.

Using *Mathematica* to Prove and Animate a Property of Cubic Polynomials

Alan Horwitz
Dept.of Mathematics
Marshall University
Huntington, West Virginia 25755

Office:(304) 696-3046
email address: Horwitz@Marshall.edu
Fax number (for College of Science):(304) 696-3243

Description:

We use *Mathematica* to prove that a tangent line to the graph of any cubic polynomial with real roots, at a location halfway between any two roots, crosses the x-axis at the third root. Animation is used to show that the above property holds as the roots vary.

Abstract

In this paper, we use the capabilities of *Mathematica* for manipulating algebraic expressions to prove the following proposition:

For any cubic polynomial p(x) with real roots r1, r2 and r3, the tangent line to the graph of y = p(x) at x = (r1 + r2)/2 always has (r3 , 0) as its x- intercept.

In addition, we explain how *Mathematica* can be used to animate the graph of y = p(x) as one root varies and the other two roots remain fixed, in order to demonstrate that the tangent line, described above, behaves in the same way for any choice of real roots. To generate each frame of the animation, we use a program which, for any function f : **R** ---> **R**, draws both the graph of f and a tangent line to the graph at a location halfway between any chosen pair of zeros for f. We also generate pictures which show that the above proposition need not hold when f is chosen to be a non-cubic polynomial.

Acknowledgement: Credit for proving the proposition above goes to Professor Darryl Geller , SUNY at Stony Brook.

A proof , using Mathematica :

Claim:

Let f(x)=(x-r1)*(x-r2)*(x-r3) where r1,r2,and r3 are distinct real roots. Then the tangent line to the graph of y=f(x) at x=(r1+r2)/2 crosses the x axis at the point (r3,0).

Proof(using *Mathematica*):

We will find the equation of the tangent line at x=(r1+r2)/2 and observe that its x intercept is (r3,0).

First, we write the point-slope equation of the tangent line through (a,f(a)).

In[1]:=
```
taneqn[a_]=y-f[a]==f'[a]*(x-a)
```
Out[1]=
```
y - f[a] == (-a + x) f'[a]
```

Next, we solve the equation for y to get the right hand side of the slope intercept equation for the tangent line.

In[2]:=
```
Solve[taneqn[a],y]
```
Out[2]=
```
{{y -> f[a] - a f'[a] + x f'[a]}}
```

We can extract the expression to the right of the arrow in the output list and make it depend on the input,a, by typing in:

In[3]:=
```
tanfcn[a_]=y/.Flatten[Solve[taneqn[a],y]]
```
Out[3]=
```
f[a] - a f'[a] + x f'[a]
```

Thus, tanfcn[a] gives us the right side of the slope-intercept equation for the tangent line at x=a. Let's check that tanfcn depends on our input,a:

In[4]:=
```
tanfcn[2]
```
Out[4]=
```
f[2] - 2 f'[2] + x f'[2]
```

Note that the position of our expression within the output list is

In[5]:=

```
Position[Out[2],Out[3]]
```

Out[5]=

```
{{1, 1, 2}}
```

so we also could have defined tanfcn[a] by typing

In[6]:=

```
tanfcn[a_]=Solve[taneqn[a],y][[1,1,2]]
```

Out[6]=

```
f[a] - a f'[a] + x f'[a]
```

We now specify our cubic polynomial, f:

In[7]:=

```
f[x_]=(x-r1)*(x-r2)*(x-r3)
```

Out[7]=

```
(-r1 + x) (-r2 + x) (-r3 + x)
```

Without reentering our earlier commands, Mathematica knows that tanfcn[a] and taneqn[a] should be written in terms of this choice of f. Let's check:

In[8]:=

```
tanfcn[a]
```

Out[8]=

```
-(a ((a - r1) (a - r2) + (a - r1) (a - r3) +
       (a - r2) (a - r3))) +
  (a - r1) (a - r2) (a - r3) +
  ((a - r1) (a - r2) + (a - r1) (a - r3) +
     (a - r2) (a - r3)) x
```

To work with the tangent line to the graph of f at x=(r1+r2)/2, we substitute this value of x into tanfcn,and simplify the (otherwise unwieldy) expression :

In[9]:=

```
Simplify[tanfcn[(r1+r2)/2]]
```

Out[9]=

```
          2
(-r1 + r2)  (r3 - x)
--------------------
         4
```

At this point, you can observe that when x=r3, the tangent line crosses the x-axis. This shows that the tangent line at x=(r1+r2)/2 has (r3,0) as its x-intercept.

The programs and examples of how to use them:

The Animation package must first be loaded in order to use the "anmtshow" program.

```
(* anmtshow animates a list
   of frames of graphics pictures with respect
   to the smallest plot range along the x and y axes
   in which all of the frames fit *)

anmtshow[shwtbl_List,rngtbl_List]:=
Module[{smstx,lgstx,smsty,lgsty,xmin,xmax,ymin,ymax,
 anmtbl,pltrngs},
anmtbl=shwtbl;
pltrngs=rngtbl;
smstx[{{a_,b_},{c_,d_}}]:=a;
lgstx[{{a_,b_},{c_,d_}}]:=b;
smsty[{{a_,b_},{c_,d_}}]:=c;
lgsty[{{a_,b_},{c_,d_}}]:=d;
(* For the set of all pairs
   {{xmin,xmax},{ymin,ymax}} in list rngtbl,
   the smallest xmin and largest xmax,
   and smallest ymin and largest ymax are found *)
xmin:=Min[Map[smstx,pltrngs]];
xmax:=Max[Map[lgstx,pltrngs]];
ymin:=Min[Map[smsty,pltrngs]];
ymax:=Max[Map[lgsty,pltrngs]];
Print["The plot range along the x and y axes
 for all frames is ",{xmin,xmax}," and ",
 {ymin, ymax}," ,respectively"];
ShowAnimation[anmtbl,DisplayFunction->$DisplayFunction
,
  PlotRange->{{xmin,xmax},{ymin,ymax}},
  AspectRatio->.7];]
```

