Root Jumping to Another Location--Secant Method.

kaw@eng.usf.edu , http://numericalmethods.eng.usf.edu/mws

NOTE: This worksheet demonstrates the use of Maple to illustrate how, in the Secant method, an initial guess close to one root can jump to a location several roots away when a function is oscillatory in nature.

Introduction

Secant method [text notes][PPT] is derived from the Newton-Raphson Method. The Secant method may or may not converge, but it converges slower then Newton-Raphson method.The same drawbacks can be seen in this method as in Newton-Raphson. One of them is Root jumping. In some cases where the function f(x) is oscillating and has a number of roots, one may choose an initial guess close to a root. However, the guesses may jump and converge to some other root. The following simulation illustrates how, in the Secant method, an initial guess close to one root can jump to a location several roots away when a function is oscillatory in nature.

```
> restart;
```

Section I : Data.

Function in f(x)=0

```
> f(x):=sin(x):
```

Initial guess 1

```
> xguess1:=8.5:
```

Initial guess 2

```
> xguess2:=7.0:
```

Upper bound of range of 'x' that is desired

```
> uxrange:=-10.0:
```

Lower bound of range of 'x' that is desired

```
> lxrange:=10.0:
```

Section II: Plotting the Data.

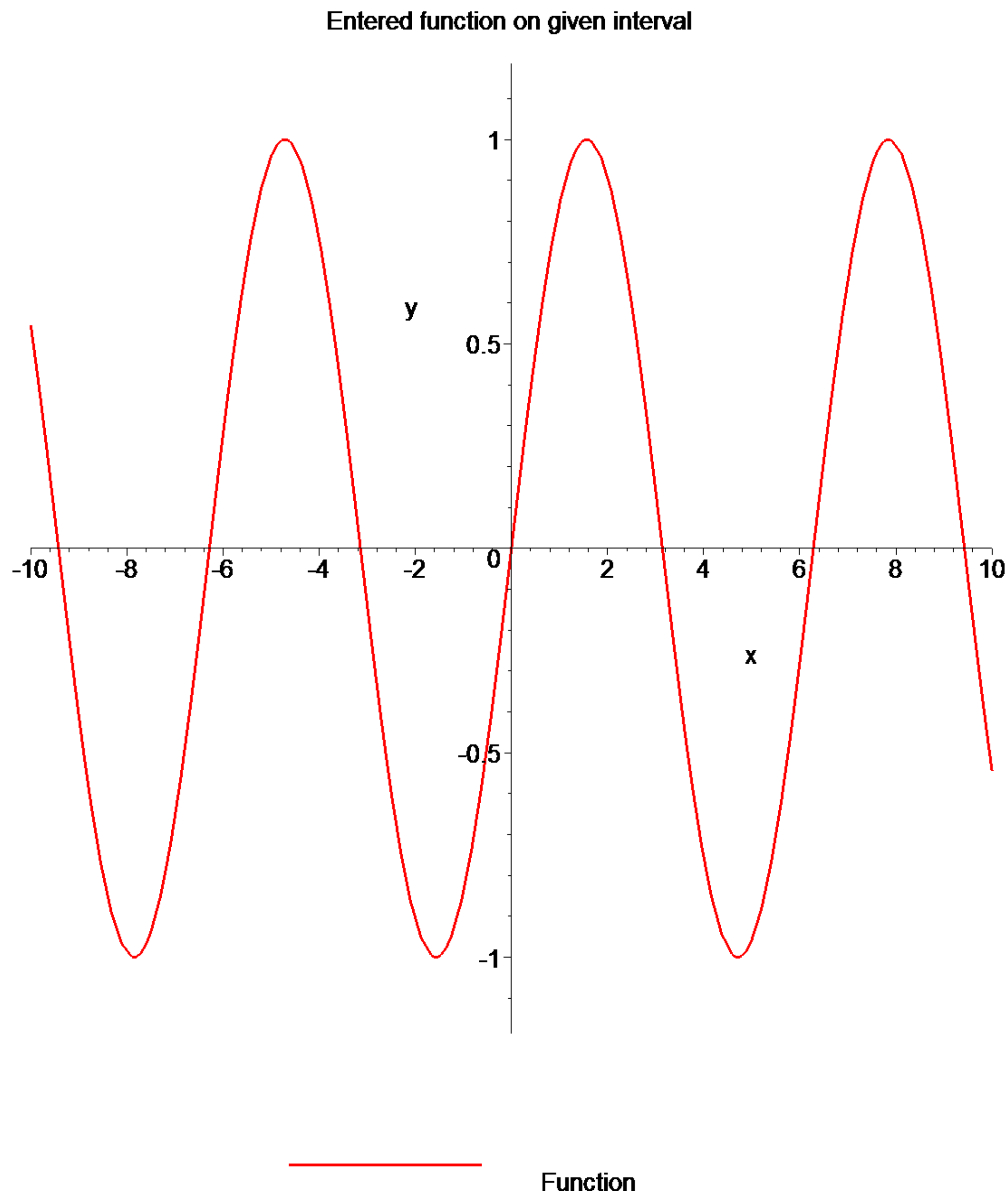
We now plot the data. The following function determines the upper and lower ranges on the Y-axis. This is done using the upper and lower ranges of the X-axis specified, and the value of the original functional at these values.

```
> yranger:=proc(uxrange,lxrange)
```

```
local i,maxi,mini,tot;
maxi:=eval(f(x),x=lxrange);
mini:=eval(f(x),x=lxrange);
for i from lxrange by (uxrange-lxrange)/10 to uxrange do
if eval(f(x),x=i)<mini then mini:=eval(f(x),x=i) end if;
if eval(f(x),x=i)>maxi then maxi:=eval(f(x),x=i) end if;
end do;
tot:=maxi-mini;
-0.1*tot+mini..0.1*tot+maxi;
end proc:
> yrange:=yranger(uxrange,lxrange):
> xrange:=lxrange..uxrange:
```

