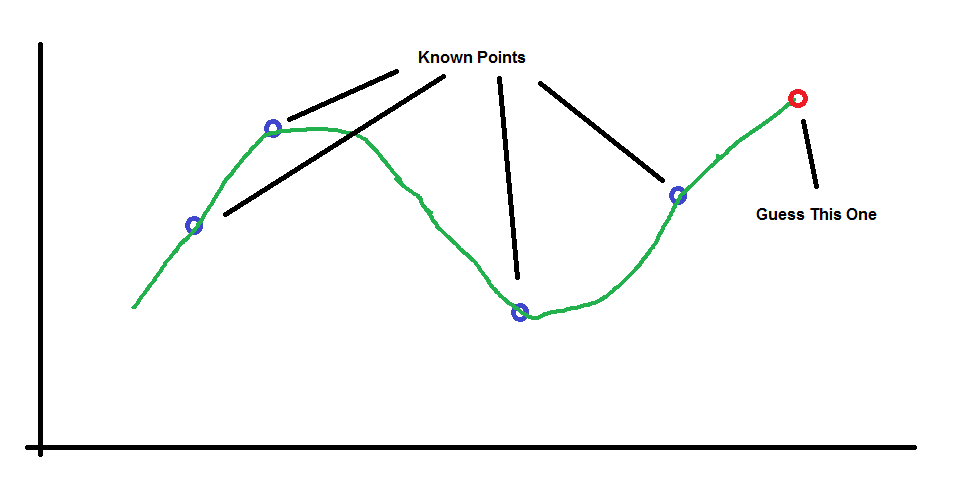
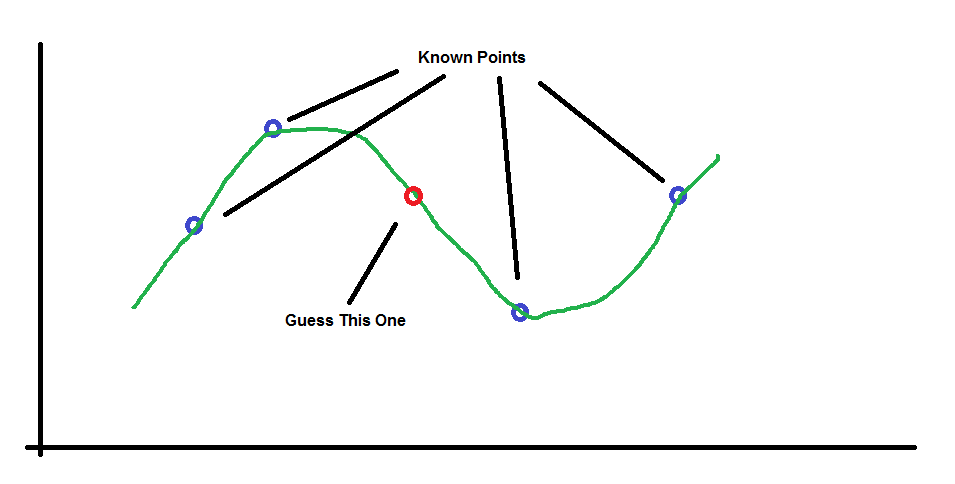
**Phi Data Cubic Extrapolation**

A common method for extrapolating (or interpolating) data points is *cubic interpolation/extrapolation.* Interpolating is when you want to guess a point between some other data points. Extrapolating is when you want to guess a point outside other data points.

Interpolation: Extrapolation:



The problem is that we don’t know the green curves beforehand so it’s actually *impossible.* However, if we assume a few things that are not too unreasonable, then the problem becomes solvable. We will assume 2 things:

1. The curve is smooth looking – not some crazy hairball type of thing
2. It has at most 2 “bends” – places where it changes direction. In other words, at most 2 reversals from “increasing” to “decreasing” (or vice versa).

A family of curves that meets our requirements is described by what is called a *cubic equation.* The idea is that we will provide some existing points and *assume* that they are on a curve that can be described with a cubic equation and ask: “if we know all these points are on a cubic curve, then what is the equation of the curve?”

The general form for any cubic equation is:



Our goal is to determine what the values of *a, b, c* and *d* using some example points that we claim are on the cubic equation we want to know. Since we want to solve for 4 values, there’s a mathematical rule that says we need 4 “examples” in order to figure it out. (It will become clear why this is so below.)

So the example points will be:



Thus there are four equations that must be true:



We want to solve for a, b, c and d in general but there is a trick that we can use to make things simpler. The reason we want to solve the cubic equation is because we want to guess a point at some other x value such as x5, but if we shift the points left or right to make sure that the point we want to guess is always at 0, then all we need to solve for is d; the equation for the point we want would then be :



Here’s how to solve it. First solve for *a* in terms of the others using the first equation:





Matlab answer:

2

b x1 + c x1 + d - y1

- ---------------------

3

x1

Now, use this value for *a* in the second equation to make *a* go away. Then solve for *b.*





Matlab answer:

The value of b is:

3

x2 (d - y1 + c x1)

d - y2 + c x2 - -------------------

3

x1

-----------------------------------

3

x2 2

--- - x2

x1

Now solve for *c* using the 3rd equation:





















Got tired of simplifying … stopped doing it here.

Matlab answer:

The value of c is:

2 2

x2 x3 (y1 - y3) x2 x3 (y1 - y2) x2 x3 (d - y1) (x2 - x3)

2 2 2 2 ---------------- - ---------------- + ------------------------

d x2 - d x3 - x2 y3 + x3 y2 x1 - x3 x1 - x2 x1

------------------------------- - --------------------------------------------------------------

2 2 x2 x3 (x2 - x3)

x2 x3 - x2 x3

And FINALLY!!! :



Lucas and I didn’t want to spend the next six hours doing this so we just gave it to Matlab.

Matlab answer:

The value of d is:

2 2 2 2 2 2

- y4 x2 x3 + y3 x2 x4 + y4 x2 x3 - y3 x2 x4 - y2 x3 x4 + y2 x3 x4

----------------------------------------------------------------------- -

2 2 2 2 2 2

- x2 x3 + x2 x4 + x2 x3 - x2 x4 - x3 x4 + x3 x4

x2 x3 x4 (x3 - x4) (y1 - y2) x2 x3 x4 (x2 - x4) (y1 - y3) x2 x3 x4 (x2 - x3) (y1 - y4)

---------------------------- - ---------------------------- + ----------------------------

x1 - x2 x1 - x3 x1 - x4

------------------------------------------------------------------------------------------

(x2 - x3) (x2 - x4) (x3 - x4)