:

```
(* proc graphs the equation y=f(x)
   within the interval [from,to] along the
   x-axis and it also graphs the tangent
   line at a location x = halfway between the
   mth and nth zeros of f(x) *)

proc[f_,mzro_,nzro_,from_,to_]:=
Module[{f1,map1,map2,isert,fzro,refzro,avg,
tanfcn,intcpts,intcplist,fcrit,refcrit,
xmax1,xmin1,xmax,xmin,ymax,ymin,tangph,fcngph,
intcpgph,pntgph,txtgph},
f1[s_]=N[f/.x->s];
map1[{x->a_}]=a;
map2[a_]={a,f1[a]};
isert[a_,b_,c_]:=Insert[a,b,Position[a,c]];
(* If zrolist already exists, then the
   existing version is used and the following
   steps for finding it are skipped *)
If[Length[zrolist]==0,
fzro=NSolve[f1[x]==0,x,10];
refzro=Cases[fzro,{x-> z_}/;(Re[z]===z),5];
zrolist=Sort[N[Map[map1,refzro]]],Continue];
(* If there are fewer than m or n zeros, then
   the program will be unable to take their average,
   so it will quit *)
If[mzro>Length[zrolist] || nzro>Length[zrolist],
 Print["mzro= ",mzro,"  or  nzro= ",nzro,"  exceeds
the
 number of zeros= ",Length[zrolist]," The program
 will terminate."];zrolist={};critlist={};
 Return[ ],
 Continue];
avg=(zrolist[[mzro]]+zrolist[[nzro]])/2;
(* The tangent line halfway between the mth and nth
zero
   and its x-intercept are computed *)
tanfcn[a_,x_]=f1[a]-f1'[a]*a+f1'[a]*x;
intcpts=Map[map1,NSolve[tanfcn[avg,x]==0,x,10]];
intcplist=Map[map2,intcpts];
intcpts=Cases[intcpts,z_/;(Re[z]===z),5];
(* If the critlist already exists, then the
```

```
   version is used, and the following steps for
   finding it are skipped *)
If[Length[critlist]==0,
fcrit=NSolve[f1'[x]==0,x];
refcrit=Cases[fcrit,{x->z_}/;(Re[z]===z),5];
critlist=N[Map[map1,refcrit]],Continue];
(* The maximum and minimum values of f,
   ymax and ymin, are computed over an interval,
   [xmin1,xmax1], which includes the mth zero,
   the nth zero, the x-intercept of the tangent
   line,if there is one. They are called
   ymax and ymin  *)
If[Length[intcpts]==0,
xmax1=Max[zrolist[[mzro]],zrolist[[nzro]]];
xmin1=Min[zrolist[[mzro]],zrolist[[nzro]]],
xmax1=Max[Max[intcpts],zrolist[[mzro]],
 zrolist[[nzro]]];
xmin1=Min[Min[intcpts],zrolist[[mzro]],
 zrolist[[nzro]]]];
(* Critical points of f(x) between xmin1 and xmax1
   are considered as places where f has absolute
   extrema. *)
critlist=Cases[critlist,z_/;(z<=xmax1 && z>=xmin1),5];
(*Print["xmax1=",xmax1," xmin1=",xmin1,
 " mthzro=",zrolist[[mzro]],"
nthzro=",zrolist[[nzro]],
 " intcpts are ",intcpts," critlist=",critlist,
 " zrolist=",zrolist]*)
xmax=Max[Max[zrolist],from,to];
xmin=Min[Min[zrolist],from,to];
ymax=Max[Max[Map[f1,critlist]],f1[xmin1],f1[xmax1]];
ymin=Min[Min[Map[f1,critlist]],f1[xmin1],f1[xmax1]];
(* The graph of y=f(x), the tangent line and point of
   tangency , and the x intercept of the tangent line
   are all plotted. The graphics output is
   suppressed *)
tangph=Plot[tanfcn[avg,x],{x,xmin,xmax},
 DisplayFunction->Identity];
tangph=isert[tangph,RGBColor[0,.8,1],Line[_]];
fcngph=Plot[f1[x],{x,xmin,xmax},
 DisplayFunction->Identity];
fcngph=isert[fcngph,RGBColor[1,0,.8],Line[_]];
```

```
fcngph=isert[fcngph,AbsoluteThickness[1.5],Line[_]];
intcpgph=ListPlot[intcplist,PlotStyle->{PointSize[.03]
},
  DisplayFunction->Identity];
intcpgph=isert[intcpgph,RGBColor[.6,0,.4],Point[_]];
pntgph=ListPlot[{{avg,f1[avg]}},

PlotStyle->{PointSize[.03]},DisplayFunction->Identity]
;
pntgph=isert[pntgph,RGBColor[.5,.5,0],Point[_]];
gphs=Show[fcngph,tangph,intcpgph,pntgph,
  AxesLabel->{"x"," "},PlotLabel->"y ="f1[x],
  DefaultFont->{"System",14}];
pltrng=Join[{{{xmin,xmax},{ymin,ymax}}},
 {FullOptions[gphs,PlotRange]}];
(* The suppressed graphics output in gphs
   is joined to a list, gphtbl, containing
   graphics output from other runs of proc.
   The list, pltrng, is joined to a list,
   totpltrng, containing information about
   the plot range of graphics output from
   other runs of proc *)
gphtbl=Join[gphtbl,{gphs}];
totpltrng=Join[totpltrng,pltrng];
zrolist={};critlist={};]
```

To draw a graph using proc without doing animation, we define a cubic polynomial p, and then choose specific values for the arguments. We must specify totpltrng={ } and gphtbl={ } to avoid error messages from the Join commands in the program. We graph the polynomial, f(x)=(x-2)(x-3)(x-1) along the x-axis from x=0 to x=4, and we graph the tangent line to f at a location halfway between the 1st & 2nd zeros of f, that is, at x=(1+2)/2.