The following calls are needed to use the plot function

```
> with(plots):
Warning, the name changecoords has been redefined
> with(plottools):
Warning, the name arrow has been redefined
> plot(f(x),x=xrange,y=yrange,title="Entered function on given
interval",legend=["Function"],thickness=3);
```



Section III: Iteration 1.

So, first we choose two initial guesses of the root. It should be noted that these two guesses do not have to bracket the root. We have called the two initial guesses xguess1 and xguess2, as that will be the format for subsequent iterations. It does not matter which guess is xguess1 or xguess2 (try switching the numbers below and see what happens! You will find that one converges faster than the other). The formula mentioned in the introduction above is then applied to find the first estimate.

```
> x1:=xguess2-(eval(f(x),x=xguess2)*(xguess1-xguess2))/(eval(f(x),x=xguess1)-eval(f(x),x=xguess2));
```

$$x1 := 0.035503046$$

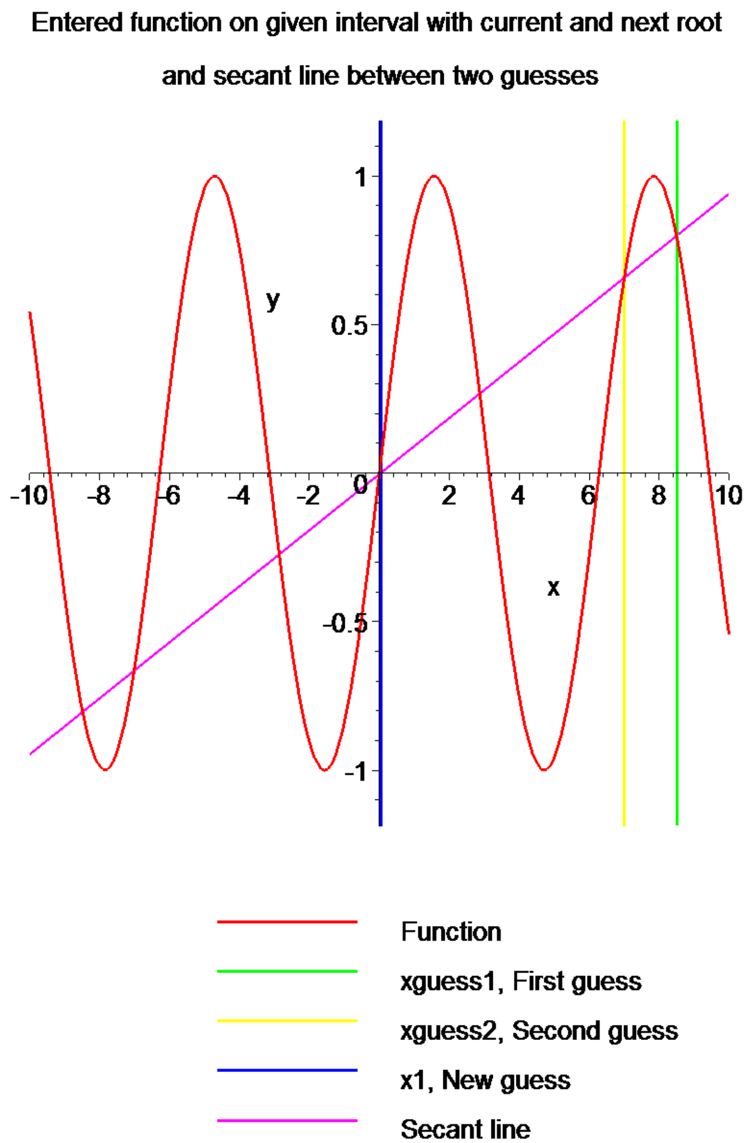
How good is that approximation? Find the absolute relative approximate error.

```
> epsilon:=abs((x1-xguess2)/x1)*100;
```

