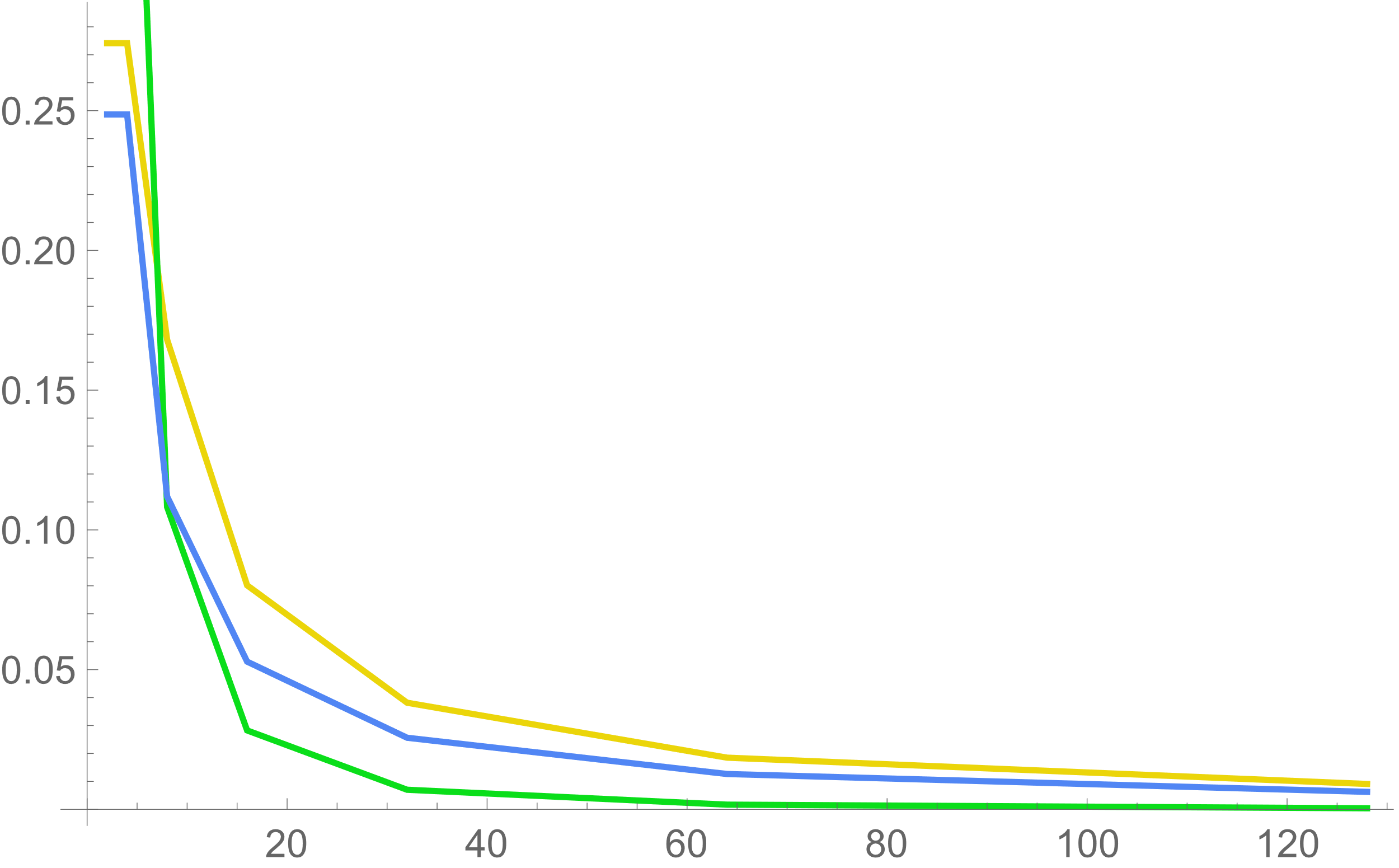


Log-Log plot:

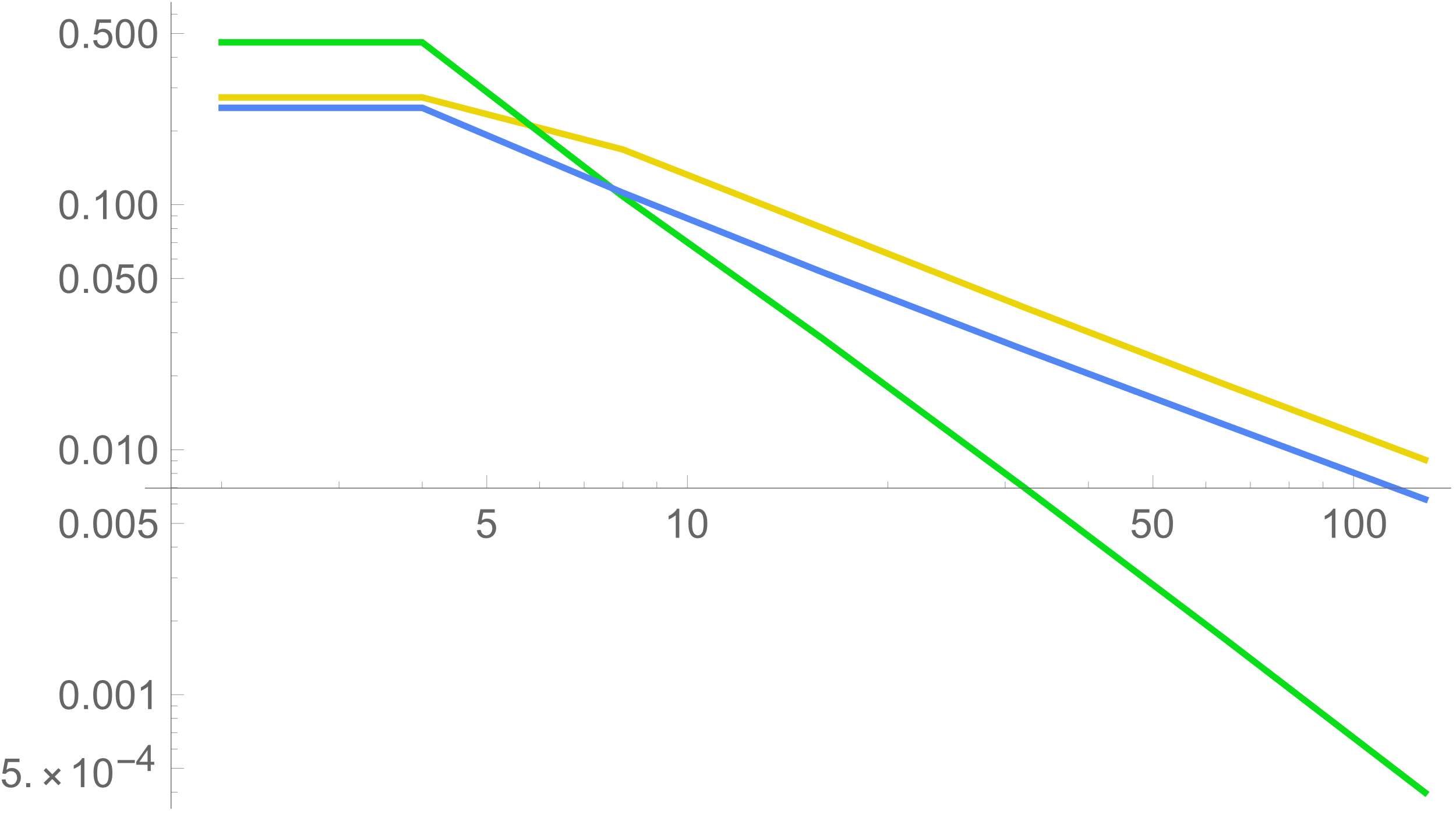


For ,

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Euler’s Method | | | | | | | |
| Division Size | 2 | 4 | 8 | 16 | 32 | 64 | 128 |
| Maximum Error | 0.27416 | 0.27416 | 0.168064 | 0.080180 | 0.038162 | 0.018504 | 0.0090956 |
| 2-order Runge-Kutta | | | | | | | |
| Maximum Error | 0.459998 | 0.459998 | 0.108191 | 0.028256 | 0.007024 | 0.001693 | 0.000398 |
| 4-order Runge-Kutta | | | | | | | |
| Maximum Error | 0.248638 | 0.248638 | 0.111969 | 0.052874 | 0.025632 | 0.012657 | 0.0062750 |



Log-Log plot:



g) Summarization

(Append by yourself)

### Oscillations of Linear Molecules

(1)

(2)(3)

For condition (2),

For condition (3),

(5)

