INTERPOLATION

Ari Maller

INTERPOLATION

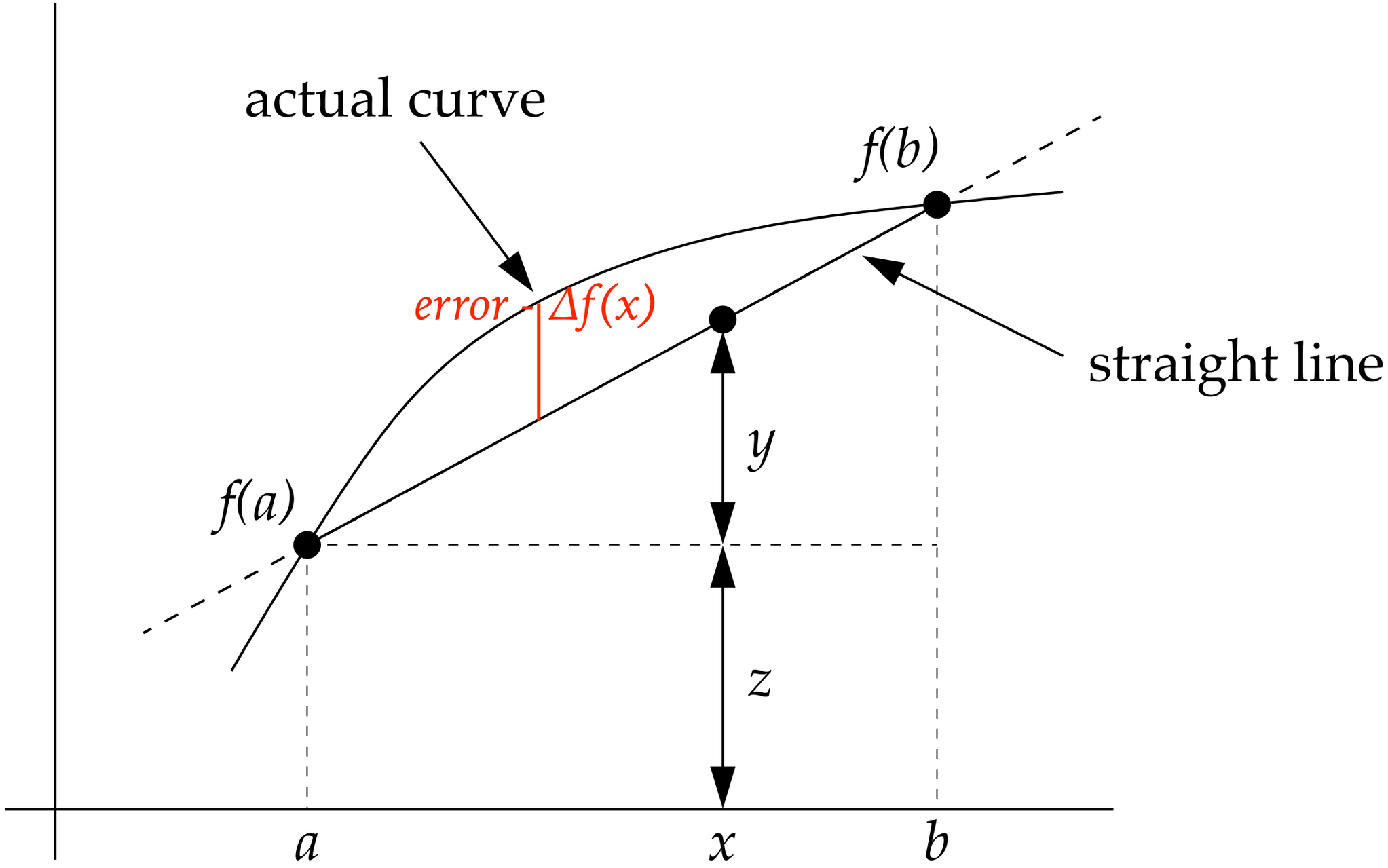
- Interpolation is the approximation of the value of a function for points in between the value that we do know.
- Interpolation and fitting a function sound very similar; however, they are trying to achieve different things.
  - *Interpolation* seeks to fill in the information in some small region of a dataset.
  - *Fitting a function* attempts to find a model that fits our data to give us some better understanding of the nature of the data.