In[1]:=

```
p[t_,a_,b_,x_]=(x-a)*(x-b)*(x-t)
```

Out[1]=

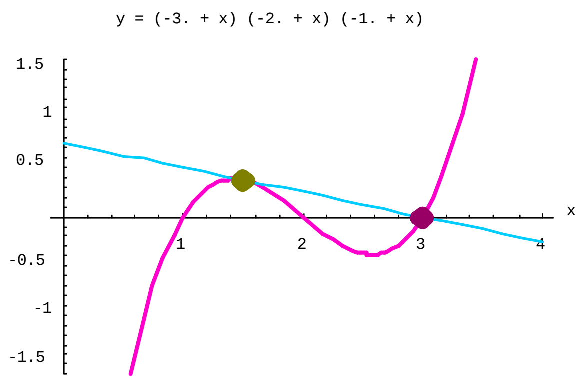
```
(-a + x) (-b + x) (-t + x)
```

In[2]:=

```
totpltrng={};gphtbl={};
proc[f=p[1,2,3,x],mzro=1,nzro=2,from=0,to=4]
```

We use the Show command to view the contents of gphtbl. Because the graphics output was suppressed in proc with the plot option, DisplayFunction -> Identity, we must unsuppress it with the plot option, DisplayFunction -> $DisplayFunction, in order to get a picture.

In[3]:=
Show[gphtbl,DisplayFunction->$DisplayFunction]

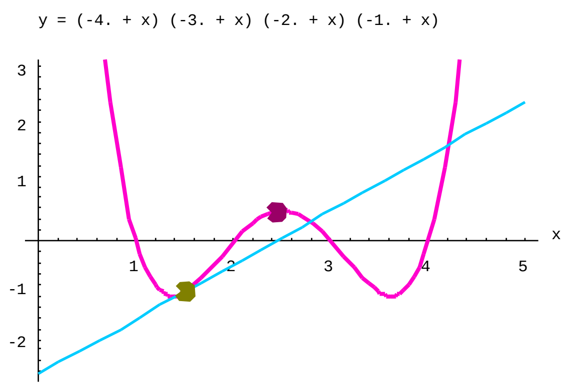


We use proc to draw a similar picture for the 4th degree polynomial, q(x)=(x-1)(x-2)(x-3)(x-4) .

In[4]:=
q[x_]=(x-1)*(x-2)*(x-3)*(x-4)
Out[4]=
(-4 + x) (-3 + x) (-2 + x) (-1 + x)

In[5]:=
totpltrng={};gphtbl={};
proc[f=q[x],mzro=1,nzro=2,from=0,to=5]

In[6]:=
Show[gphtbl,DisplayFunction->$DisplayFunction]



Observe that the tangent line halfway between the 1st zero, x=1, and the 2nd zero, x=2, does not cross the x-axis at any of the remaining zeros.

Because proc uses the NSolve command, it is unable to find zeros of f(x) for transcendental functions, such as Cos[x], Log[x], etc. It only finds one zero for f(x)=Cos[x].

In[7]:=
```
totpltrng={};gphtbl={};
proc[f=Cos[x],mzro=1,nzro=2,from=-2*Pi,to=4*Pi]
```
Out[7]=

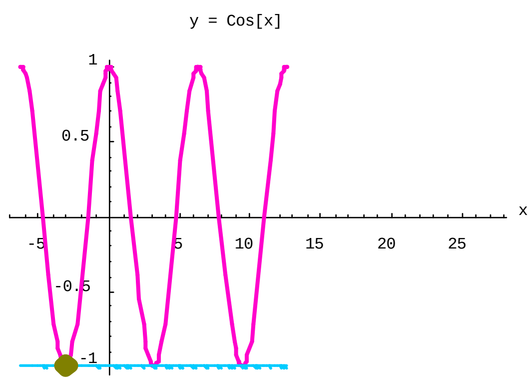
Solve::ifun:
  Warning: Inverse functions are being used by Solve,
   so some solutions may not be found.
```
mzro= 1  or  nzro= 2  exceeds the   number of zeros= 1
   The program   will terminate.
```

To draw a graph of f(x)=Cos[x] and a tangent line halfway between the 1st zero, x=-3*Pi/2 and the 2nd zero, x=-Pi/2, we can specify the zeros of f, and the critical points of f on the interval [-2*Pi,4*Pi] as follows:

In[8]:=
```
zrolist=N[{-3Pi/2,-Pi/2,Pi/2,3Pi/2,5Pi/2,7Pi/2}]
```
Out[8]=
```
{-4.71239, -1.5708, 1.5708, 4.71239, 7.85398, 10.9956}
```
In[9]:=
```
critlist=N[{-2Pi,-Pi,0,Pi,2Pi,3Pi,4Pi}]
```
Out[9]=
```
{-6.28319, -3.14159, 0, 3.14159, 6.28319, 9.42478,
  12.5664}
```

Note the use of the command N[....]. Without it , Mathematica would try to symbolically express all computions in terms of Pi . The program would take much longer to execute, if it could finish at all. Let's continue as before:

In[10]:=
```
totpltrng={};gphtbl={};
proc[f=Cos[x],mzro=1,nzro=2,from=-2*Pi,to=4*Pi]
```
In[11]:=
```
Show[gphtbl,DisplayFunction->$DisplayFunction]
```



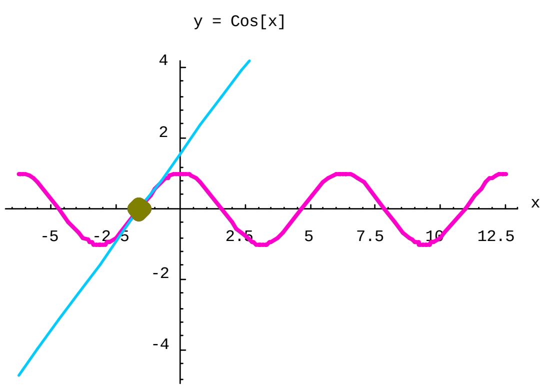
We repeat the same exercise, except we graph the tangent line between the 1st and 3rd zero of f(x)=Cos[x]. Semi-colons were used to suppress some of the printed output.