$$\varepsilon := 19616.61812$$

Although it is not necesary for the method, it is helpful to define the equation for the secant line passing through the two guesses. This function will be used for the graph.

```
> m:=(eval(f(x),x=xguess2)-eval(f(x),x=xguess1))/(xguess2-xguess1
  ):
> secantline:=m*x+eval(f(x),x=xguess2)-m*xguess2:
> plot([f(x),[xguess1,t,t=yrange],[xguess2,t,t=yrange],[x1,t,t=yr
  ange],secantline(x)],x=xrange,y=yrange,title="Entered function
  on given interval with current and next root\n and secant line
  between two guesses",legend=["Function", "xguess1, First
  guess", "xguess2, Second guess", "x1, New guess", "Secant
  line"],thickness=3);
```



Section IV: Iteration 2.

Using the same method, calculate the next estimate of the root.

```
> x0:=xguess2:
```

Estimate of the root

```
> x2:=x1-(eval(f(x),x=x1)*(x0-x1))/(eval(f(x),x=x0)-eval(f(x),x=x1));
```

$$x2 := -0.3622644370$$

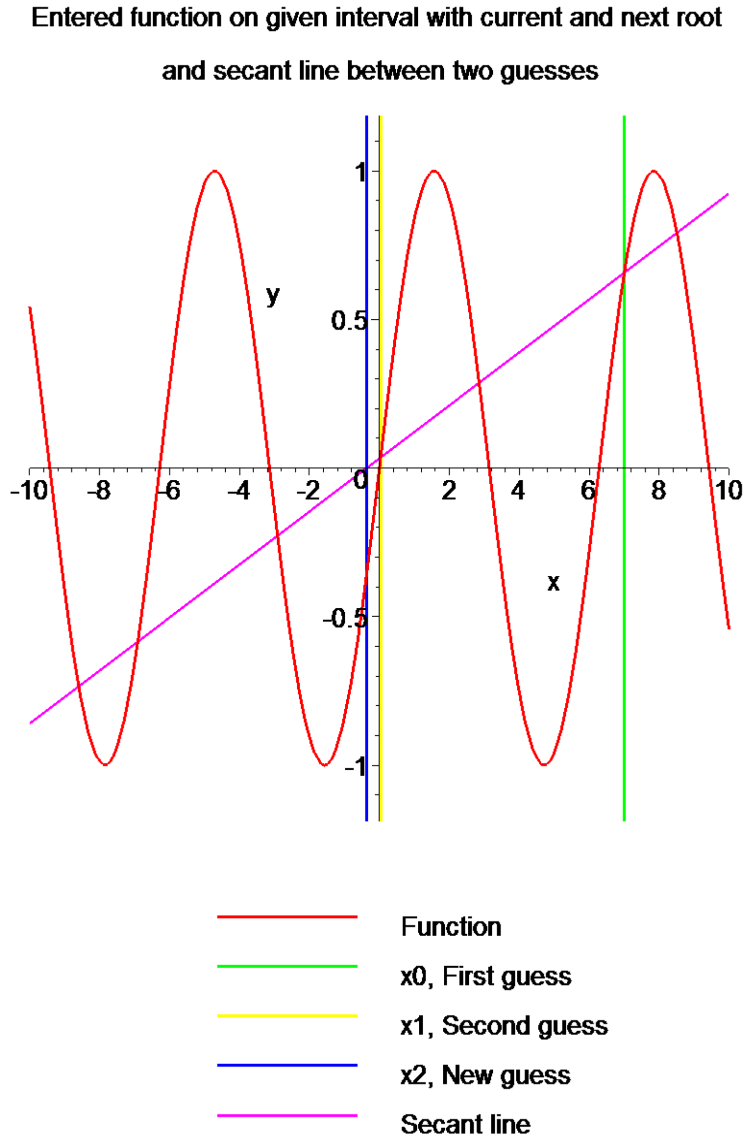
Absolute relative approximate error

```
> epsilon:=abs((x2-x1)/x2)*100;
```

$$\varepsilon := 109.8003123$$

Secant line for the graph

```
> m:=(eval(f(x),x=x1)-eval(f(x),x=x0))/(x1-x0):
> secantline:=m*x+eval(f(x),x=x1)-m*x1:
> plot([f(x),[x0,t,t=yrange],[x1,t,t=yrange],[x2,t,t=yrange],seca
  ntline(x)],x=xrange,y=yrange,title="Entered function on given
  interval with current and next root\n and secant line between
  two guesses",legend=["Function", "x0, First guess", "x1, Second
  guess", "x2, New guess", "Secant line"],thickness=3);
```



Section V: Iteration 3.