LINEAR INTERPOLATION

- The simplest form of interpolation is linear. Take the two points closest to where you want to interpolate, fit a line to them and then from the equation for that line you have the value of your previously unknown point.
- Note that your point doesn't need to lie between the two points, though the farther away it is the worst your interpolation is likely to be.
- It can be shown that the error in your interpolation $\Delta f(x)$ goes like

$$\Delta f(x) = \frac{1}{2} f''(x)(x - x_i)(x - x_{i+1})$$

- So the error in the interpolation goes like $O(\Delta x^2)$.



- The interpolated value will have some error from the correct value.
- The closer x is to a and b the better the interpolation will be.

QUADRATIC INTERPOLATION

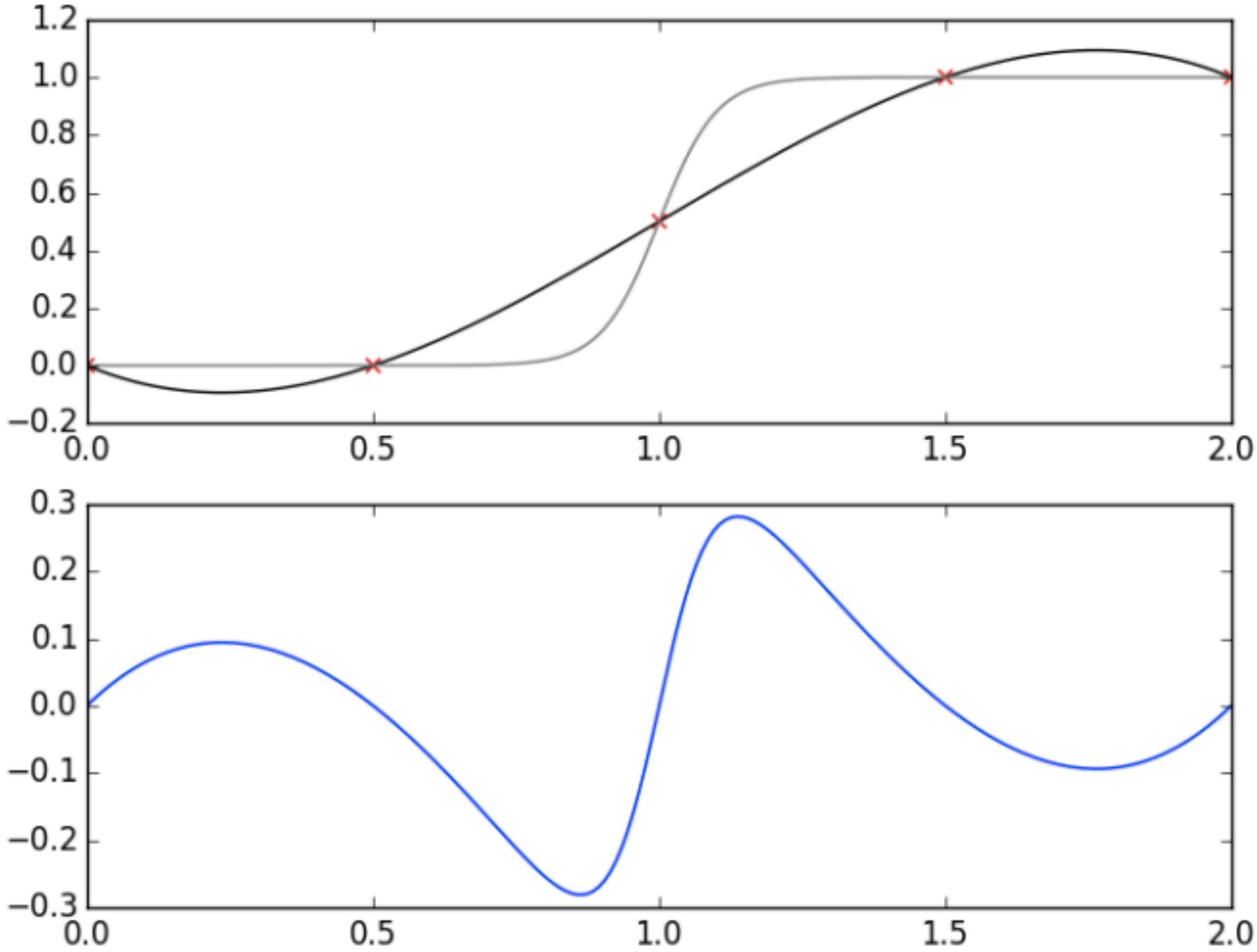
- Next we can fit a quadratic to three points and use that for the interpolation.
- One thing that happens with a quadratic is that our interpolated value can be outside the range of the points we use to generate it. f(x) can be larger or smaller than $f(x_{k-1})$, $f(x_k)$, $f(x_{k+1})$.
- This overshooting or undershooting can cause problems if it gives nonphysical answers. That is if the values should only lie in some range.
- This is one problem with higher order interpolation.