In[12]:=

```
zrolist=N[{-3Pi/2,-Pi/2,Pi/2,3Pi/2,5Pi/2,7Pi/2}];
critlist=N[{-2Pi,-Pi,0,Pi,2Pi,3Pi,4Pi}];
totpltrng={};gphtbl={};
proc[f=Cos[x],mzro=1,nzro=3,from=-2*Pi,to=4*Pi]
```

In[13]:=

```
Show[gphtbl,DisplayFunction->$DisplayFunction]
```



We provide some examples of how to animate the graph of a cubic polynomial y=p(x) as one root varies and the other two remain fixed. In the following, we graph p(x)=(x-1)(x-3)(x-t) for x in [0,4] as t varies from x=1 to x= 3 in increments of .1. At each time t, we also observe the tangent line, halfway between the 2nd and 3rd zeros, at x=(t+3)/2. We run proc in a "Do" loop, letting t vary at each step. With every run, information about plot range and graphics output is joined to the big lists, totpltrng and gphtbl.

In[14]:=

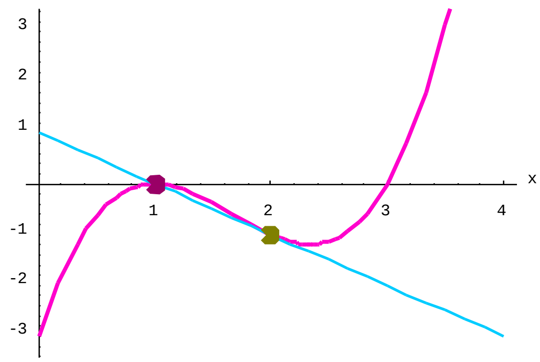
```
totpltrng={};gphtbl={};
Do[proc[f=p[t,1,3,x],mzro=2,nzro=3,from=0,to=4],
 {t,1,3,.1}]
```

The program, anmtshow, is used to draw the frames of animation in a common plot range for all of the frames. Observe that the tangent line passes through the 1st root, x=1, in every frame. We deleted every other frame in the following.

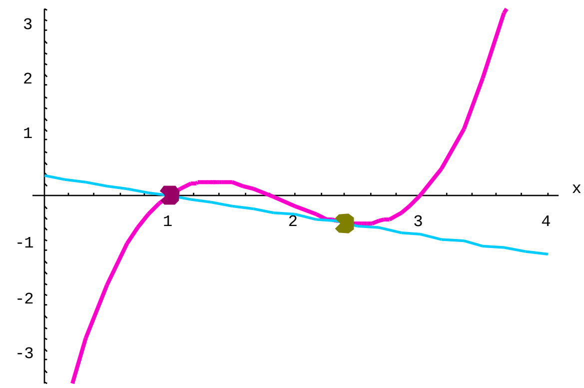
In[15]:=

```
anmtshow[shwtbl=gphtbl,rngtbl=totpltrng]
```

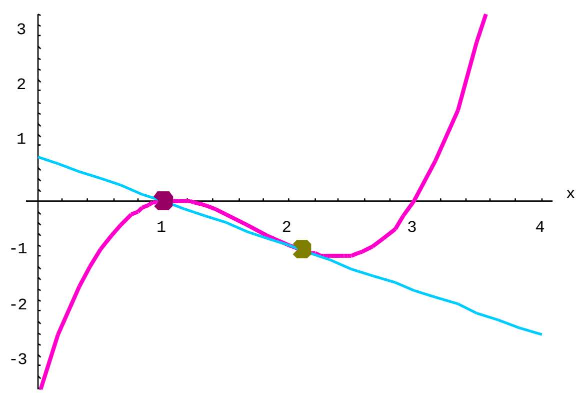
$y = (-3. + x)\,(-1. + x)^2$



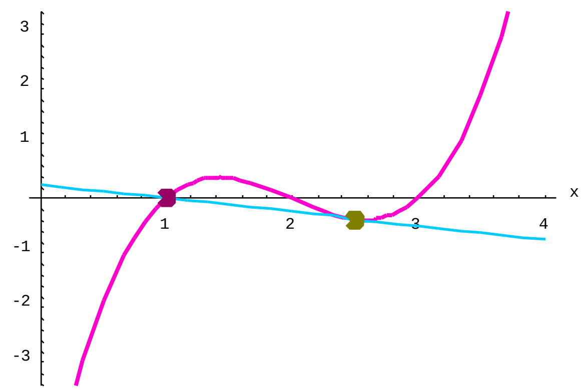
y = (-3. + x) (-1.8 + x) (-1. + x)



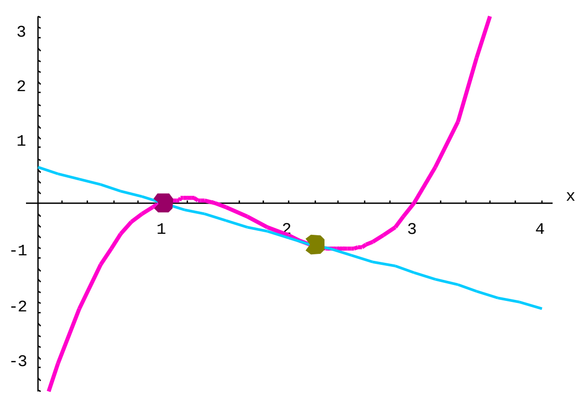
y = (-3. + x) (-1.2 + x) (-1. + x)



