Polynomial Approximations – Interpolation

The reason people have such interest in polynomials – and in polygons and polyhedra – is because of the following approximation theorem

Weierstrass Approximation Theorem

If f is any continuous function on the finite closed interval $[a,b]$, then for every $\varepsilon > 0$ there exists a polynomial $p(x)$ (whose degree & coefficients depend on ε) such that

$$\max_{x\in[a,b]} |f(x) - p(x)| < \varepsilon .$$

Of course, this theorem doesn't tell us how to construct $p(x)$, or even what the degree of $p(x)$ is.

We will look at the construction of interpolating polynomials, at error estimates, and some difficulties. In practice, one patches together series of low-order polynomial approximations, such as in splines and Bezier patches.

Possible applications

- Modelling scenes with sparse data.
- CAD representations.
- Data compression. (e.g., tables)
  inverse kinematics
- Generating robot trajectories that pass through given set points.

Polynomial forms

$$p(x) = a_0 + a_1 x + a_2 x^2 + \cdots + a_n x^n, \qquad a_n \neq 0$$

degree n

(power form)

Numerical round-off can lead to a loss of significance for x not near zero.

$$p(x) = a_0 + a_1(x-c) + a_2(x-c)^2 + \cdots + a_n(x-c)^n$$

(shifted power form)

Taylor expansion for $p(x)$ about c

$$a_i = \frac{p^{(i)}(c)}{i!}$$

$$\begin{aligned} p(x) = {} & a_0 + a_1(x-c_1) \\ & + a_2(x-c_1)(x-c_2) \\ & + \cdots \\ & + a_n(x-c_1)(x-c_2)\cdots(x-c_n) \end{aligned}$$

(Newton form)

$n + \frac{n(n+1)}{2}$ adds ; $\frac{n(n+1)}{2}$ multiplies

$$p(x) = a_0 + (x-c_1)\Big\{ a_1 + (x-c_2)\Big[a_2 + \cdots + (x-c_{n-1})\big(a_{n-1} + (x-c_n)a_n\big)\Big]\Big\}$$

(Nested Newton form)

$2n$ adds; n multiplies

It is easy to write a recursive algorithm to evaluate $p(x)$ in nested Newton form:

To evaluate $p(x)$ given $x, a_0, \dots, a_n, c_1, \dots, c_n$:

poly_eval $(x;\ a_0, \dots, a_n;\ c_1, \dots, c_n)$
$a_0 + (x-c_1) \cdot$ poly_eval $(x;\ a_1, \dots, a_n;\ c_2, \dots, c_n)$

Suppose $f : [a,b] \to \mathbb{R}$

And suppose $x_0, x_1, \dots, x_n$ are $n+1$ distinct points in $[a,b]$.

We want to construct a polynomial of degree n or less that interpolates $f(x)$ at the points $\{x_i\}$.
That is, we want a poly $p(x)$ such that

$$p(x_i) = f(x_i) \qquad i = 0, \dots, n.$$

There exists a unique such polynomial.
We will exhibit it shortly.
To see uniqueness, suppose p & q are each polynomials of degree at most n that agree at the $\{x_i\}$. Then $p-q$ is a polynomial of degree at most n that vanishes at the $\{x_i\}$. Therefore, by the Fundamental Theorem of Algebra

$$(p-q)(x) = c \prod_{i=0}^{n} (x - x_i)$$

where c is some real number. The LHS has degree at most n. The RHS has degree exactly $n+1$, unless $c=0$. So, $c=0$, and then $p=q$, meaning that p is unique.

Uniqueness tells us that the different ways of writing the interpolating polynomial $p(x)$ are really the same, and must therefore differ primarily on numerical grounds (e.g., compactness, stability, ...).

The **Lagrange Form** is a straightforward way of constructing the interpolating polynomial:

$$p(x) = a_0 \, l_0(x) + a_1 \, l_1(x) + \cdots + a_n \, l_n(x)$$

where each $l_i(x)$ is a polynomial of degree n that satisfies

$$l_i(x_k) = \begin{cases} 1 & \text{if } i = k \\ 0 & \text{if } i \neq k \end{cases} \qquad i = 0, \dots, n$$

and $a_i = f(x_i)$.

