Contents

Fourth International Derive TI-89/92 Conference
Liverpool John Moores University, July 12 – 15, 2000

Projects Using TI-Interactive to Study Polynomials and Their Properties

Carl Leinbach
Liverpool John Moores University, UK *whilst of leave from* **Gettysburg College, USA**
Email: leinbach@cs.gettysburg.edu

Introduction

This Workshop has two purposes. The first, and most important, purpose is to show three projects that can be done by students to learn about the static and dynamic properties of polynomial curves. The second purpose is to illustrate a new product that is a sibling of the two CAS's mentioned in the title of this conference, Derive and the TI-89/92 series of calculators. Its name is TI-Interactive. It should prove to be a very useful tool for the mathematics classroom. We will not show all of its properties, but will concentrate on the CAS that is available within this package. We will also point out some ways in which TI-Interactive differs from Derive 5 and the TI-89/92 as well as ways in which it is similar. For example, TI-Interactive has all of the functions of the TI-89 and 92 plus calculators, but it does not have their programming languages. Furthermore, it can not import a program written on these calculators. It can also perform many of the operations of Derive, but it can not capture sub expressions, do partial calculations, and import utility files. On the other hand, TI-Interactive provides a comfortable interface for students and also, as we hope to illustrate, allows instructors to write complete classroom exercises for their students.

The workshop consists of three projects. The first is a recreation using Computer Algebra of the method for solving cubic equations. Finding roots is what I classify as a "static" application. The polynomial already exists and the roots are there waiting to be found. Nothing changes throughout the course of the process. Yes, there is some very interesting and clever mathematics involved. However, once the result is learned, it can be memorised or stored and used repeatedly.

The second and third projects deal with dynamic properties of cubic curves. In particular, we find curves that meet certain specifications. Properties that may change with every problem. These problems also involve finding derivatives and so, may need to be introduced at a higher level than the first project.

In the second project the participants will "customise" a race track that was a straight oval into a more exciting design for motor cross racing. They will have to design a segment of the track that satisfies certain placement and 'smoothness' conditions.

The third project, and the most ambitious of the three, deals with cubic splines to reproduce a random drawing. It involves "programming" in the sense of T|I-Interactive. This is pure functional programming and uses the decision structure of the TI-89/92 calculators and recursion. We will describe the details as we get to them.

The following pages are printouts of the TI-Interactive sessions that I have written. So let's get on with it!

Contents

Duplicating the Cardan Method for Solving a Cubic Equation
A Warm Up Exercise

Standard Form

We assume that we will be dealing with equations of the form that have a cubic expression on the right side and zero on the left.

$$\alpha x^3+\beta x^2+\delta x+\varepsilon=0 \qquad (1)$$

We begin by taking the cubic expression and converting it into the standard form, i.e.

$$u^3+mu+n \qquad (2)$$

This is done by a simple shift of the origin along the *x*-axis to the *x*-co-ordinate of the inflection point for the function.

$$f(x)=\alpha x^3+\beta x^2+\delta x+\varepsilon \qquad (3)$$

We use IT-Interactive to find this point, and prove our point.

$$f(x):=\alpha\cdot x^3+\beta\cdot x^2+\delta\cdot x+\varepsilon \qquad \text{"Done"}$$

$$\frac{d^2}{dx^2}(f(x)) \qquad 6\cdot\alpha\cdot x+2\cdot\beta$$

$$\text{solve}(\text{ans}(1)=0,x) \qquad x=\frac{-\beta}{3\cdot\alpha}$$

$$\text{expand}\left(f\left(x-\frac{\beta}{3\cdot\alpha}\right),x\right) \qquad \alpha\cdot x^3+\frac{(3\cdot\alpha\cdot\delta-\beta^2)\cdot x}{3\cdot\alpha}+\frac{27\cdot\alpha^2\cdot\varepsilon-9\cdot\alpha\cdot\beta\cdot\delta+2\cdot\beta^3}{27\cdot\alpha^2}$$

Note that whilst the coefficients are messy, we have eliminated the x^2 term. Thus, we work only with an expression of the form of (2).

We first expand the expression $(a+b)^3$

$$\text{expand}((a+b)^3) \qquad a^3+3\cdot a^2\cdot b+3\cdot a\cdot b^2+b^3 \qquad \text{factor}(3\cdot a^2\cdot b+3\cdot a\cdot b^2,a\cdot b) \qquad 3\cdot a\cdot(a+b)\cdot b$$

We have come to the big insight in the history of solving cubic equations: if we substitute (*a* + *b*) for *u* in the left side of (2), then we have

$$(a+b)^3+m\cdot(a+b)+n$$

If we substitute in the above for m and n as indicated below, we see that we have the cubic equation shown in (2).

Contents

$$\text{ans}(1)\mid m=-3\cdot a\cdot b \text{ and } n=-\left(a^{3}\right)-b^{3} \qquad 0$$

All that really remains is to find a and b in tems of m and n. This looks like we have gone in a circle, but we will see that we have really made progress.

$$\text{solve}(m=-3\cdot a\cdot b,\ b) \qquad b=\frac{-m}{3\cdot a}$$

$$n=-\left(a^{3}\right)-b^{3}\mid b=\frac{-m}{3\cdot a} \qquad n=\frac{m^{3}}{27\cdot a^{3}}-a^{3}$$

$$\text{ans}(1)\mid a=\sqrt[3]{c} \qquad n=\frac{m^{3}}{27\cdot c}-c$$

$$\text{ans}(1)\cdot(27\cdot c) \qquad 27\cdot c\cdot n=-\left(27\cdot c^{2}-m^{3}\right)$$