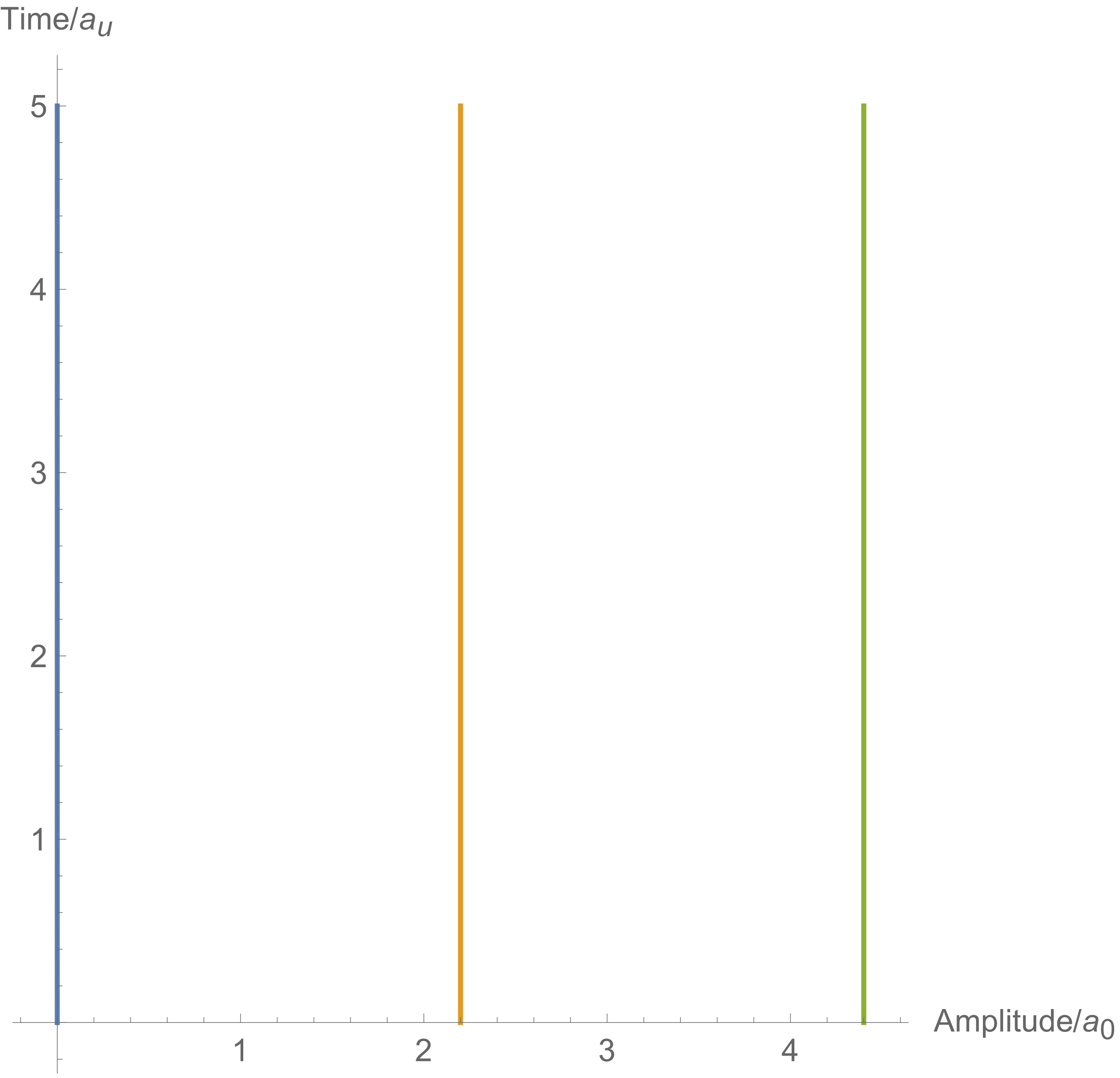
(6) Inertial mode of CO2 molecule:

(7) For the symmetric stretch, the amplitude of two outside atoms is opposite to each other, thus there is .