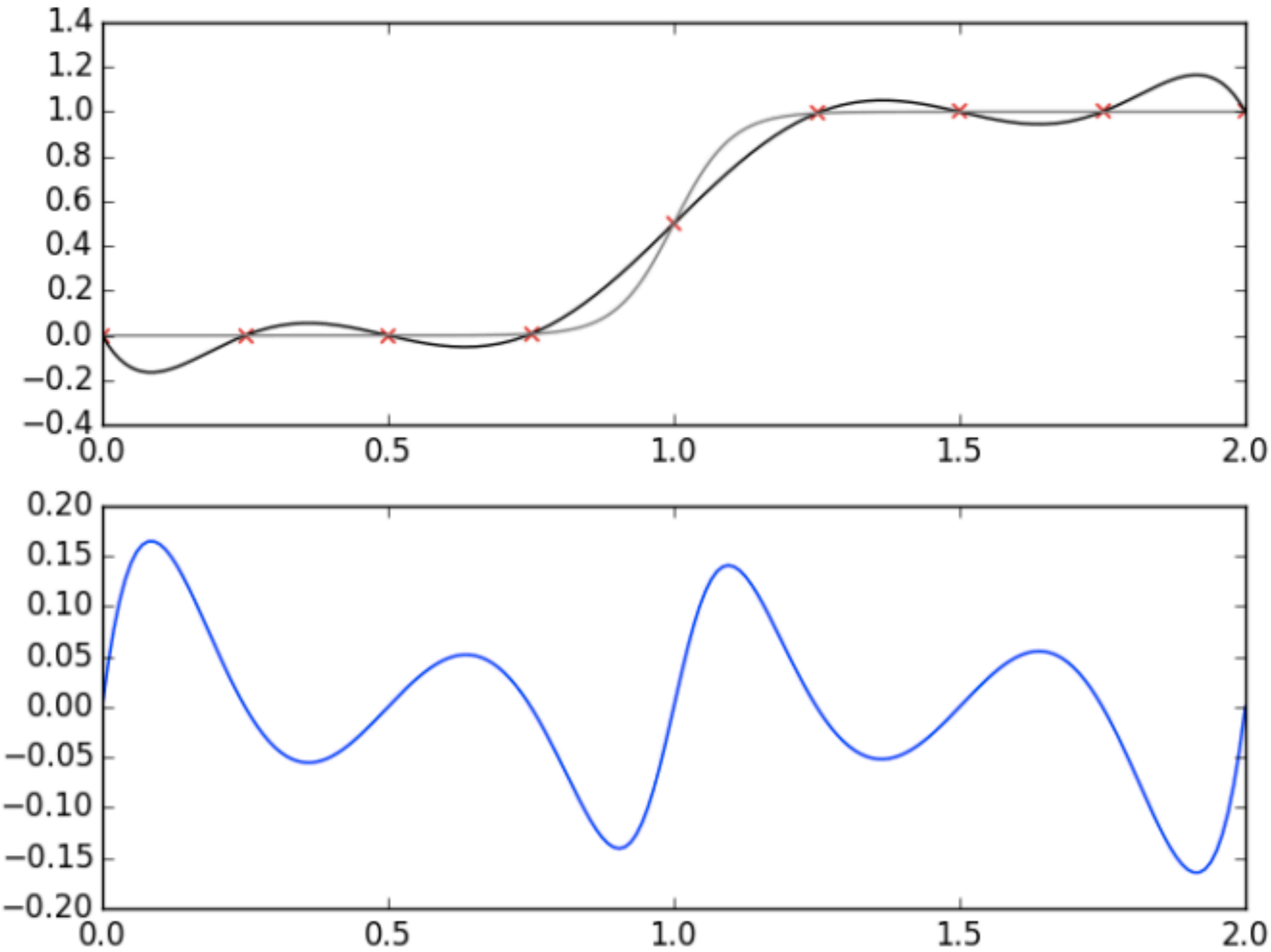
LAGRANGE INTERPOLATION

- We can use Lagrange interpolation like we saw when discussing Gaussian quadrature. That is using the interpolating polynomial.