At this point we have a quadratic in c which we know how to solve, but we let TI-Interactive do the work and then find u. Just follow along

$$l1:=\text{zeros}\left(27\cdot c^{2}+27\cdot c\cdot n-m^{3},\ c\right) \qquad \left\{\frac{\sqrt{3\cdot\left(4\cdot m^{3}+27\cdot n^{2}\right)}-9\cdot n}{18},\ \frac{-\left(\sqrt{3\cdot\left(4\cdot m^{3}+27\cdot n^{2}\right)}+9\cdot n\right)}{18}\right\}$$

$$a:=\sqrt[3]{l1_{[1]}} \qquad \frac{\sqrt[3]{3\cdot\left(\sqrt{3\cdot\left(4\cdot m^{3}+27\cdot n^{2}\right)}-9\cdot n\right)}\cdot 2^{2/3}}{6}$$

$$b:=\frac{-m}{3\cdot a} \qquad \frac{-m\cdot 3^{2/3}\cdot\sqrt[3]{2}}{3\cdot\sqrt[3]{\sqrt{3\cdot\left(4\cdot m^{3}+27\cdot n^{2}\right)}-9\cdot n}}$$

$$u:=(a+b) \qquad \frac{\sqrt[3]{3\cdot\left(\sqrt{3\cdot\left(4\cdot m^{3}+27\cdot n^{2}\right)}-9\cdot n\right)}\cdot 2^{2/3}}{6}-\frac{m\cdot 3^{2/3}\cdot\sqrt[3]{2}}{3\cdot\sqrt[3]{\sqrt{3\cdot\left(4\cdot m^{3}+27\cdot n^{2}\right)}-9\cdot n}}$$

$$\text{comDenom}(\text{ans}(1)) \qquad \frac{\left(2\cdot\left(\sqrt{3\cdot\left(4\cdot m^{3}+27\cdot n^{2}\right)}-9\cdot n\right)\right)^{2/3}\cdot\sqrt[3]{3}-2\cdot m\cdot 3^{2/3}\cdot\sqrt[3]{2}}{6\cdot\sqrt[3]{\sqrt{3\cdot\left(4\cdot m^{3}+27\cdot n^{2}\right)}-9\cdot n}}$$

This answer is certainly messy and should not be memorised, or in my opinion, even used. There are two reasons for this. The first is that Computer Algebra Systems, such as that found in TI-Interactive, can do the process. Thus, once the students have studied and learned the valuable lessons about the mathematical discovery and moved on in their mathematical development, they can use the CAS to find roots when they need them. The second, and much more important one is that if we are teaching to have our students LEARN Mathematics instead of just do mathematics, the important material is the method or algorithm and the ability to apply it. Let's see how much you learned from the process. In the following, You will generate a random cubic polynomial and then apply the method described above.

First generate a random cubic polynomial. Don't use mine! Generate one of your own.

$\mathrm{randint}(-9,9)\cdot x^3+\mathrm{randint}(-9,9)\cdot x^2+\mathrm{randint}(-9,9)\cdot x+\mathrm{randint}(-9,9)$ $5.\cdot x^3-2.\cdot x^2+4.\cdot x+1.$

Eliminate the x^2 term by translating along the *x*-axis. Also, normalise the cubic by making the coefficient of x^3 equal to 1. You can get a Math Box by clicking on the icon at the left side of the tool bar or by pressing <ctrl>-M

To help you remember what substitutions to make expand

Solve for *a* and *b*.

Find a root for the equation with no squared term. How can you find the other roots?

Find the roots for the original cubic.

CONGRATULATIONS, you have just completed your first TI-Interactive lesson.

Contents

Looking at Polynomials as Dynamic Mathematical Objects

Introduction

In this section we will consider the problem of constructing the expression for a polynomial curve based on the dynamic properties of the curve. By that we mean a description of the behavior of the curve at certain points within the domain of the curve. We use TI-Interactive to do the algebraic manipulations and we concentrate on the hard work of solving the problem. We begin with a fanciful story to spark our students' interest.

> *You and your friend have just inherited a motor racing track. It is three miles long with one-mile long straight sections and has been used for racing automobiles. The two of you decide that it would be much more exciting (and profitable) to make a motor-cross track (a track for racing motor cycles geared for off road travel) that might include a section that includes a jump. You also believe that the oval shape of the track is "boring" and decide to change the shape of the east end of the track to cut across the existing infield and include the jump in the middle. Since you were the "math whiz" in school, you are asked to design the new track*

Let's look at the skills that we will need for this project:

To draw the existing track:

1. Drawing parametric plots for two straight sections and two semicircles on either end of the straight sections.
2. Since the plot feature of TI-Interactive, like that of the TI-89 and TI-92 calculators only accepts one range of t values for each plot screen, it will be necessary to translate our given range into the desired range for the plots. For example, for the semicircle on the west side of the track we want a range starting at $\pi/2$ and ending at $3\pi/2$. For the semicircle at the east end we will want the range to go from $-\pi/2$ to $\pi/2$. For the straight lines the range will go from -1 to 1.

To construct the new section of track

1. Determine an expression that meets certain predefined characteristics.
2. Solve a system of linear equations
3. Find the derivative (gradient of a function).
4. Convert a function expression into a parametric expression.

Contents

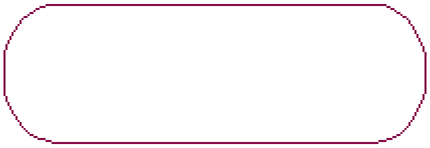
Drawing the Existing Track

We begin at the left side of the track (west end) with the semi circle. Thus our range for *t* will be $\pi/2 \le t \le 3\pi/2$. We set this by choosing from the Insert option on the main menu bar and choosing "Graph" and then "Parametric". On the graphing pallet that appears choose the "Format" button and set `tmin` to `pi/2` and `tmax` to `3pi/2.` You will also want to set the range for *x* and *y* as well as the grids, and axes. Inset a graph here and explore the options. You can remove it when you are finished with your exploration and want to leave this document in its original condition.

We begin by drawing the semicircle on the left (west end) of the track. Since the track is three miles long and the straight sections are one mile each, that means the circular sections are each 1/2 mile long. The circumference of the entire circle is 1 mile. Thus, the radius of the circle is: $1/(2\pi)$. Of course, you can do this in your head, but let's try out TI-Interactive. Click on the icon with the square root symbol pointing to a page. Notice that a calculator appears and also an equation box that looks like an equation editor box. In that box type: **solve(πr=1, r)**. This is the same command as on the TI-89 or 92. When you press the enter key, by default you see the solution of the equation below your command. Note that this was an algebraic solution. TI-Interactive will execute the same symbolic commands as the TI-89/92. You can remove this the results of the trial exercise by placing your cursor after the equation box and pressing the back space key on your computer key board. The parametric equations of a circle with centre at (c1,c2) and radius, r, are

x1(t):= c1 + r*cos(t)
y1(t):= c2 + r*sin(t)

Double click on the graph below and look at the first equation.