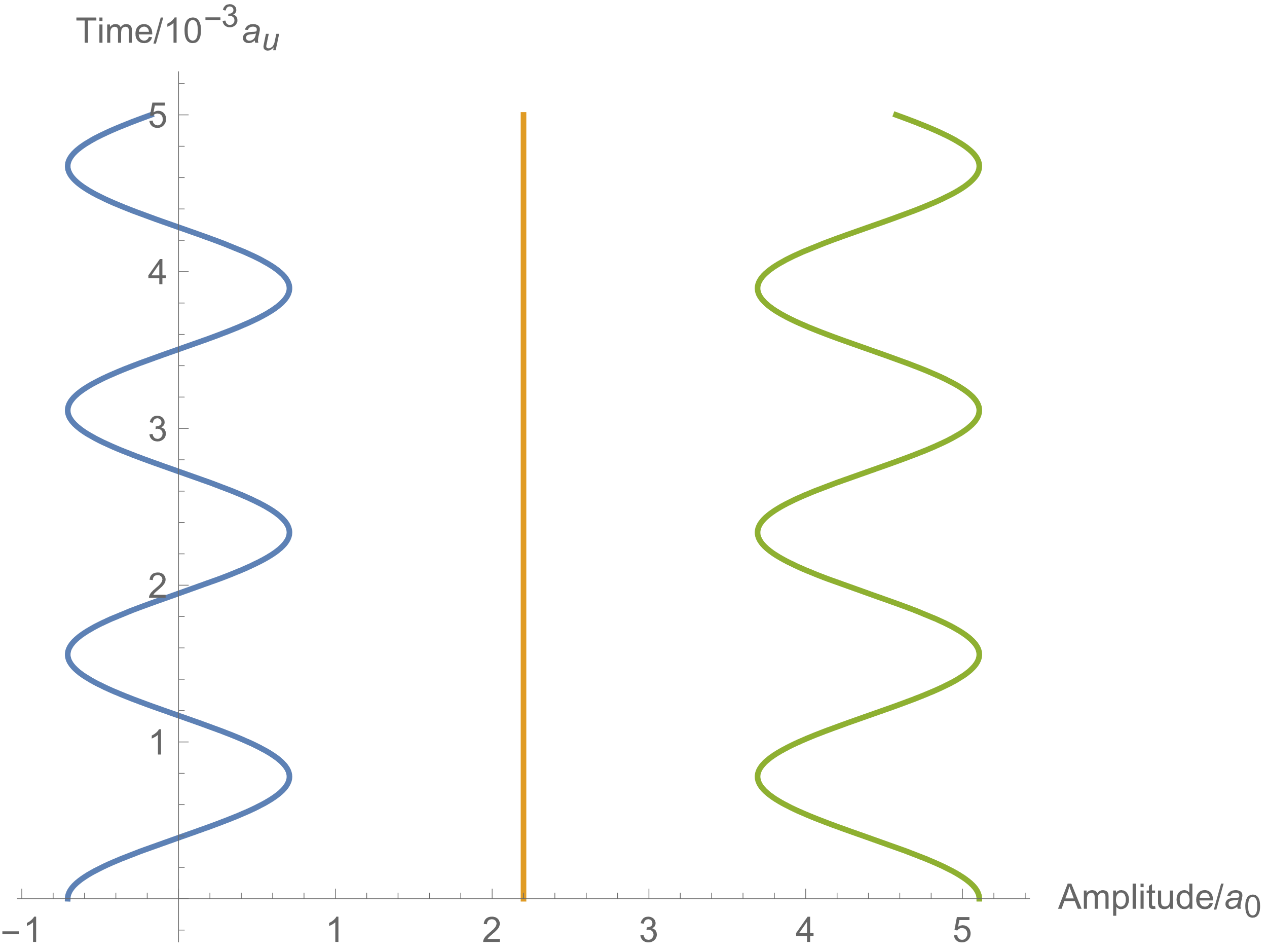
Therefore, the remaining one mode that does not satisfy the symmetricity above is asymmetric mode:

(8) Plot the graph vertically:

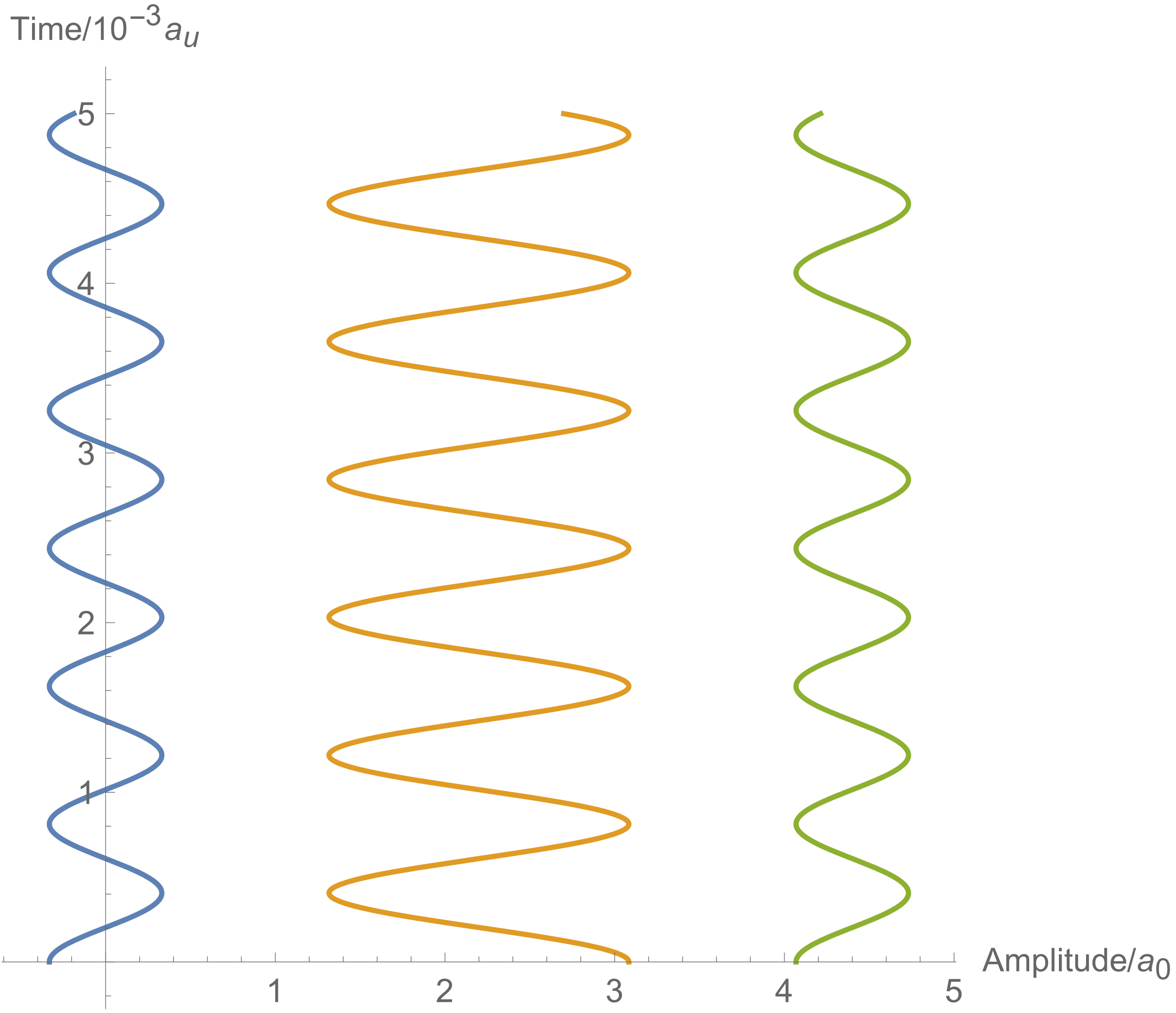
Inertial mode :