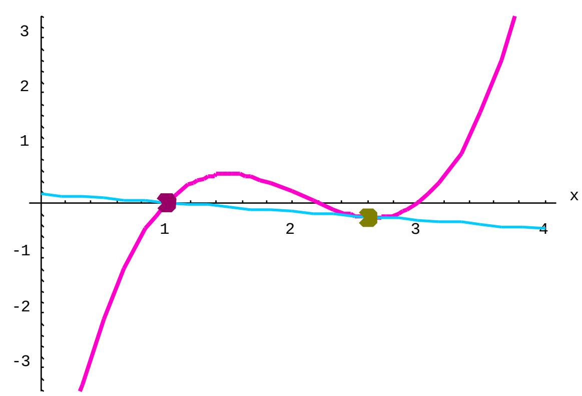
y = (-3. + x) (-2. + x) (-1. + x)



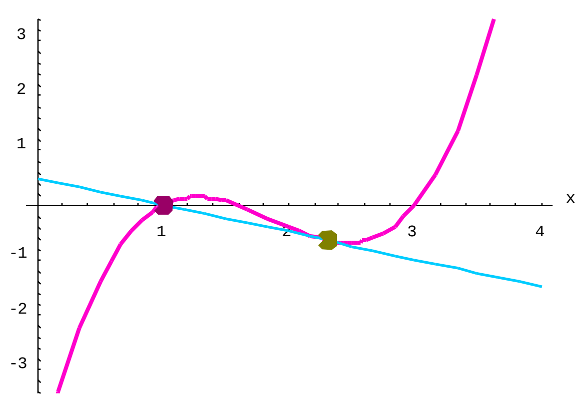
y = (-3. + x) (-1.4 + x) (-1. + x)



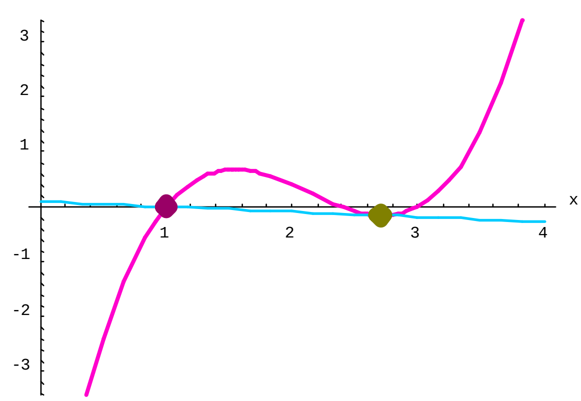
y = (-3. + x) (-2.2 + x) (-1. + x)

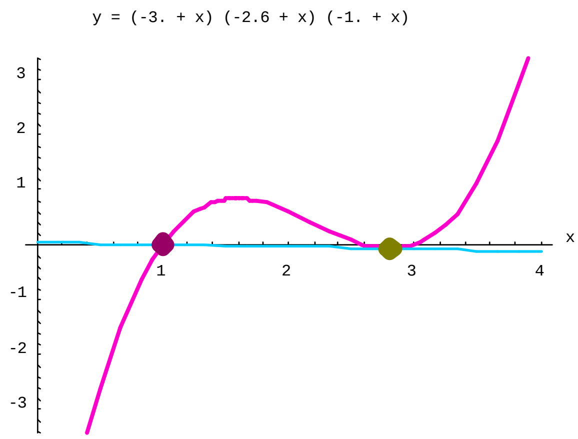


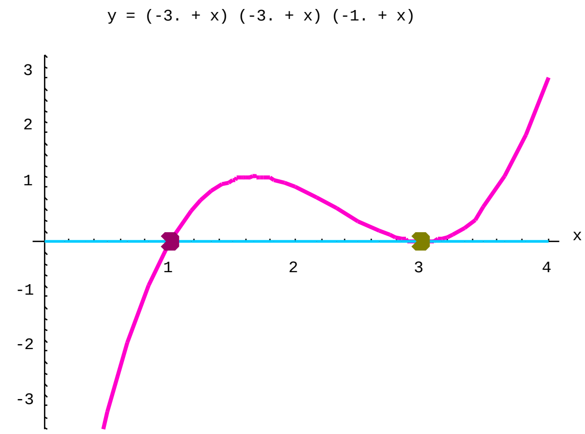
y = (-3. + x) (-1.6 + x) (-1. + x)

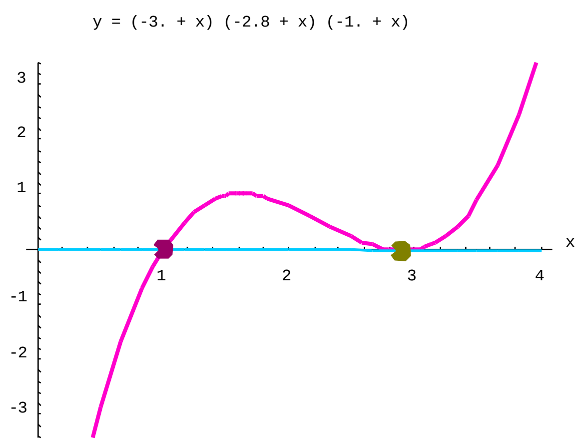


y = (-3. + x) (-2.4 + x) (-1. + x)









In the following example, we graph p(x)=(x-1)(x-3)(x-t) as t varies from x=1 to x=3, but we observe the tangent line halfway between the 1st and 2nd zeros, that is , at x=(1+t)/2 . This time, the tangent line always passes through the 3rd root, x=3, on the x-axis.

In[16]:=

```
totpltrng={};gphtbl={};
Do[proc[f=p[t,1,3,x],mzro=1,nzro=2,from=0,to=4],
{t,1,3,.1}]
```

Out[16]=

Infinity::indet:
Indeterminate expression 0. ComplexInfinity
encountered.