Can you explain all of the terms in those equations? Also explore the other options. Check out the format button and note the options that were selected.

$$\text{solve}\left(x\cdot\left(\frac{\pi}{2}\right)+y=-1 \text{ and } x\cdot\left(\frac{3\cdot\pi}{2}\right)+y=1,\ \{x, y\}\right)$$

$$x=\frac{2}{\pi} \text{ and } y=-2$$

To graph the other half of the circle at the east end we merely change the position of the centre and replace t by t - π. To find the equations for the straight sections, the y-co-ordinate is , of course, a constant, the x coordinate is found by transforming the interval [π/2, 3π/2] into [-1,1]. We can find the transformation using TI-Interactive and the **solve** command

$$\text{solve}\left(x\cdot\left(\frac{\pi}{2}\right)+y=-1 \text{ and } x\cdot\left(\frac{3\cdot\pi}{2}\right)+y=1, \{x, y\}\right)$$

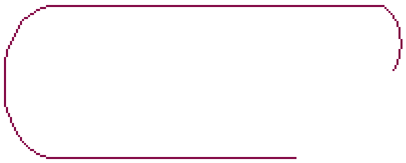
$$x=\frac{2}{\pi} \text{ and } y=-2$$

We leave it to you to verify the parametric equations for the top straight section of the track.

Changing the Design of the Track

You decide to leave the top straight section the same and to cut a quarter of a mile off of the bottom one. You also realise that you must have a way to lead the traffic onto the top section. To do this last task, you decide to reduce the size of the semicircle joining the to circle to one quarter of a mile in length. Use TI-Interactive to find the circumference of that section and the parametric equations of the section. You are going to have your other new section of track join this smaller semicircle at an angle of -π/4 with the horizontal.

Click on the graph below and compare your answers with the second and fourth set of equations.



We are now ready for the main portion of this exercise, to find the expression for the section of track that cuts across the infield.

Finding a Polynomial to Meet Certain Criteria

At this point we know two things about the section of the track that is to be constructed. It leaves to bottom straight section at the point (1/4, 0) and meets the (new) small circle at the point where the radius makes an angle of -π/4 with the horizontal. Where is that point? Use TI-Interactive and list the values of (x2(t), y2(t)) when t = π/2. We can make these values for decimals by using the **approx(** command, but for now we leave the calculations as exact.

Contents

$$\left[x2\left(\frac{\pi}{2}\right)\quad y2\left(\frac{\pi}{2}\right)\right]$$

$$\left[\frac{\sqrt{2}}{8\cdot\pi}+\frac{1}{2}\quad \frac{-\left(\sqrt{2}-6\right)}{8\cdot\pi}\right]$$

With only two pieces of information all we can do is draw a straight line joining the two points. This will not be a good solution since the line may join the circle along the tangent, but it will not make a smooth connection with the bottom. Try it with just a straight edge. The angle at the bottom will be very sharp. We want the places where the curves join to be 'smooth.' That is we want the entire track to look like one smooth closed curve. On the old track we did this without even thinking about it. We had the circles join the straight sections at the top and bottom, i.e. along the tangents to the semicircles We will do the same thing here. We will have the new stretch of track meet the old so that the tangents to the curve and the existing track are equal. That means that the first derivatives, or gradients, of the new section of track at these points must be the same as the gradients at the existing sections at the points where they join. We now have four pieces of information. That means we have enough equations to solve for four parameters.

It is a well-known fact that a polynomial of degree n - 1 has n parameters. Thus, we are looking for a cubic polynomial.

$$f(x) := ax^3 + bx^2 + cx + d$$

Its' derivative is, of course

$$f'(x) := 3ax^2 + 2bx + c\ .$$

The equations that we have to work with to find the expression for the new section of track that can be found using TI-Interactive

$$f(x) := a\cdot x^3 + b\cdot x^2 + c\cdot x + d$$

$$f\left(\frac{1}{4}\right) = 0 \rightarrow eq1 \qquad \frac{a}{64}+\frac{b}{16}+\frac{c}{4}+d=0$$

$$f\left(x2\left(\frac{\pi}{2}\right)\right) = y2\left(\frac{\pi}{2}\right) \rightarrow eq2$$

$$\frac{a\cdot\left(\sqrt{2}+4\cdot\pi\right)^3}{512\cdot\pi^3}+\frac{b\cdot\left(\sqrt{2}+4\cdot\pi\right)^2}{64\cdot\pi^2}+c\cdot\left(\frac{\sqrt{2}}{8\cdot\pi}+\frac{1}{2}\right)+d=\frac{-\left(\sqrt{2}-6\right)}{8\cdot\pi}$$

$$\left(\frac{d}{dx}\left(f(x)\right) \mid x=\frac{1}{4}\right)=0 \rightarrow eq3$$

$$\frac{3\cdot a}{16}+\frac{b}{2}+c=0$$

For the fourth equation note that the tangent line to the circle will have slope of 1 since it will be perpendicular to the radius, which has slope -1.

$$\left(\frac{d}{dx}(\mathrm{f}(x))\,|\,x=x2\left(\frac{\pi}{2}\right)\right)=1\rightarrow eq4$$

$$\frac{3\cdot a\cdot\left(\sqrt{2}+4\cdot\pi\right)^2}{64\cdot\pi^2}+\frac{b\cdot\left(\sqrt{2}+4\cdot\pi\right)}{4\cdot\pi}+c=1$$

Note how simply we were able to create the equations using the symbolic capabilities of TI-Interactive. We could take these equations and put the coefficients in a matrix and use the **rref(** command, but we will use the **solve(** command as above.

$$\mathrm{approx}(\mathrm{solve}(eq1 \text{ and } eq2 \text{ and } eq3 \text{ and } eq4, \{a, b, c, d\}))$$

$$a=-2.04172 \text{ and } b=4.10182 \text{ and } c=-1.66808 \text{ and } d=.19256$$

For convenience, since we may at some later point change the conditions for the cubic expression, we will store the particular cubic that we have found as *g*(*x*).

$$g(x):=-2.04172\cdot x^3+4.10182\cdot x^2-1.66808\cdot x+.19256$$

We have found the expression for the section of track, but to represent it on our parametric graph, we need to do a transformation of the interval [π/2, 3π/2] into the interval [1/4, x2(π/2)]. We can continue using the **solve(** command, but we will show the (alternative) matrix approach to the solution. This will show you another group of TI-Interactive commands.