$$l_i(x) = \prod_{j=0, i \neq j} \frac{x - x_j}{x_i - x_j}$$

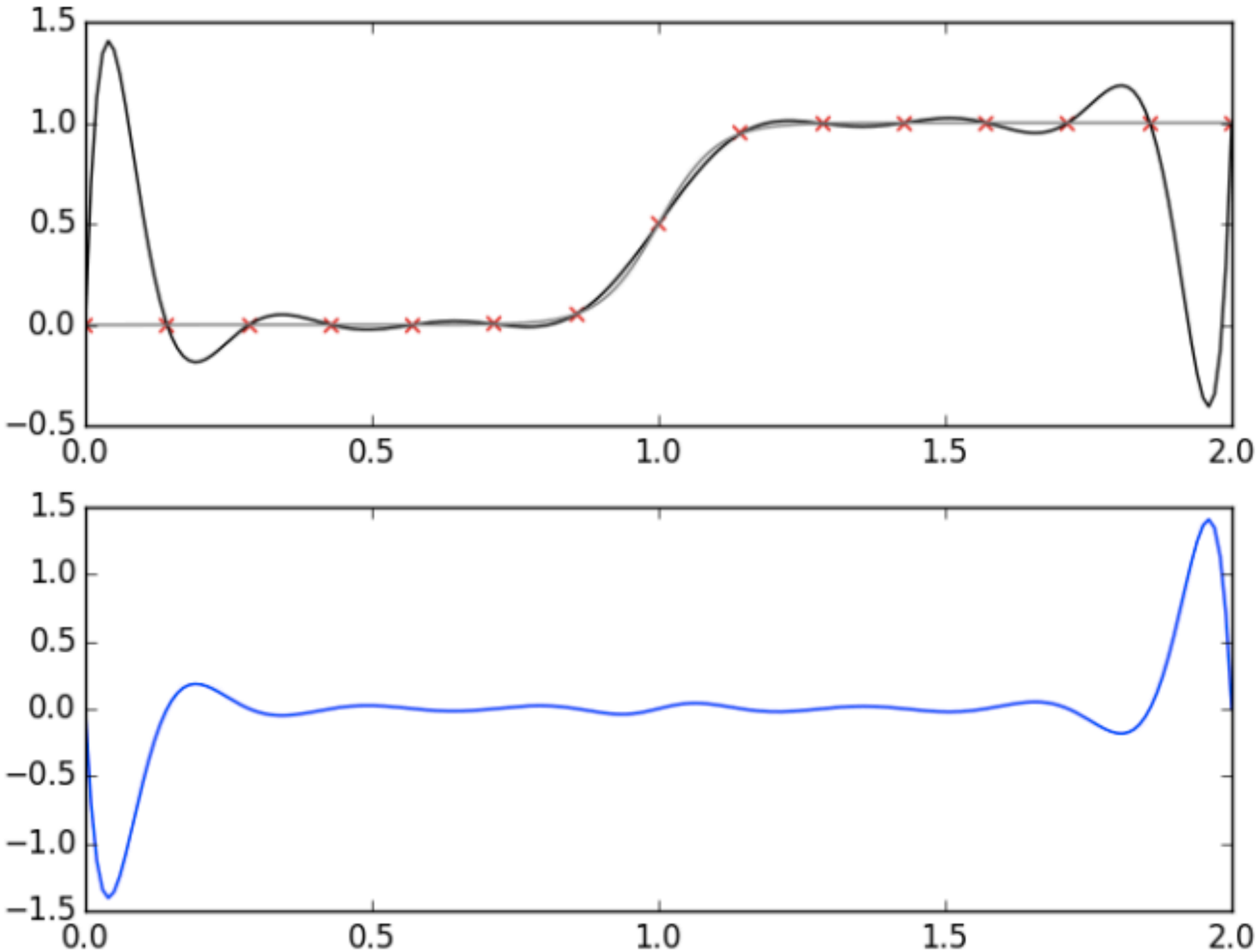
- This function is guaranteed to go through all the points that we use for the fit.
- However, in between the points this function could be much higher or lower. Just because it correctly fits the points we have does not mean it correctly matches the values between those points.