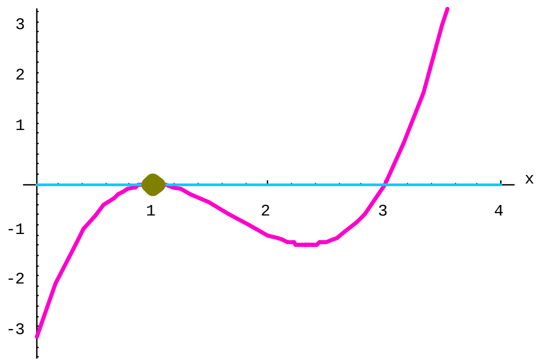
In[17]:=

```
anmtshow[shwtbl=gphtbl,rngtbl=totpltrng]
```

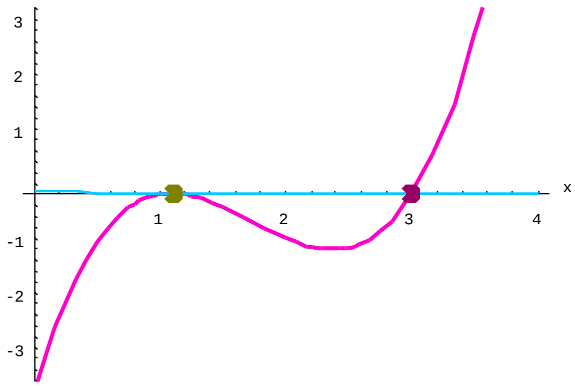
Out[17]=

The plot range along the x and y axes for all\
frames is {-0.1, 4.1} and {-3.42367, 3.42389}
,respectively

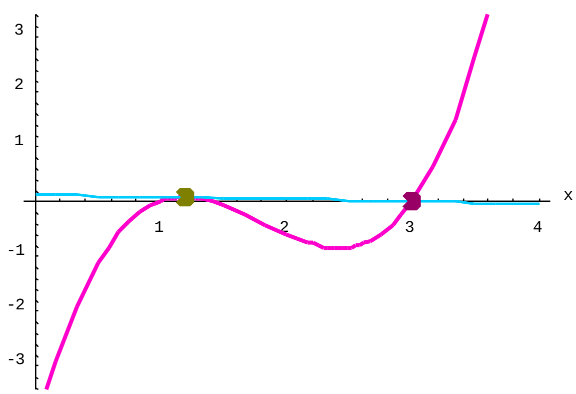
y = (-3. + x) (-1. + x)



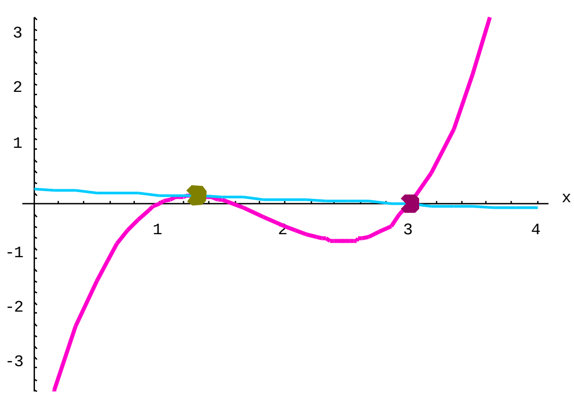
y = (-3. + x) (-1.2 + x) (-1. + x)



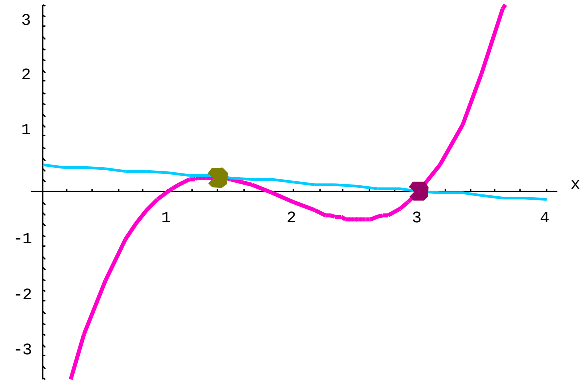
y = (-3. + x) (-1.4 + x) (-1. + x)



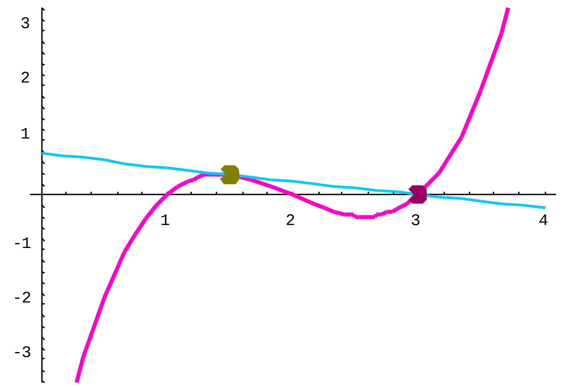
y = (-3. + x) (-1.6 + x) (-1. + x)



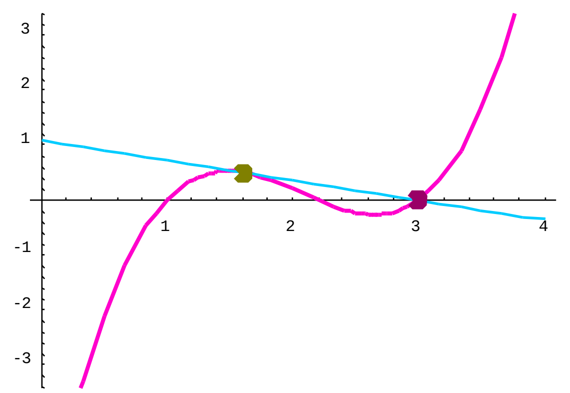
y = (-3. + x) (-1.8 + x) (-1. + x)



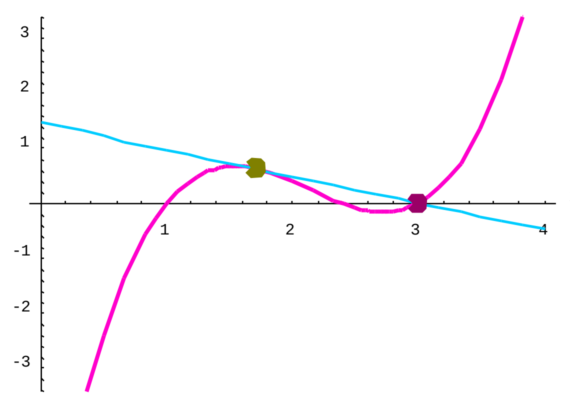
y = (-3. + x) (-2. + x) (-1. + x)