The system of equations is:

$$\frac{\pi\cdot\alpha}{2}+\beta=\frac{1}{4}$$

and

$$\frac{3\cdot\pi\cdot\alpha}{2}+\beta=x2\left(\frac{\pi}{2}\right)$$

We now create a 2X3 matrix which we will call, m

$$\begin{bmatrix}\frac{\pi}{2} & 1 & \frac{1}{4}\\ \frac{3\cdot\pi}{2} & 1 & x2\left(\frac{\pi}{2}\right)\end{bmatrix}\rightarrow m$$

$$\begin{bmatrix}\frac{\pi}{2} & 1 & \frac{1}{4}\\ \frac{3\cdot\pi}{2} & 1 & \frac{\sqrt{2}}{8\cdot\pi}+\frac{1}{2}\end{bmatrix}$$

Now we row reduce m to echelon form and call the result m1.

$$\mathrm{rref}(m) \to m1$$

$$\begin{bmatrix} 1 & 0 & \dfrac{\sqrt{2}+2\cdot\pi}{8\cdot\pi^2} \\ 0 & 1 & \dfrac{-(\sqrt{2}-2\cdot\pi)}{16\cdot\pi} \end{bmatrix}$$

Thus, we have

$$\alpha := \left(m1_{[1]}\right)_{[3]} \qquad \beta := \left(m1_{[2]}\right)_{[3]}$$

$$\frac{\sqrt{2}+2\cdot\pi}{8\cdot\pi^2} \qquad \frac{-(\sqrt{2}-2\cdot\pi)}{16\cdot\pi}$$

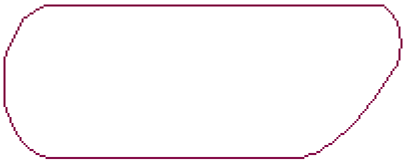
We define the transformation function, *tr*(*t*) as

$$\mathrm{tr}(x) := \alpha\cdot t + \beta$$

Now we are ready to draw the track. We copy the previous graph and place it here and define

x5(t) := tr(t)
y5(t):= g(tr(t))

And here is the result.



Exercises

1. Change the west end of the track in a manner similar to the way the east end was changed, but put the small circular section at the bottom of the track.
2. Suppose the curve that we constructed is not "exciting" enough. There is a possibility of a water or mud jump for the motor cycles if we guarantee the new section of track passes through the point (.4, .1).
    a. Verify that this point is not already on your section of track.
    b. Assuming that it is not, what degree polynomial do you now use to construct the segment of track?
    c. Construct that segment using the techniques shown above.

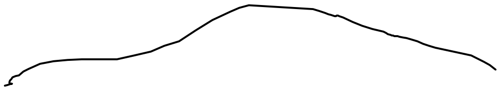
Reproducing Random Curves On a Calculator Screen

Mathematical Ideas Used: *Linear, Quadratic, and Cubic Polynomials, Interpolation, Gradients and Derivatives (alternatively first and second rates of change), Solution of a System of Linear Equations*

Introduction

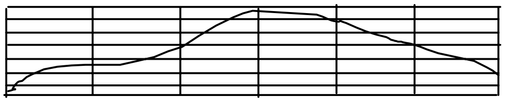
How do they achieve those nice smooth drawings of curves on a computer graphics screen? They seem to flow smoothly and do not have that bumpy effect that one gets if you just plot a lot of points and join them by straight line segments. The answer to the question is that the computer software for reproducing the drawing samples the points on the curve and uses a smooth interpolation method. Many times the interpolation method that is used is the cubic spline interpolation method. It uses mathematics that you already know in a very clever way. We will show you how it is done.

Let's begin with a very crude drawing that represents the top half of an automobile



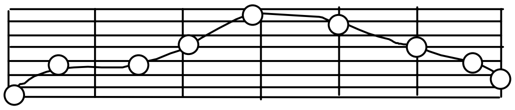
The drawing leaves much to be desired, but it will suffice as a target for the method we will develop.

The first thing that we need to do is to be able to pick out points on the curve and enter them into our calculator. Begin by imposing a co-ordinate system over top of our drawing as in the following illustration



Now that there is a co-ordinate system, some decisions need to be made. The first is where to place the origin. Chose the obvious place, the lower left-hand corner of the curve. Next what about units? We can be arbitrary here. I looked at the dimensions of the picture and decided (and I stress the arbitrary, but necessary nature of the choice) that the width should be 12 units and the height 2 units.

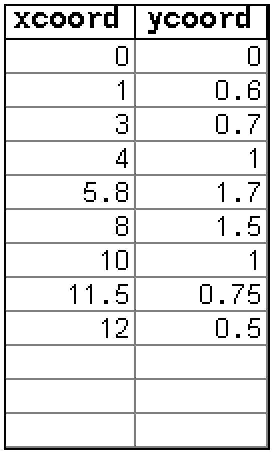
Next comes the decision of what points to pick from the curve to be stored in the calculator? This is an art and may require several retries. A general rule of thumb is not to allow gaps that are too large in between points and also to sample most frequently from areas where the curve is changing the most. For example, in the figure below the points that are most frequent are at the bumper areas and at the front wind screen. The relatively flat areas of the bonnet, roof, and boot are less frequently sampled. We choose only 9 points in order to reduce the number of equations that we will write. You will see the reason later as we develop the method.



Using the dimensions given in the earlier paragraph, we can estimate the co-ordinates of the points

x	0.0	1.0	3.0	4.0	5.8	8.0	10.0	11.5	12.0
y	0.0	0.6	0.7	1.0	1.7	1.5	1.0	0.75	0.5

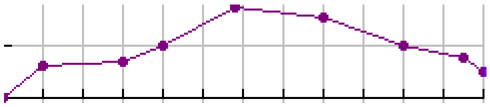
We can now begin to transfer the picture to a table in TI-Interactive



xcoord	ycoord
0	0
1	0.6
3	0.7
4	1
5.8	1.7
8	1.5
10	1
11.5	0.75
12	0.5

Linear Interpolation – A First Attempt

After using the List editor (click on the button shown on the toolbar) to enter the coordinates of the points in a data table as the lists **xcoord** and **ycoord**, we use the Plot option on the tool bar and enter "xcoord" as the independent variable and "ycoord" as the dependent variable. We choose xy-line as our plotting style. Click on the graph above and explore the options. Also click on the "Format" button to look at the style for the plot display.



Yes, the result looks something like our original drawing and is not a bad first attempt. The lines joining the points are just line segments connecting adjacent points. That is, given any two points, (x_1, y_1) and (x_2, y_2), the line segment is a portion of the straight line

$$y = y_1 + \frac{y_2 - y_1}{x_2 - x_1}(x - x_1)$$

that joins the points.