Lagrange interpolation of tanh using n=5 fixed points
1.2
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
0.0
0.5
1.0
1.5
2.0
error
0.3
0.2
0.1
0.0
−0.1
−0.2
−0.3
0.0
0.5
1.0
1.5
2.0


Lagrange interpolation of tanh using n=9 fixed points
1.4
1.2
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4
0.0
0.5
1.0
1.5
2.0

error

0.20
0.15
0.10
0.05
0.00
−0.05
−0.10
−0.15
−0.20
0.0
0.5
1.0
1.5
2.0


Lagrange interpolation of tanh using n=15 fixed points
1.5
1.0
0.5
0.0
−0.5
0.0
0.5
1.0
1.5
2.0

error

1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
0.0
0.5
1.0
1.5
2.0

RUNGE PHENOMENA

- You'll notice that at some moment as the number of points increases the error actually increases.
- This is an example of the Runge phenomena:
  - Oscillation becomes large at the end of the interval with polynomial interpolation.
- One can do better with variable spacing. The problem is that you have very little information at the end points to constrain the fit. Thus high order polynomials will tend to trade wild oscillation at the end points for better fits to the middle points.
- The problem can be thought of as fitting the location of the points but not their derivatives.

SPLINES

- Another way to do interpolation is to match the derivatives of the function at the end points. This is called a spline.
- The most common one used is a cubic spline that matches the first and second derivative at each data point.
- This results in a smooth appearance and avoids the severe oscillations of higher order polynomials.
- The goal here is not to fit a large number of points, but to combine a number of different fits so that they pass though the points and there derivatives match where they connect.
- Does not provide a functional fit to the entire dataset.

EXAMPLE OF A CUBIC SPLINE

- Lets look at a 3 interval / 4 point fit with 3 cubic splines.
  - cubic - m=3
  - Intervals (# of cubics) - n=3
  - We need (m+1)n = 12 constraints.
  - Interior points 1 and 2:

At x_1:

$p_0(x_1) = f_1$

$p_1(x_1) = f_2$

$p'_0(x_1) = p'_1(x_1)$

$p''_0(x_1) = p''_1(x_1)$

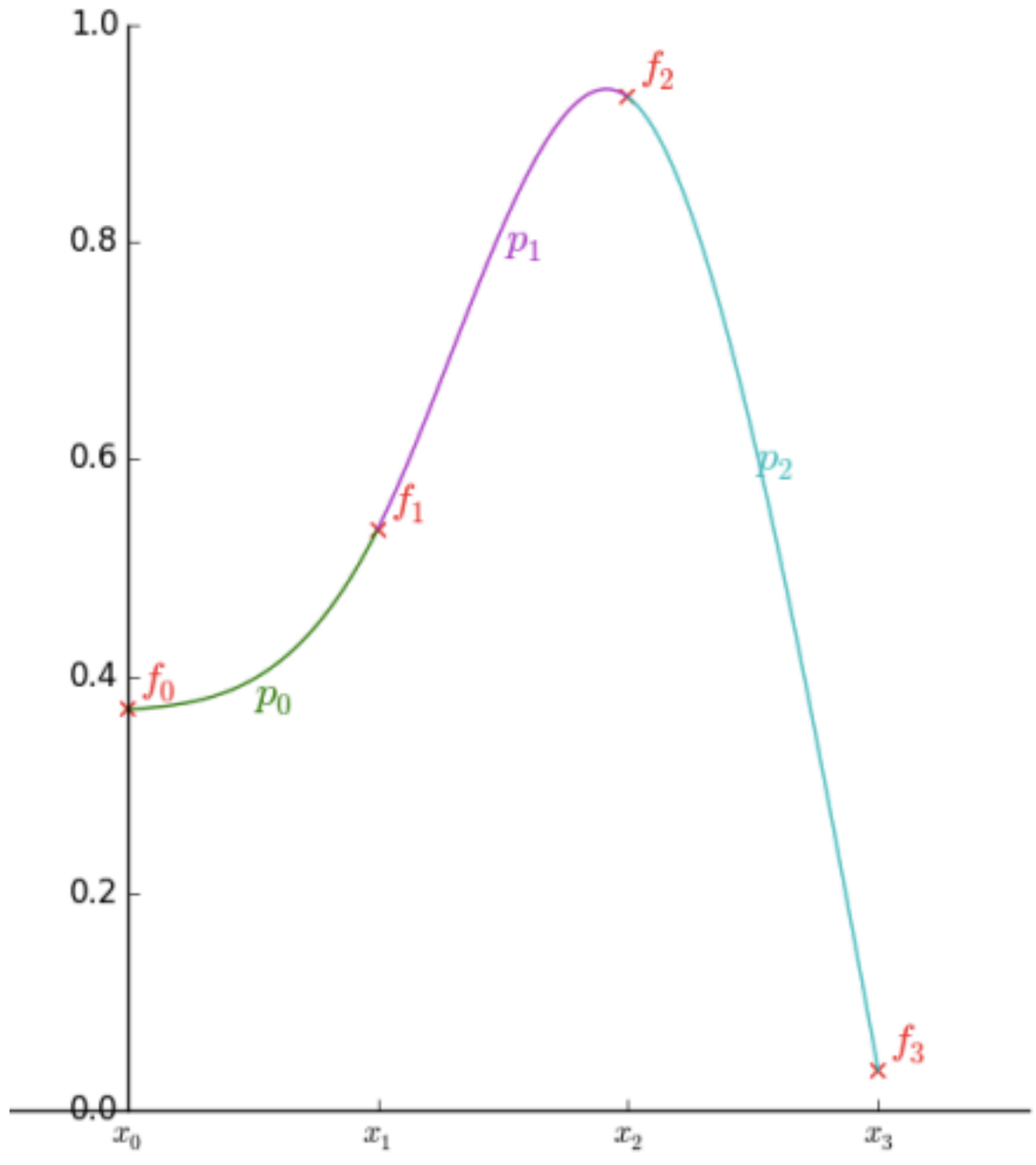
At x_2:

$p_1(x_2) = f_2$

$p_2(x_2) = f_2$

$p'_1(x_2) = p'_2(x_2)$

$p''_1(x_2) = p''_2(x_2)$



EXAMPLE OF A CUBIC SPLINE

- Lets look at a 3 interval / 4 point fit with 3 cubic splines.
  - cubic - m=3
  - Intervals (# of cubics) - n=3
  - We need (m+1)n = 12 constraints.
  - Boundary points 0 and 3:

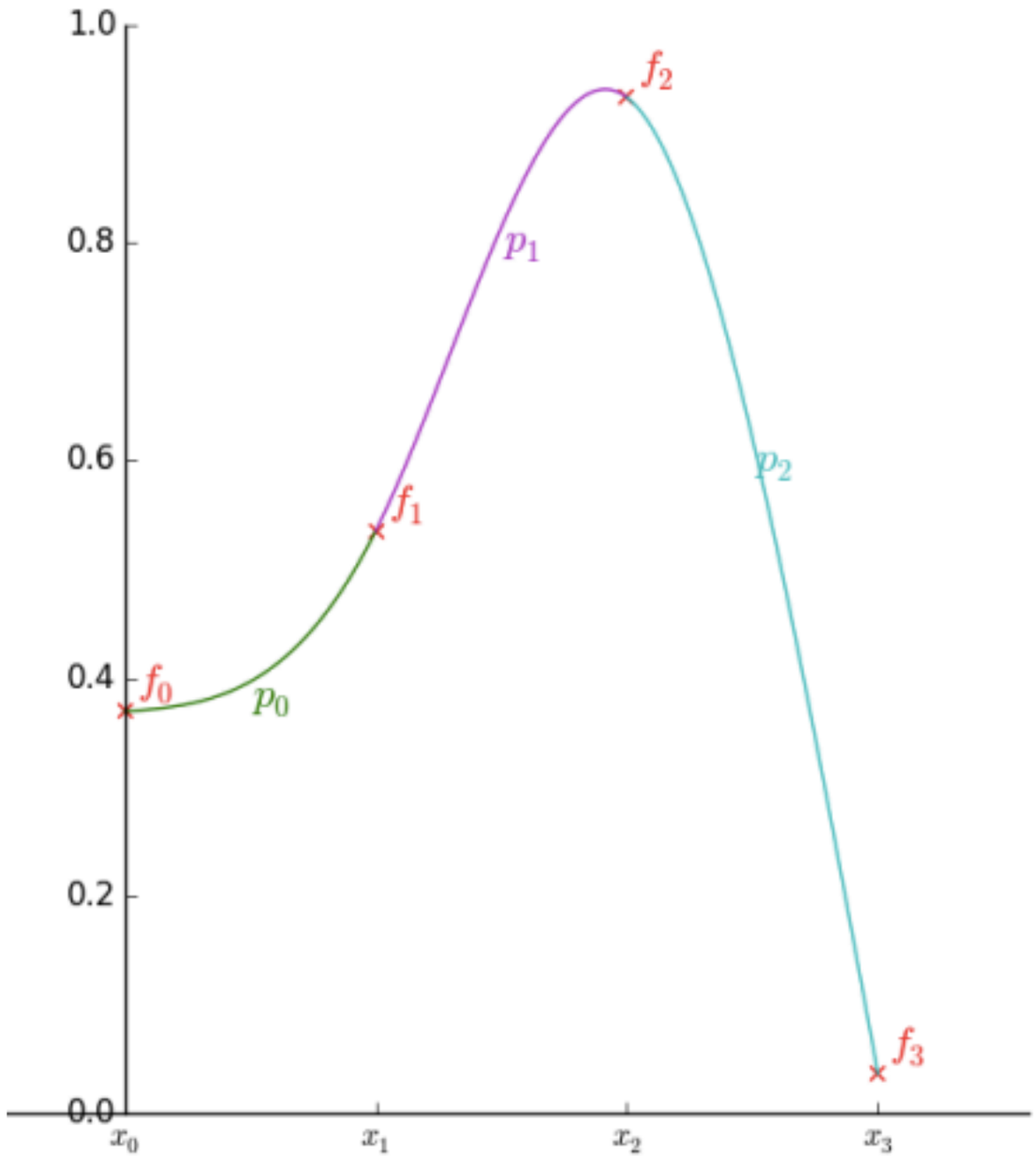
    $p_0(x_o) = f_o$

    $p_2(x_3) = f_3$

  - Still need two constraints, we'll choose the second derivatives to be zero at the boundary

    $p''_0(x_o) = 0$

  - $p''_2(x_3) = 0$



CUBIC SPLINES

- After a lot of algebra we can get the following equation for the splines:

$$p''_{i-1}\Delta x + 4p''_i\Delta x + p''_{i+}\Delta x = \frac{6}{\Delta x}(f(x_{i-1}) - 2f(x_i) + f(x_{i+1}))$$

- while this looks pretty bad it is a linear equation and can be solved with standard linear algebra techniques that we will discuss next section.
- Note: cubic splines are not necessarily the most accurate interpolation scheme (other methods may give smaller values of $\Delta f(x)$).
- But for plotting and graphics applications they look right!

NUMPY.INTERP AND SCIPY.INTERPOLATE

- One can perform linear interpolation using numpy.interp(x, xp, yp), where x are the points you want to interpolate at and xp, yp are the x and y values that you know.
- Alternatively the scipy interpolate sub-package has many functions to call for interpolation. The function interpld() can create an instance of your interpolator and then be used on a point. There are many choices of how to interpolate. f = interpld(x, y, kind='cubic') and then f(x_{new}) returns your y_{new} values using cubic interpolation.
- There are also function for interpolating in higher dimension and performing spline interpolation.

TERMINOLOGY

- Interpolation - approximation for the value of a function using values at known points.
- Lagrange interpolation - a method of interpolation using the interpolating polynomial which goes exactly through all known points.
- Interpolating polynomial - a way of generating a polynomial that goes through a set of given points.
- Spline - a series of functions that go through points but also match their derivatives when they meet.