Using the same method, calculate the next estimate of the root.

Estimate of the root

```
> x3:=x2-(eval(f(x),x=x2)*(x1-x2))/(eval(f(x),x=x1)-eval(f(x),x=x2));
```

$$x3 := -0.0007098757$$

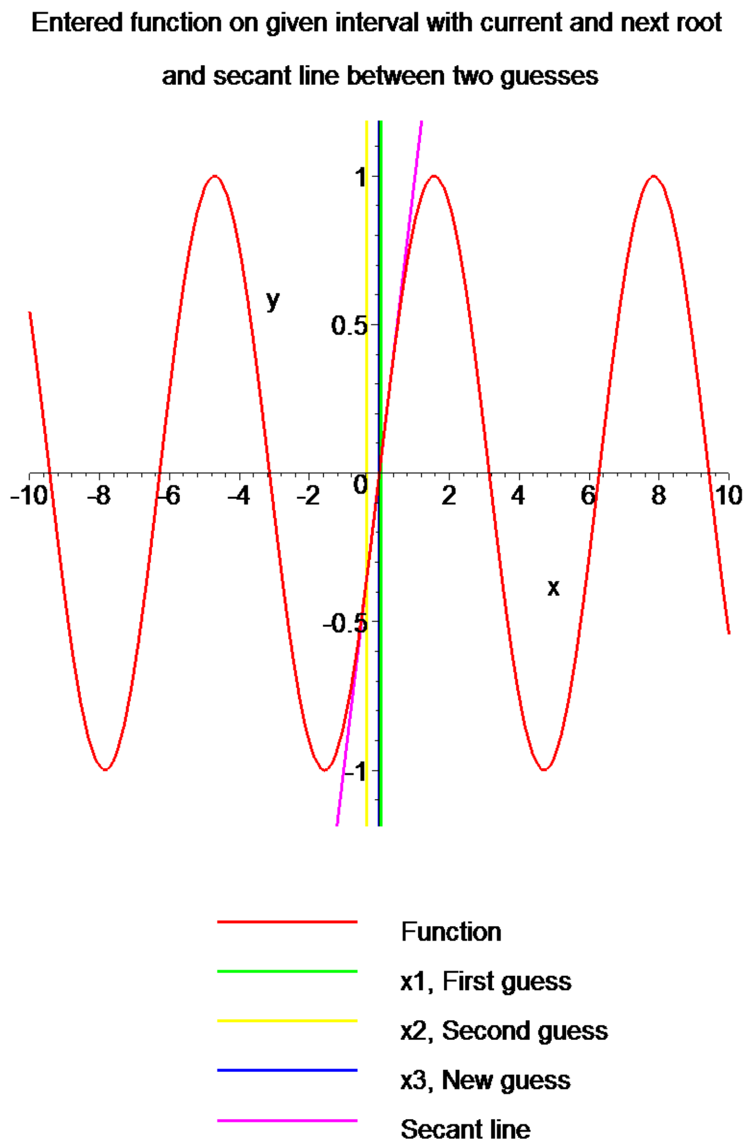
Absolute relative approximate error

```
> epsilon:=abs((x3-x2)/x3)*100;
```

$$\varepsilon := 50932.09435$$

Secant line for the graph

```
> m:=(eval(f(x),x=x2)-eval(f(x),x=x1))/(x2-x1):
> secantline:=m*x+eval(f(x),x=x2)-m*x2:
> plot([f(x),[x1,t,t=yrange],[x2,t,t=yrange],[x3,t,t=yrange],secantline(x)],x=xrange,y=yrange,title="Entered function on given interval with current and next root\n and secant line between two guesses",legend=["Function", "x1, First guess", "x2, Second guess", "x3, New guess", "Secant line"],thickness=3);
```



>

Section VI: Conclusion.

Maple helped us to apply our knowledge of numerical methods of finding roots of a nonlinear equation to simulate how, in the Secant method, an initial guess close to one root can jump to a location several roots away when a function is oscillatory in nature.

References

[1] *Nathan Collier, Autar Kaw, Jai Paul , Michael Keteltas, Holistic Numerical Methods Institute, See http://numericalmethods.eng.usf.edu/mws/gen/03nle/mws_gen_nle_txt_secant.pdf*