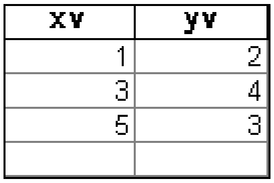
The problem with this type of approximation of the drawing is rather jagged. Look at the beginning of the bonnet and where the windscreen meets the roof as well as other places on the curve. We need to have the curves connecting the points join each other 'smoothly.' But how do we create 'smoothness?'

One measure of smoothness is that the rate of change for the curve leaving a point is the same as the rate of change for the curve entering the point. Another way of saying this is that the gradients of the two curves must be the same at the point they share in common. This is impossible if we are using linear functions since then the coefficient of x, the independent variable, would have to be the same in the expression for both lines. This would mean that all of the lines would have the same slope. We need to look at a curve having a higher degree.

A Flawed Attempt – Interpolating with Quadratics

The logical first choice for attempting a smooth interpolation technique is to try to use quadratic curves between the two points. The gradients are not constant between the two points. This means we may have a way of adjusting the curves so that they join each other smoothly at their common point. This is indeed the case, but there is a problem. We will illustrate with a smaller data set. Consider the following data for three points. Disregard the fact that with three points we can find one, unique quadratic that passes through all three. We want to talk about interpolation.

x	1.0	3.0	5.0
y	2.0	4.0	3.0



xv	yv
1	2
3	4
5	3

This choice of data means that we want to find two quadratic curves that meet smoothly at the point (3, 4). The first curve passes through the point (1, 2) and the second through (5, 3) in addition to each passing through (3, 4).

The expressions for the two quadratic polynomials are given by

$$f(x) = ax^2 + bx + c$$

$$g(x) = px^2 + qx + r$$

We already know that $f(1) = 2$, $f(3) = 4$, $g(3) = 4$, and $g(5) = 3$. Thus, we have the following equations:

$$\begin{array}{rcl} a + b + c & & = 2 \\ 9a + 3b + c & & = 4 \\ & 9p + 3q + r & = 4 \\ & 25p + 5q + r & = 3 \end{array}$$

This is a set of four equations with six unknown variables. We can not get a unique solution for this system. However we have not used the smoothness criterion. In this case it means that $f'(3) = g'(3)$. This translates to the equation

$$6a + b = 6p + q$$

or

$$6a + b - 6p - q = 0$$

We now have a system of five equations with six unknown variables. This still leaves us in a situation where we can not uniquely determine the coefficients of the polynomials. One more equation is needed if we are going to have any hope of determining the coefficients.

One possibility may be to force the second derivatives at the point where the curves join to be the same. This would force the coefficients of x^2 to be the same for each polynomial. This would have a cascading effect forcing all of the coefficients of x to be the same for each of the interpolating polynomials. This would be unsatisfactory as well as impossible in many cases.

Another idea may be to prescribe the value of the derivatives of the first and last polynomials at the end points of the curve we are trying to approximate. That would be consistent with our choice of having the derivatives equal at the points where the curves join. In our case suppose we, arbitrarily, say that $f(1) = 1$ and that $g(5) = 3/2$. Thus, we have the equations

$$2a + b = 1$$

and

$$10p + q = 3/2$$

But, now we have one equation too many! We may say that we only take one of the two. Will it matter? We will show that it does. First we take the equation given by the derivative at the right end point. We finally have the system of 6 equations in six unknown variables.

$$\begin{aligned} a + b + c &= 2 \\ 9a + 3b + c &= 4 \\ 6a + b - 6p - q &= 0 \\ 9p + 3q + r &= 4 \\ 25p + 5q + r &= 3 \\ 10p + q &= 3/2 \end{aligned}$$

We enter the coefficients from the above system as an augmented matrix in TI-Interactive, as shown on the next page. We store this matrix with the name, m

$$\mathrm{rref}(m) \rightarrow m1$$

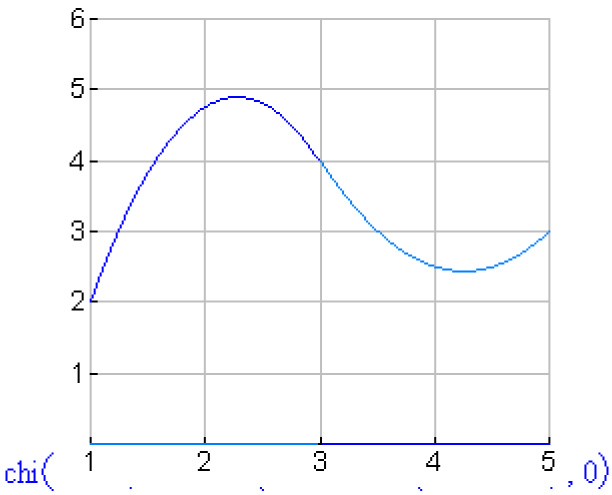
$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & \frac{-7}{4} \\ 0 & 1 & 0 & 0 & 0 & 0 & 8 \\ 0 & 0 & 1 & 0 & 0 & 0 & \frac{-17}{4} \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & \frac{-17}{2} \\ 0 & 0 & 0 & 0 & 0 & 1 & \frac{41}{2} \end{bmatrix}$$

$$\begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 2 \\ 9 & 3 & 1 & 0 & 0 & 0 & 4 \\ 6 & 1 & 0 & -6 & -1 & 0 & 0 \\ 0 & 0 & 0 & 9 & 3 & 1 & 4 \\ 0 & 0 & 0 & 25 & 5 & 1 & 3 \\ 0 & 0 & 0 & 10 & 1 & 0 & \frac{3}{2} \end{bmatrix} \rightarrow m$$

We have the TI-89/92 matrix handling capabilities at our disposal. In particular, we can use the **rref** command to find the values for *a*, *b*, *c*, *d*, *p*, *q*, and *r*. These are given down the last column of the matrix.

Before proceeding further we need to define an old Derive friend, the characteristic function, chi

Now let's look at the graph of the first "quadratic spline."



We urge you to go back and use the derivative at the left hand side to draw that "quadratic spline." Note that it is a quite different curve. In each case the two curves both join smoothly at the center point, but they are entirely different.

Which curve is the one we want? The answer is, unfortunately, ”Neither.“ The reason is that neither of the curves satisfies the derivative condition given for both end points. We can claim that the derivatives at the endpoints were only added to the set of assumptions to give us enough conditions to get a set of equations that we could solve. However, if you drew both of the "splines," they show that it makes a difference which endpoint we consider when finding the coefficients for the interpolating polynomials. We need to go on: quadratic interpolation is not a practical option.