Symmetric Mode :



Asymmetric Mode :



(9) Solve the problem starting from the Lagrange function:

Obtain the coefficient matrix by the Mathematica code below:

KHCN={};

Module[{XList,CList,V,i,j},

XList={xh,xc1,xc2,xc3,xn};

CList={};

V=Collect[Expand[1/2\*Grad[k1\*(xc1-xh)^2+k12\*(xc2-xc1)^2+k23\*(xc3-xc2)^2+k3\*(xc3-xn)^2,{xh,xc1,xc2,xc3,xn}]],{xh,xc1,xc2,xc3,xn}];

For[i=1,i<=Length[V],i++,

For[j=1,j<=Length[XList],j++,

AppendTo[CList,

Coefficient[V[[i]],XList[[j]]]

]

];

AppendTo[KHCN,CList];

CList={};

];

Return[KHCN];]

Substituting all masses and force coefficients into the rigidity matrix above,

Plot all the 5 modes:

PlotModes[m\_]:=Module[{DList,FList,A,FuncList},

DList={0,2.2081,2.2716+2.2081,2.6664+2.2716+2.2081,2.1776+2.6664+2.2716+2.2081};

FList=%105[[1]];

A=%105[[2]];

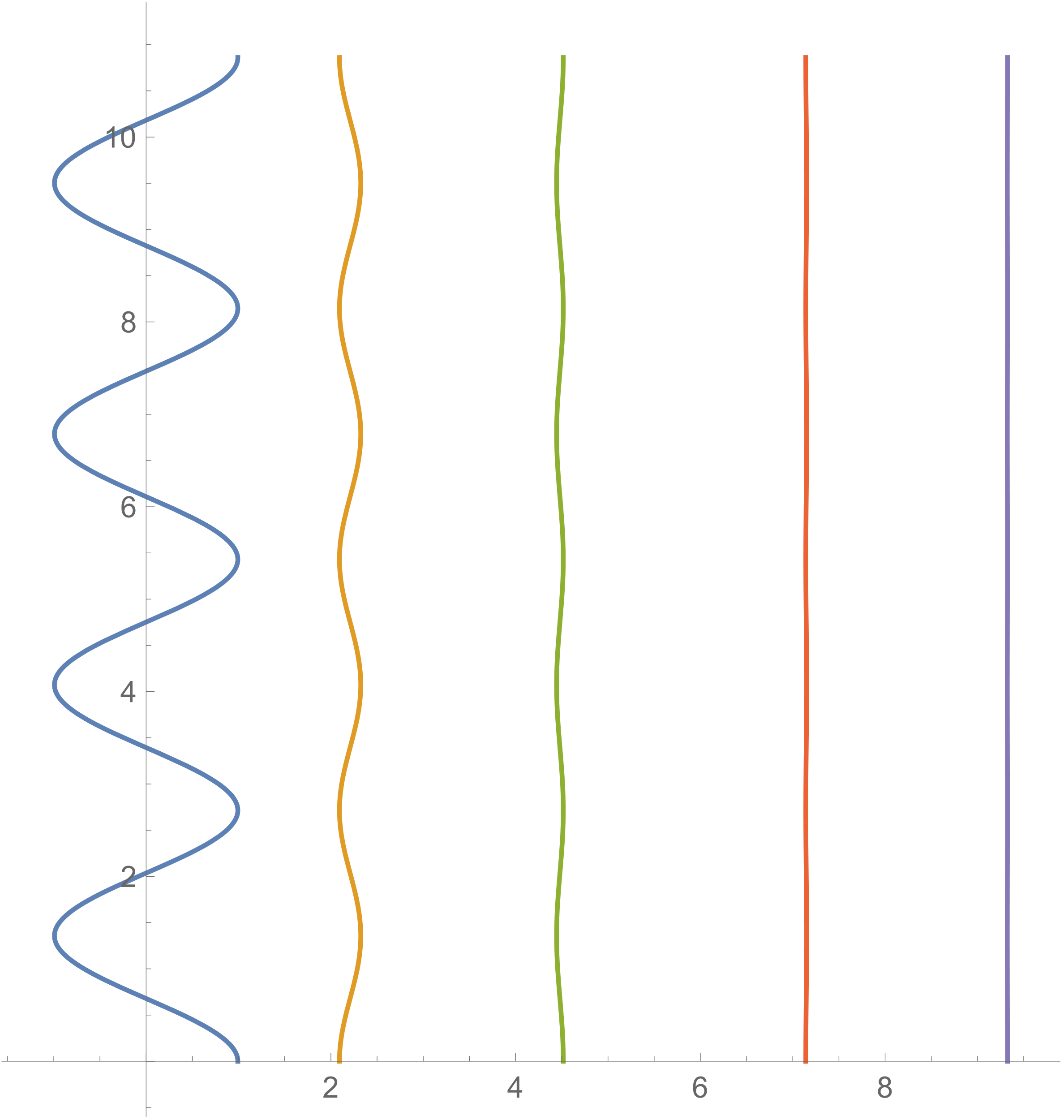
FuncList=Table[{DList[[k]]+A[[m]][[k]]\*Cos[FList[[m]]\*t\*10^4],t},{k,1,5}];

ParametricPlot[FuncList,{t,0,4\*2Pi/(10^4\*FList[[m]])},

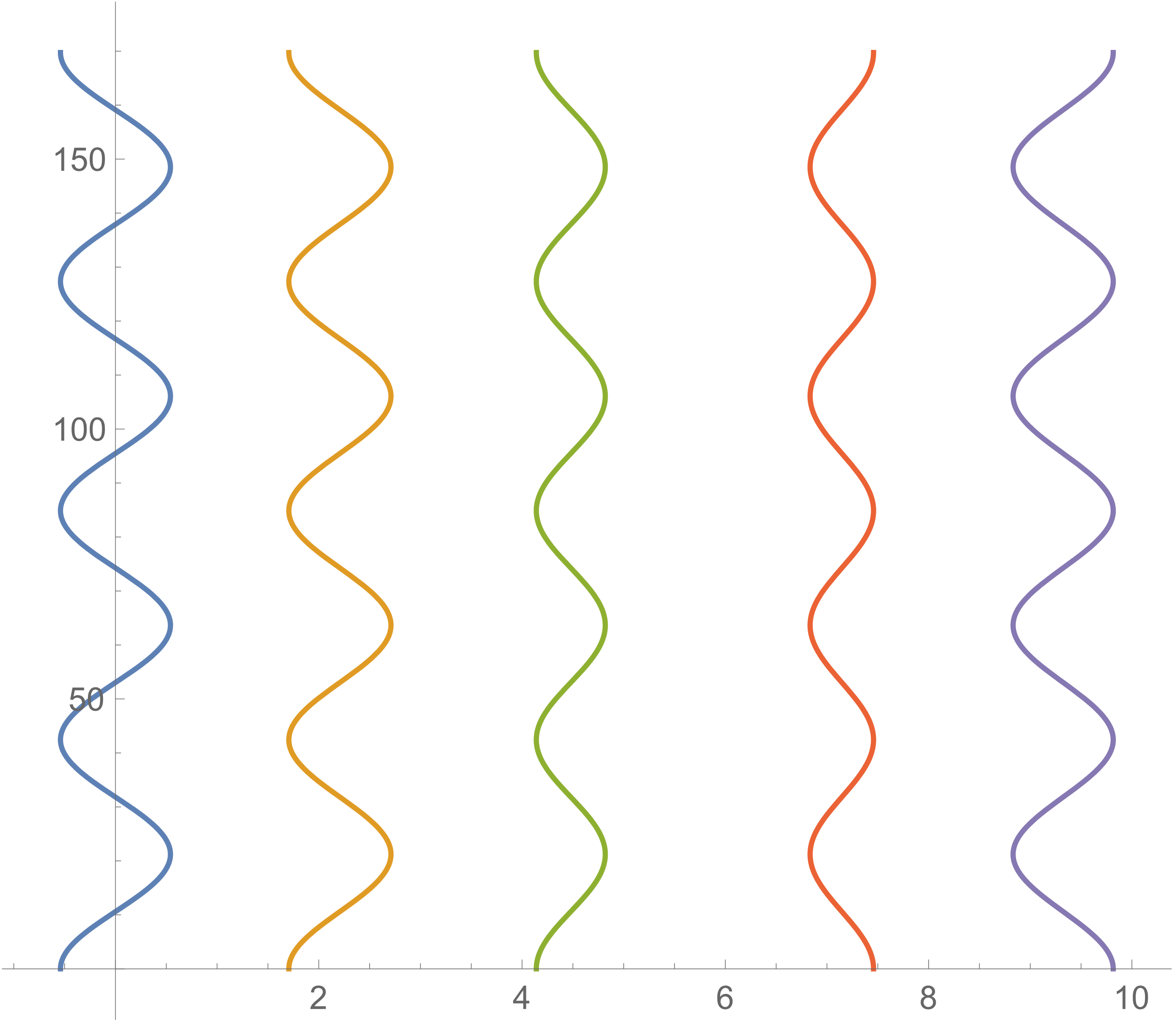
PlotRange->Full,PlotLegends->{"H","C1","C2","C3","O"},AspectRatio->FList[[m]]/FList[[1]]\*13.6]