In other words,

$$l_i(x) = \prod_{\substack{k=0 \\ k \neq i}}^{n} \frac{x - x_k}{x_i - x_k}$$

<u>Ex</u> Suppose we start with a polynomial, say

$$f(x) = (x-1)^2$$

And suppose we have two values, at $x_0 = 0$ & $x_1 = 1$:

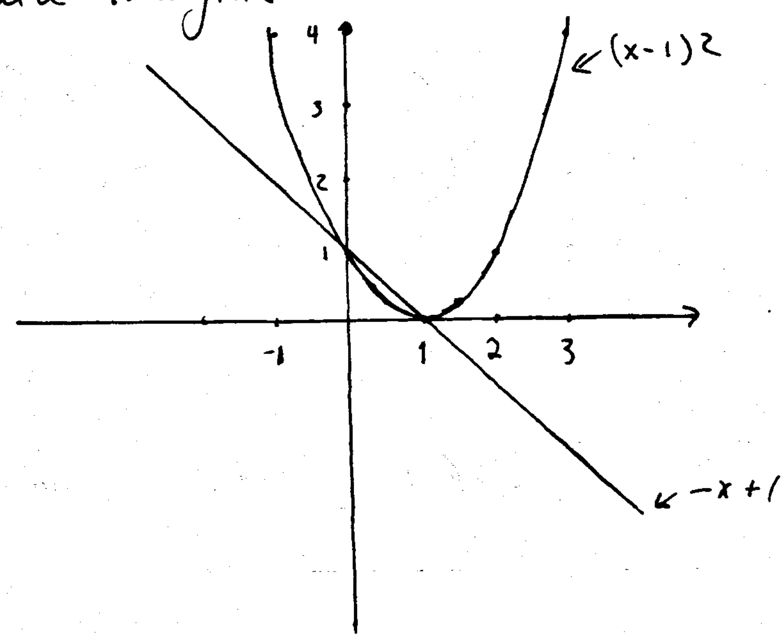
$$f(0) = 1 \implies a_0 = 1$$
$$f(1) = 0 \implies a_1 = 0$$

Now $p(x) = a_0 l_0(x) + a_1 l_1(x)$

Since $a_1 = 0$, we only care about $l_0(x)$.

$$l_0(x) = \frac{x - x_1}{x_0 - x_1} = \frac{x-1}{0-1} = -x+1$$

so $p(x) = -x+1$, which is a line.
This polynomial interpolates $f(x)$, but not very well, as one would imagine



So, let's add the value of f at a third point, say $x=-1$.

$f(-1) = 4$, so

$$x_0 = 0 \qquad a_0 = 1$$
$$x_1 = 1 \qquad a_1 = 0$$
$$x_2 = -1 \qquad a_2 = 4$$

$$l_0(x) = \frac{x - x_1}{x_0 - x_1} \cdot \frac{x - x_2}{x_0 - x_2} = -(x-1)(x+1)$$

$$l_1(x) = \frac{x - x_0}{x_1 - x_0} \cdot \frac{x - x_2}{x_1 - x_2} = \text{we don't care}$$

$$l_2(x) = \frac{x - x_0}{x_2 - x_0} \cdot \frac{x - x_1}{x_2 - x_1} = \tfrac{1}{2} x (x-1)$$

Now, $p(x) = a_0 l_0(x) + a_1 l_1(x) + a_2 l_2(x)$

$$= -(x-1)(x+1) + 2x(x-1)$$
$$= (x-1)\left[2x - (x+1)\right]$$
$$= (x-1)(x-1)$$
$$= (x-1)^2$$

So, as one would hope, this approximation is exact.

Comments

- The goodness of an approximation depends on the number of approximating points and also on their location.

  We will look at error estimates.

- One problem with the Lagrange form of the interpolating polynomial is that we need

  $$2(n+1) \quad \text{mul/div}$$
  $$\& \quad 2n+1 \quad \text{add/sub}$$

  to evaluate $p(x)$ at a given point x, <u>even after</u> all the the denominators have been calculated once and for all and divided into the a_i values.

  If we could write the interpolating poly in nested form we could get away with half that many mul operations. — We won't worry about this.

- Another problem is that in practice, one may be uncertain as to how many interpolation points to use. So, one may want to increase them over time and see whether the approximation gets better. In doing so, one would like to use the old approximation. It isn't clear how to do that easily with the Lagrange form. We will look at this

Divided Differences

This is basically a way of writing the interpolating polynomial in Newton form.

Let $p_n(x)$ denote the interpolating polynomial of degree n (or less), that interpolates $f(x)$ at $x_0, \dots, x_n$

Suppose we write $p_n(x)$ as

$$p_n(x) = A_0 + A_1(x-x_0) + A_2(x-x_0)(x-x_1) + \cdots + A_n(x-x_0)\cdots(x-x_{n-1})$$

Now notice that this last term vanishes at $x_0, \dots, x_{n-1}$

Therefore, the previous n terms must comprise the interpolating polynomial of degree $n-1$ (or less) that agrees with $f(x)$ at $x_0, \dots, x_{n-1}$. In other words,

$$p_{n-1}(x) = A_0 + A_1(x-x_0) + \cdots + A_{n-1}(x-x_0)\cdots(x-x_{n-2})$$

with same $\{A_i\}$ coefficients as for $p_n(x)$.