Extending the Idea – Cubic Interpolating Polynomials

In this case we consider two cubic polynomials

$$f(x) = ax^3 + bx^2 + cx + d$$

$$g(x) = px^3 + qx^2 + rx + s$$

A first thought is that this will be worse. We have one more coefficient in each expression to take into account. This means that we will have to have eight equations! The values at the designated points and the fact that the gradients at the common points are equal once again results in five equations. But we need eight equations. We use one of the ideas given (and rejected) in the previous section. We set the second derivative of the functions equal at the points they have in common. In the case of our example that yields six equations in eight unknown variables. That is actually a better situation than five equations in six unknown variables. We have the two endpoints to consider.

There are two choices for things to do at this point. We can use the values of the first derivatives at each of the end points. This gives us eight equations and we do not need to show a preference for either end point. The cubic interpolating polynomials done in this way are called a *clamped cubic spline* because we are assuming that the curve is started and ended going in prescribed directions, i.e. that the curve is being held or clamped at the end points of the interval.

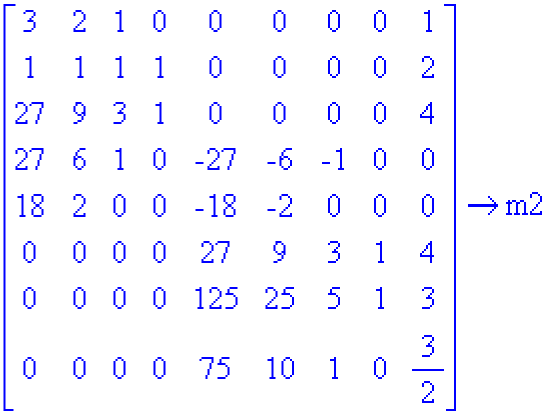
The other, more common, assumption involves the second derivative. This assumption is that the second derivative of the curve is zero at each end point. This also gives us two additional equations and we do not have to make any assumptions about the direction of the curve at the end points. The cubic interpolating polynomials that are generated under this assumption are called a *free cubic spline*. It is close to the curve that you would draw if you merely laid a flexible piece of metal over all of the points and drew the curve.

We will assume that we are given the derivatives at the end points as in the previous section and produce the interpolated curve. We leave it to the reader to find the functions *f* and *g* for the free cubic spline and draw the resulting curve. The eight equations that we derive from the information about *f* and *g* given in the previous section is:

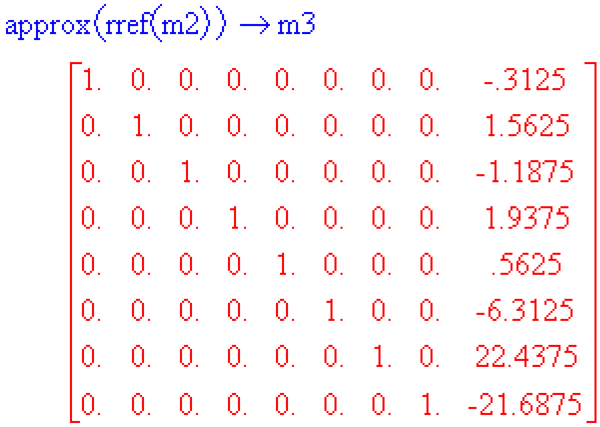
$$\begin{aligned} 3a + 2b + c &= 1 \\ a + b + c + d &= 2 \end{aligned}$$

$$\begin{array}{lllll}
27a + 9b + 3c + d & & & = 4 & \\
27a + 6b + c & -27p - 6q - r & & = 0 & \text{(gradients equal)} \\
18a + 2b & -18p - 2q & & = 0 & \text{(2nd derivatives equal)} \\
 & 27p + 9q + 3r + s & & = 4 & \\
 & 125p + 25q + 5r + s & & = 3 & \\
 & 75p + 10q + r & & = 3/2 &
\end{array}$$

We enter the system coefficients in matrix form and use the **rref** operator to reduce the system and read of the coefficients as shown below.



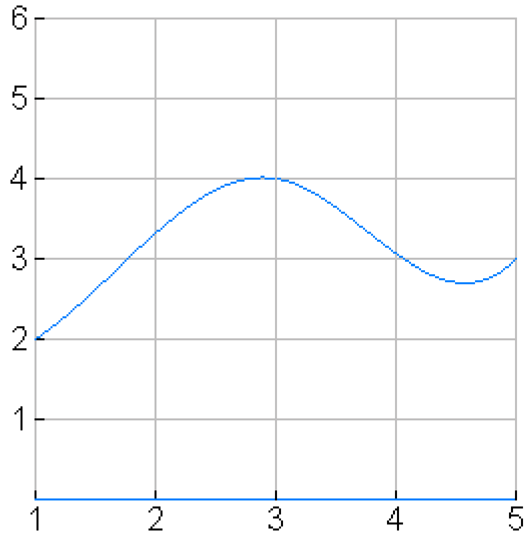
$$\begin{bmatrix} 3 & 2 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 2 \\ 27 & 9 & 3 & 1 & 0 & 0 & 0 & 0 & 4 \\ 27 & 6 & 1 & 0 & -27 & -6 & -1 & 0 & 0 \\ 18 & 2 & 0 & 0 & -18 & -2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 27 & 9 & 3 & 1 & 4 \\ 0 & 0 & 0 & 0 & 125 & 25 & 5 & 1 & 3 \\ 0 & 0 & 0 & 0 & 75 & 10 & 1 & 0 & \frac{3}{2} \end{bmatrix} \rightarrow m2$$



$$\text{approx}(\text{rref}(m2)) \rightarrow m3$$

$$\begin{bmatrix} 1. & 0. & 0. & 0. & 0. & 0. & 0. & 0. & -.3125 \\ 0. & 1. & 0. & 0. & 0. & 0. & 0. & 0. & 1.5625 \\ 0. & 0. & 1. & 0. & 0. & 0. & 0. & 0. & -1.1875 \\ 0. & 0. & 0. & 1. & 0. & 0. & 0. & 0. & 1.9375 \\ 0. & 0. & 0. & 0. & 1. & 0. & 0. & 0. & .5625 \\ 0. & 0. & 0. & 0. & 0. & 1. & 0. & 0. & -6.3125 \\ 0. & 0. & 0. & 0. & 0. & 0. & 1. & 0. & 22.4375 \\ 0. & 0. & 0. & 0. & 0. & 0. & 0. & 1. & -21.6875 \end{bmatrix}$$

These coefficients are then entered into the expressions for the cubic polynomials. We showed the approximate result because the exact were fractions and became too large for the display box. The graph can be drawn using this information.