]

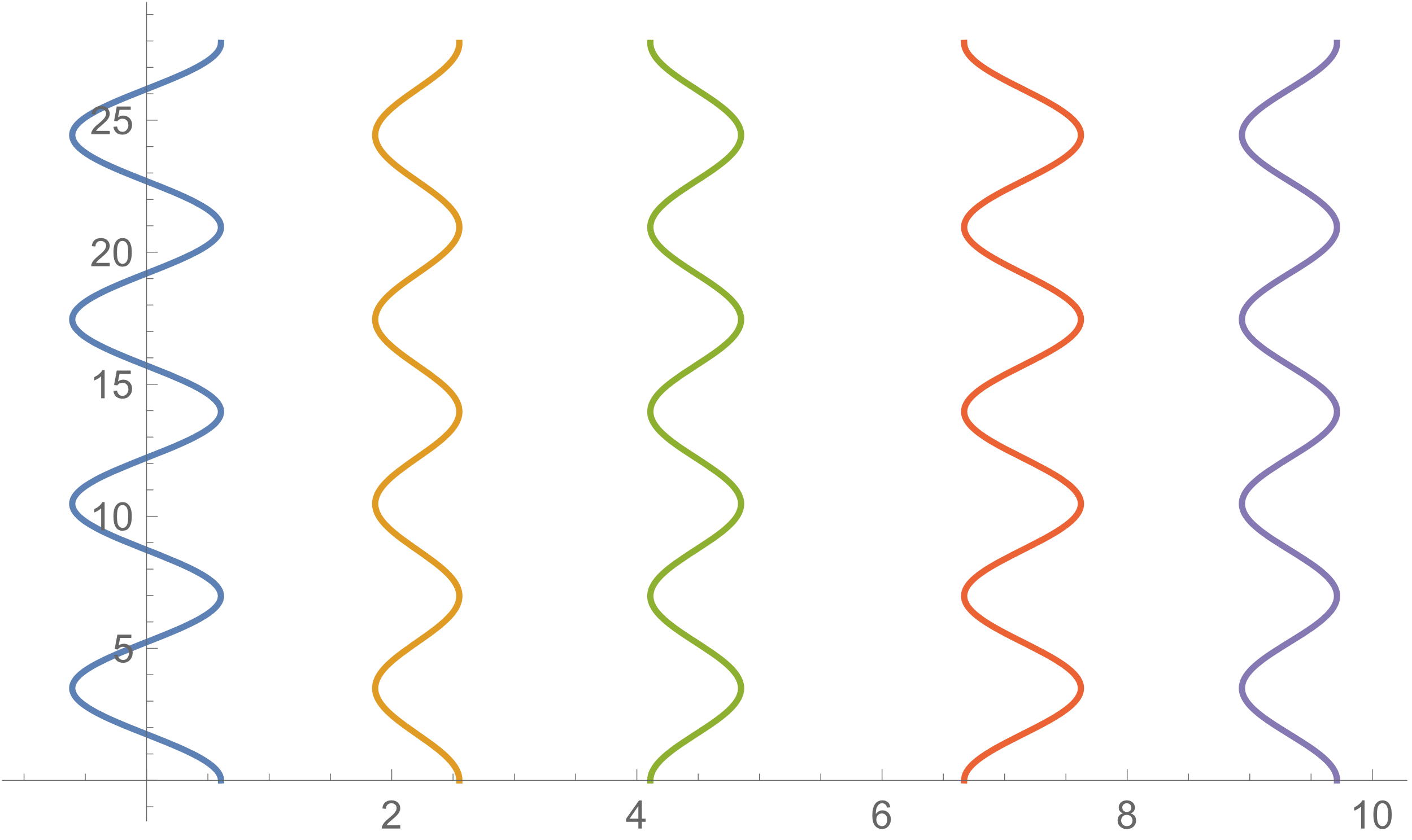
Mode 1:



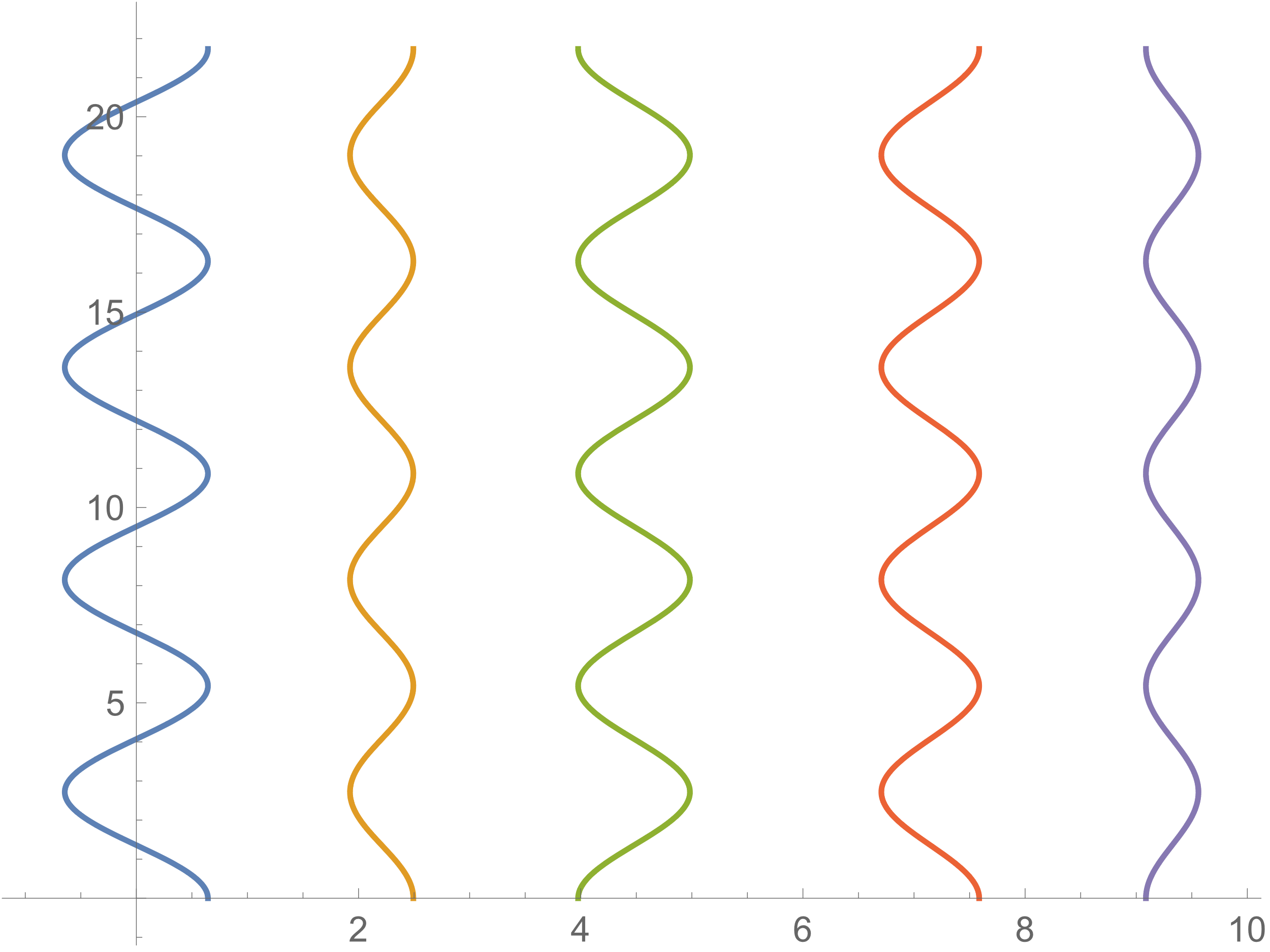
Mode 2:



Mode 3:



Mode 4:



Mode 5(Inertial Mode):