In short, $\quad p_n(x) = p_{n-1}(x) + A_n(x-x_0)\cdots(x-x_{n-1})$,

So, we can build $p_n(x)$ from $p_{n-1}(x)$.

Now, what are the $\{A_i\}$?

Suppose we define $f[x_i, x_{i+1}, \ldots, x_m]$ to be the leading coefficient of the polynomial of degree $m-i$ (or less) that agrees with $f(x)$ at the distinct points $x_i, \ldots, x_m$.

Then by construction, $A_k = f[x_0, \ldots, x_k]$

called the k^{th} *divided difference* of f at the points $x_0, \ldots, x_k$

Observe that $f[x_0] = f(x_0)$, and that $f[x_0, x_1] = \dfrac{f(x_0) - f(x_1)}{x_0 - x_1}$

constant approximation — approximation to the first derivative.

More generally, we will see that

$$f[x_0, \ldots, x_k] = \frac{f[x_0, \ldots, x_{k-1}] - f[x_1, \ldots, x_k]}{x_0 - x_k}$$

This is easily expressed as a table:

	$f[\ ]$	$f[\ ,\ ]$	$f[\ ,\ ,\ ]$	$f[\ ,\ ,\ ,\ ]$	$f[\ ,\ ,\ ,\ ,\ ]$
x_0	$f(x_0)$	$f[x_0, x_1]$			
x_1	$f(x_1)$		$f[x_0, x_1, x_2]$		
x_2	$f(x_2)$	$f[x_1, x_2]$	$f[x_1, x_2, x_3]$	$f[x_0, x_1, x_2, x_3]$	$f[x_0, x_1, x_2, x_3, x_4]$
x_3	$f(x_3)$	$f[x_2, x_3]$		$f[x_1, x_2, x_3, x_4]$	
x_4	$f(x_4)$	$f[x_3, x_4]$	$f[x_2, x_3, x_4]$		

The previous method uses a table to compute the coefficients of the polynomials $\{p_n\}$.

A similar triangular table can be used to iteratively build up the polynomials directly. See §3.1 of NRiC.

Indeed that table is based on a recursive formula used to prove correctness of the formulas for the A_k:

Let us verify the formula for $A_k \left(= f[x_0, \ldots, x_k]\right)$, namely that

$$A_k = \frac{f[x_0, \ldots, x_{k-1}] - f[x_1, \ldots, x_k]}{x_0 - x_k}.$$

Note by the way that it is clear for $k=0$ or 1.
Indeed, for $k=1$, the interpolating polynomial is

$$p_1(x) = f(x_0) + \underbrace{\frac{f(x_1) - f(x_0)}{x_1 - x_0}}_{\text{leading coefficient}} (x - x_0)$$

More generally, let us define two sets of polynomials:

$p_{k-1}(x)$ is the poly of degree $k-1$ (or less) which agrees with $f(x)$ at $x_0, \ldots, x_{k-1}$

$q_{k-1}(x)$ is the poly of degree $k-1$ (or less) which agrees with $f(x)$ at $x_1, \ldots, x_k$

The key recursive formula is:

$$p_k(x) = \frac{x - x_0}{x_k - x_0} q_{k-1}(x) + \frac{x_k - x}{x_k - x_0} p_{k-1}(x)$$

Now observe that

$$\begin{aligned} A_k &= f[x_0, \dots, x_k] \\ &= \text{leading coeff of } p_k(x) \\ &= \frac{\text{leading coeff of } q_{k-1}(x)}{x_k - x_0} - \frac{\text{leading coeff of } p_{k-1}(x)}{x_k - x_0} \\ &= \frac{f[x_1, \dots, x_k] - f[x_0, \dots, x_{k-1}]}{x_k - x_0} \end{aligned}$$

<u>Ex</u> $f(x) = (x-1)^2$

x_i	$f[\ ]$	$f[\ ,\ ]$	$f[\ ,\ ,\ ]$
$x_0 = 0$	1		
		$\frac{1-0}{0-1} = -1$	
$x_1 = 1$	0		$\frac{-1 - -2}{0 - -1} = 1$
		$\frac{0-4}{1 - -1} = -2$	
$x_2 = -1$	4		