It will be interesting to compare this graph with the graph of the free spline that you draw.

Returning to the Original Problem

Now let's get back to the top of the automobile. As you can notice from the previous section, if we have *n* points we need to find 4(*n* -1) coefficients. This means that we need to solve a system of 4(*n*-1) equations. Thus, for the top of the automobile we will have a system of 32 linear equations to solve. Entering the data can be tedious. We will write a TI-Interactive program to do the process for us. Your first reaction may be that TI-Interactive does not have a programming language. That is not really true! It allows for function definitions with the ':=' operator or '->' (STO) operator. It has a decision structure, the 'when(' operator. It also performs loops via a use of recursion. The actual programming is done in a .tii file called **`matbld.tii`** that is also stored in the accessible area of the hard drive. We will describe how that program, which is actually a set of TI-Interactive commands works.

If you do not plan ahead, functional programming can be very messy. On the other hand, with careful planning, it can be very logical and straightforward. Let's think about what we need to do to create the matrix for the parameters of the cubic polynomials.

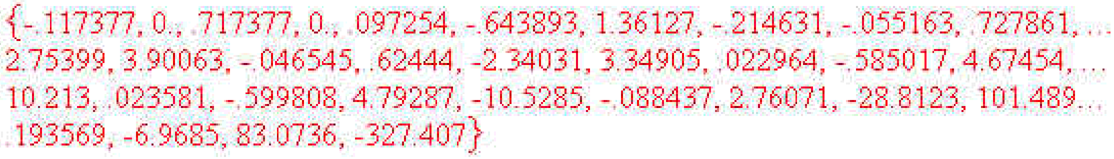
1. Create one equation for each of the left hand endpoints of the polynomials (4*n* - 1 equations).
2. Create one equation for each of the right hand endpoints of the polynomials (4*n* – 1 equations).
3. Create one equation equating the first derivatives at each interior point (4*n* – 2 equations).
4. Create one equation equating the second derivatives at each interior point (4*n* –2 equations).
5. Create the two equations for either the <u>free</u> or clamped spline case.

For each row of the matrix we need

1. The leading zeros, if any

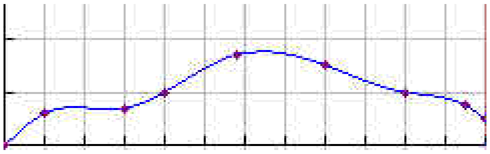
2. The parameters associated with the condition.
3. The trailing zeros, if any.
4. The right hand side of the equation.

All of the functions to do this are in **`matbld.tii`**. Once we have the matrix, we can use the **rref(** operator to find the coefficients of the polynomial. Copying the result from the **`matbld`** file, we see the list of coefficients.



```
{-.117377, 0., .717377, 0., .097254, -.643893, 1.36127, -.214631, -.055163, .727861, ...
2.75399, 3.90063, -.046545, .62444, -2.34031, 3.34905, .022964, -.585017, 4.67454, ...
10.213, .023581, -.599808, 4.79287, -10.5285, -.088437, 2.76071, -28.8123, 101.489...
.193569, -6.9685, 83.0736, -327.407}
```

The resulting graph of the splines is shown below



A Comment on Polynomials of Higher Degree

The thought may have arisen that if cubic polynomials can be used, perhaps it might be a good to try polynomials of fourth or fifth degree or maybe even higher degrees. This is not a good idea. The first reason is that we would be increasing the number of equations that need to be solved. For fourth degree polynomials we would have forty equations to solve to find the car top. For fifth degree there would be forty-eight equations. This is not an unreasonable number for a larger computer, but the increase in time on the calculator would be unreasonable.

More importantly, this is not a good idea because as we increase the degree of the polynomial, the curve may have a tendency to oscillate between the sampled points and give a wrong impression of the shape. We want to keep the degree of the curve as small as possible and still meet the smoothness criterion. The cubic curves are the first feasible candidates for the interpolating curves to meet this criterion.

We also not that any even degree polynomials are unsuitable for interpolating curves because we will have to express a preference for one of the endpoints as in the case of the quadratic curves. Our next jump up would have to be to fifth degree polynomials. Then the danger of oscillation between two points is increased.

Conclusion

We have shown that using some knowledge of cubic curves and their derivatives, we can use the calculator to gain some insight into a powerful graphics technique. Our sampling may have been limited and the calculation time longer than we would like, but the fact remains that a powerful technique has been revealed.

Contents

More efficient algorithms for determining cubic splines exist, but they require transformations of the original equations. This makes the technique more computationally efficient. Unfortunately, it makes the method harder to understand for school students. These techniques are more properly presented in advanced course. This exposition is for understanding and student enjoyment. Efficiency is an important issue to be dealt with at a later time.

APPENDIX

The File Matbld

Creating the Cubic Spline for the Car Top

First we enter the two lists of co-ordinates.

$$\mathrm{xcoord} := \{0, 1, 3, 4, 5.8, 8, 10, 11.5, 12\} \qquad \mathrm{ycoord} := \{0, .6, .7, 1, 1.7, 1.5, 1, .75, .5\}$$

First step: The values of the function on the left side of each curve. Total of 8 conditions for this data. First we take care of the leading zeros.

$$\mathrm{front}(n) := \mathrm{when}(n > 1, \mathrm{seq}(0, i, 1, 4\cdot(n - 1)), \{\})$$

Second concern is the coefficients for the parameters.

$$\mathrm{middle}(n, \mathrm{xcoord}) := \mathrm{seq}\left(\left(\mathrm{xcoord}_{[n]}\right)^{k}, k, 3, 0, -1\right)$$

Finally, the trailing zeros and the value of the function. Please note that in all that follows, we could have made these functions by adding a variable, *m*, for the number of data points,and where we have 32 put 4(*m* - 1) and for 8 put (*m* - 1).

$$\mathrm{back}(n, \mathrm{ycoord}) := \mathrm{when}\left(n < 8, \mathrm{augment}\left(\mathrm{seq}(0, i, 4\cdot n + 1, 32), \left\{\mathrm{ycoord}_{[n]}\right\}\right), \left\{\mathrm{ycoord}_{[8]}\right\}\right)$$

We define a row of the matrix for each of these conditions.