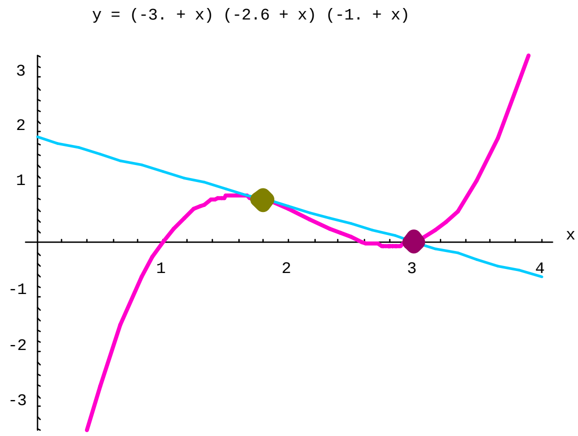


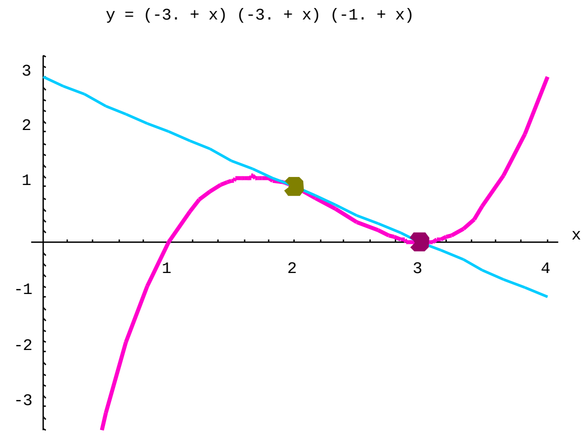
y = (-3. + x) (-2.2 + x) (-1. + x)

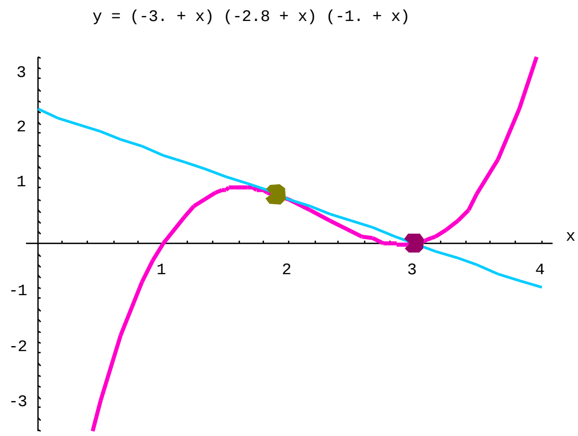


y = (-3. + x) (-2.4 + x) (-1. + x)









Finally, we look at the graph of the same polynomial with the tangent line halfway between the 1st and 3rd roots, that is, at x=2. Observe that it always passes through the x axis at the location of the 2nd root, x = t.

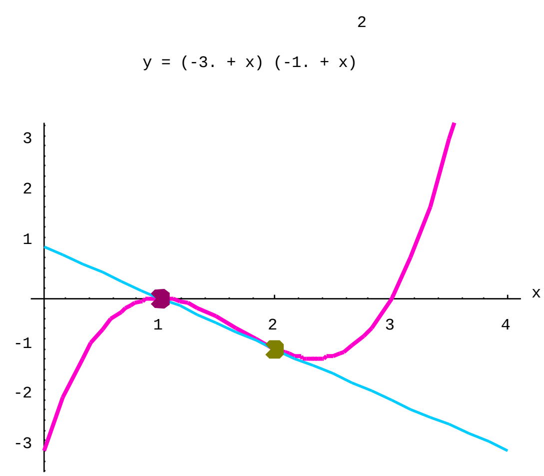
In[18]:=

```
gphtbl={};totpltrng={};
Do[proc[f=p[t,1,3,x],mzro=1,nzro=3,from=0,to=4],
{t,1,3,.1}]
```

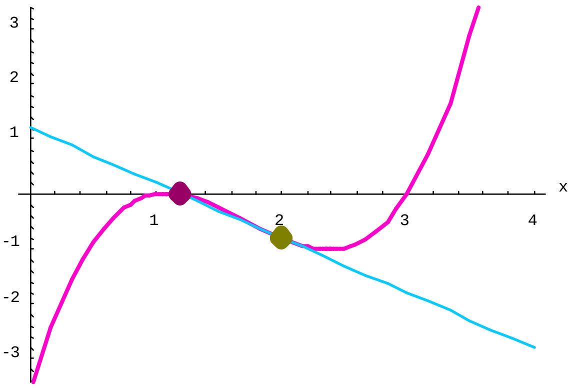
Out[18]=

```
anmtshow[shwtbl=gphtbl,rngt
bl=totpltrng]
```

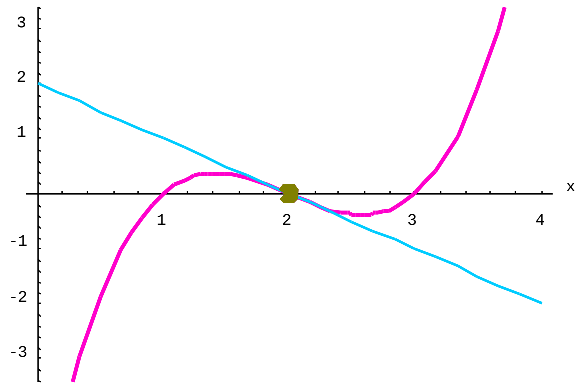
The plot range along the x and y axes for all frames is {-0.1, 4.1} and {-3.42388, 3.42391} ,respectively



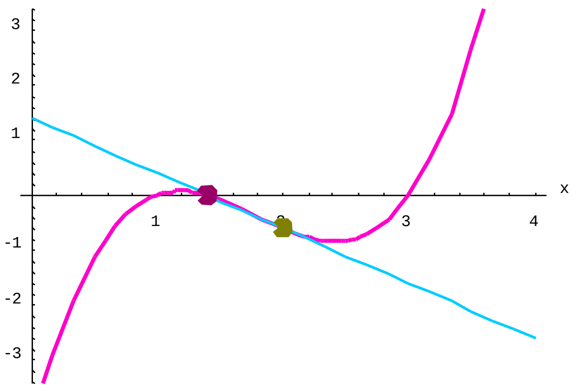
y = (-3. + x) (-1.2 + x) (-1. + x)



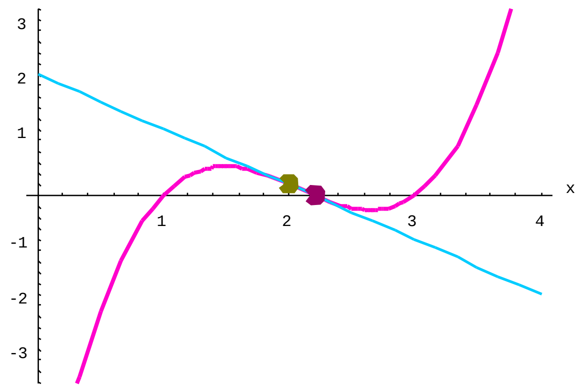
y = (-3. + x) (-2. + x) (-1. + x)



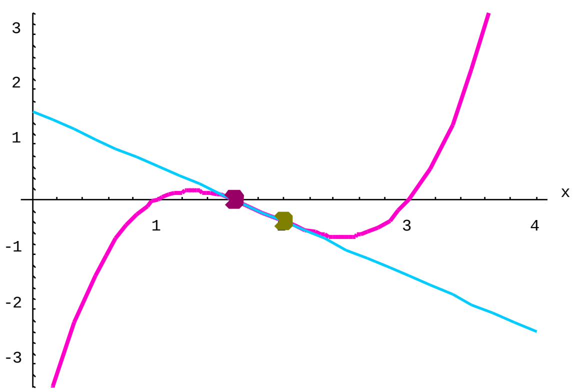
y = (-3. + x) (-1.4 + x) (-1. + x)



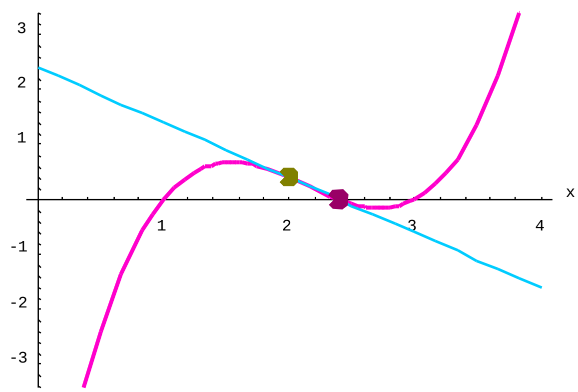
y = (-3. + x) (-2.2 + x) (-1. + x)



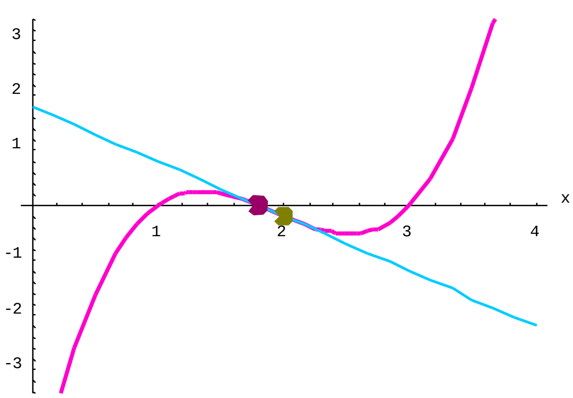
y = (-3. + x) (-1.6 + x) (-1. + x)



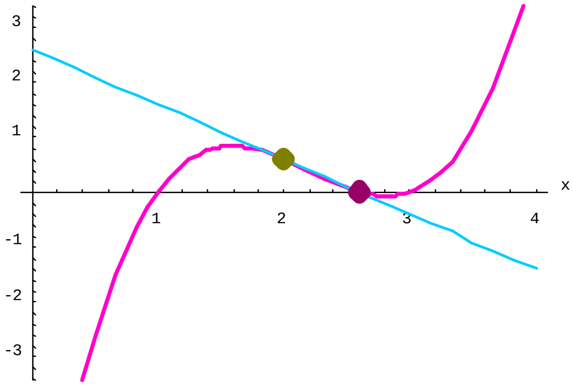
y = (-3. + x) (-2.4 + x) (-1. + x)



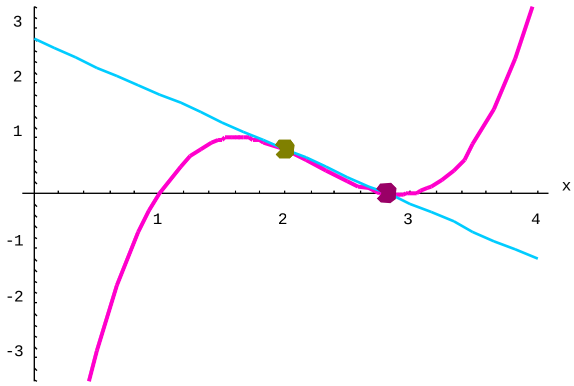
y = (-3. + x) (-1.8 + x) (-1. + x)



y = (-3. + x) (-2.6 + x) (-1. + x)



y = (-3. + x) (-2.8 + x) (-1. + x)



y = (-3. + x) (-3. + x) (-1. + x)