$$\mathrm{rowdef}(n, \mathrm{xcoord}, \mathrm{ycoord}) := \mathrm{augment}(\mathrm{augment}(\mathrm{front}(n), \mathrm{middle}(n, \mathrm{xcoord})), \mathrm{back}(n, \mathrm{ycoord}))$$

At this point we create a list of the first 8 rows of the matrix.

$$\mathrm{fval}(n, 11, 12) := \mathrm{when}(n > 1, \mathrm{augment}(\mathrm{fval}(n - 1, 11, 12), \mathrm{rowdef}(n, 11, 12)), \mathrm{rowdef}(1, 11, 12))$$

The second step: We create the data for the rows of the matrix pertaining to the function values at the right hand side of each of the curves. We will just describe the various equations. The following is the coefficient for the parameters. NOTE: We already have the leading zeros function created.

$$\mathrm{fval}(n, 11, 12) := \mathrm{when}(n > 1, \mathrm{augment}(\mathrm{fval}(n - 1, 11, 12), \mathrm{rowdef}(n, 11, 12)), \mathrm{rowdef}(1, 11, 12))$$

The trailing zeros and the function value.

$$\mathrm{back2}(n, \mathrm{ycoord}) := \mathrm{when}\left(n < 8, \mathrm{augment}\left(\mathrm{seq}(0, i, 4\cdot n + 1, 32), \left\{\mathrm{ycoord}_{[n+1]}\right\}\right), \left\{\mathrm{ycoord}_{[9]}\right\}\right)$$

Define the row.

$$\begin{aligned}&\mathrm{rowdef2}(n, \mathrm{xcoord}, \mathrm{ycoord}) := \ldots\\ &(\mathrm{augment}(\mathrm{front}(n), \mathrm{middle2}(n, \mathrm{xcoord})), \mathrm{back2}(n, \mathrm{ycoord}))\end{aligned}$$

The data for the second section of the matrix

$$\mathrm{fval2}(n, 11, 12) := \mathrm{when}(n > 1, \mathrm{augment}(\mathrm{fval2}(n - 1, 11, 12), \mathrm{rowdef2}(n, 11, 12)), \mathrm{rowdef2}(1, 11, 12))$$

The next step is to set the derivatives equal at each of the interior points. A total of seven equations.

$$\text{middle3}(n, \text{xcoord}) := \left\{3\cdot\left(\text{xcoord}_{[n]}\right)^2, 2\cdot\text{xcoord}_{[n]}, 1, 0, -3\cdot\left(\text{xcoord}_{[n]}\right)^2, -2\cdot\text{xcoord}_{[n]}, -1, 0\right\}$$

$$\text{back3}(n) := \text{seq}(0, i, 4\cdot n + 1, 33)$$

$$\text{derrow}(n, \text{xcoord}) := \text{augment}(\text{augment}(\text{front}(n - 1), \text{middle3}(n, \text{xcoord})), \text{back3}(n))$$

$$\text{fderval}(n, \text{xcoord}) := \text{when}(n > 1, \text{augment}(\text{fderval}(n - 1, \text{xcoord}), \text{derrow}(n, \text{xcoord})), \{\})$$

We move on to the second derivative conditions.

$$\text{middle4}(n, \text{xcoord}) := \left\{6\cdot\text{xcoord}_{[n]}, 2, 0, 0, -6\cdot\text{xcoord}_{[n]}, -2, 0, 0\right\}$$

$$\text{secrow}(n, \text{xcoord}) := \text{augment}(\text{augment}(\text{front}(n - 1), \text{middle4}(n, \text{xcoord})), \text{back3}(n))$$

$$\text{fsecval}(n, \text{xcoord}) := \text{when}(n > 1, \text{augment}(\text{fsecval}(n - 1, \text{xcoord}), \text{secrow}(n, \text{xcoord})), \{\})$$

Finally, we have the two free spline conditions

$$\text{lfree}(\text{xcoord}) := \text{augment}\left(\left\{6\cdot\text{xcoord}_{[1]}, 2\right\}, \text{seq}(0, i, 3, 33)\right)$$

$$\text{rfree}(\text{xcoord}) := \text{augment}\left(\text{seq}(0, i, 1, 28), \left\{6\cdot\text{xcoord}_{[9]}, 2, 0, 0, 0\right\}\right)$$

We split the total data set into two pieces. This step is unnecessary, but it saves having to write a more messy function for the final creation of the matrix.

$$\text{listToMat}(\text{augment}(\text{half1}(8, \text{xcoord}, \text{ycoord}), \text{half2}(8, \text{xcoord})), 33) \rightarrow \text{cumat}$$

We have finished all the steps to create the matrix. Once again, note that if we had written the functions in their slightly more general form the 8 would be (m - 1) and the 33 would be (4m - 3).

To find the coefficients we place the matrix in row reduced echelon form.

$$\text{rref}(\text{cumat}) \rightarrow \text{comat}$$

Here are the coefficients!

$$\text{seq}\left(\left(\text{comat}_{[i]}\right)_{[33]}, i, 1, 32\right) \rightarrow \text{coef}$$

$$\begin{gathered}\{-.117377, 0., .717377, 0., .097254, -.643893, 1.36127, -.214631, -.055163, .727861, \ldots\\ 2.75399, 3.90063, -.046545, .62444, -2.34031, 3.34905, .022964, -.585017, 4.67454, \ldots\\ 10.213, .023581, -.599808, 4.79287, -10.5285, -.088437, 2.76071, -28.8123, 101.489\ldots\\ .193569, -6.9685, 83.0736, -327.407\}\end{gathered}$$

We are not done with functional programming. We still need to create the function to draw the result. Basically, what we are creating here is a For - loop. First the action inside the loop, which is a when (decision) statement.

$$\text{splpiece}(x, i, \text{xcoord}) := \ldots$$

$$\left(x \geq \text{xcoord}_{[i]} \text{ and } x < \text{xcoord}_{[i+1]}, \text{coef}_{[4\cdot i-3]}\cdot x^3 + \text{coef}_{[4\cdot i-2]}\cdot x^2 + \text{coef}_{[4\cdot i-1]}\cdot x + \text{coef}_{[4\cdot i]}, 0\right)$$

Now, the function itself. The variable n will be given the value of 8 in the graph tool when we define y1(x):=spl(x,8,xcoord).

$$\text{spl}(x, n, \text{xcoord}) := \text{when}(n > 0, \text{splpiece}(x, n, \text{xcoord}) + \text{spl}(x, n - 1, \text{xcoord}), 0)$$