So, just using the points x_0 & x_1, we get

$$p_1(x) = f[x_0] + f[x_0, x_1](x - x_0)$$

$$= 1 + -1\,(x - 0)$$

$$= 1 - x \quad , \text{ as before.}$$

Using all three points $x_0, x_1, \& x_2$, we get

$$p_2(x) = f[x_0] + f[x_0, x_1](x - x_0) + f[x_0, x_1, x_2](x - x_0)(x - x_1)$$

$$= p_1(x) + f[x_0, x_1, x_2](x - x_0)(x - x_1)$$

note the recursive ease

$$= 1 - x + 1 \cdot (x - 0)(x - 1)$$

$$= (x - 1)^2 , \quad \text{as before.}$$

Error of the interpolating polynomial

The interpolation error of $p_n(x)$ is given by:

$$e_n(x) = f(x) - p_n(x).$$

Suppose $p_n(x)$ interpolates $f(x)$ at $x_0, x_1, \ldots, x_n$.

Let $\bar{x}$ be any point different from these.
If $p_{n+1}(x)$ interpolates $f(x)$ at $x_0, x_1, \ldots, x_n, \bar{x}$, then

$$p_{n+1}(x) = p_n(x) + f[x_0, x_1, \ldots, x_n, \bar{x}] \prod_{i=0}^{n} (x - x_i)$$

Therefore, for all $\bar{x} \notin \{x_0, \ldots, x_n\}$

$$e_n(\bar{x}) = f[x_0, x_1, \ldots, x_n, \bar{x}] \prod_{i=0}^{n} (\bar{x} - x_i)$$

This shows that the error is the "next" term in the Newton form.

Of course, without knowing $f(\bar{x})$ we do not know $f[x_0, \ldots, x_n, \bar{x}]$, and therefore we do not know how big the error is.

However, two theorems will allow us to estimate the error in certain cases.

Th$^{\underline{m}}$ Suppose $f: [a,b] \to \mathbb{R}$ is k times differentiable. If $x_0, \dots, x_k$ are $k+1$ distinct points in $[a,b]$, then $\exists\ \xi \in (a,b)$ such that

$$f[x_0, \dots, x_k] = \frac{f^{(k)}(\xi)}{k!}$$

So, if we have a bound on the k^{th} derivative of f, then we can get a handle on the error e_{k-1}. Indeed:

Th$^{\underline{m}}$ Suppose $f: [a,b] \to \mathbb{R}$ is $n+1$ time differentiable. If $p_n(x)$ interpolates $f(x)$ at $x_0, \dots, x_n$, then for every $\bar{x} \in [a,b]$ there exists $\xi \in (a,b)$ such that

$$e_n(\bar{x}) = \frac{f^{(n+1)}(\xi)}{(n+1)!} \prod_{i=0}^{n} (\bar{x} - x_i)$$

Note: ξ depends on $\bar{x}$.

Ex (CdB, 2.7)

Determine the spacing h in a table of evenly spaced values of the function $f(x) = \sqrt{x}$ between 1 & 2, so that interpolation with a second-degree polynomial in this table will yield a desired accuracy.

Soln: Table contains the values $f(x_i)$, $i = 0, \ldots, n$ with $n = \frac{1}{h}$ and $x_i = 1 + i \cdot h$.

If $\bar{x} \in [x_{i-1}, x_{i+1}]$ then we approximate $f(\bar{x})$ with $p_2(\bar{x})$ where $p_2(\bar{x})$ is the poly of degree 2 that interpolates f at x_{i-1}, x_i, x_{i+1}.

By the previous theorem the error is

$$e_2(\bar{x}) = \frac{f'''(\xi)}{3!} (\bar{x} - x_{i-1})(\bar{x} - x_i)(\bar{x} - x_{i+1})$$

where ξ depends on $\bar{x}$.

We don't know ξ, but we can estimate:

$$|f'''(\xi)| \le \max_{1 \le x \le 2} |f'''(x)|$$

$$= \max_{1 \le x \le 2} \frac{3}{8} x^{-\frac{5}{2}}$$

$$= \frac{3}{8}$$

And $\left|(\bar{x}-x_{i-1})(\bar{x}-x_i)(\bar{x}-x_{i+1})\right| \le \max_{y\in[-h,h]} \left|(y-h)\,y\,(y+h)\right|$

(for $\bar{x}\in[x_{i-1},x_{i+1}]$)

$$= \frac{2}{3}\frac{1}{\sqrt{3}}h^3$$

(Why? Well consider the function $g(y) = (y-h)y(y+h)$
$= y^3 - yh^2$

$$g(h) = g(-h) = g(0) = 0$$

So $g(h)$ achieves its maximum on $[-h,h]$ in the interior.

$g'(h) = 3y^2 - h^2$

$g'(h) = 0$ iff $3y^2 = h^2$ iff $y = \pm \frac{h}{\sqrt{3}}$

$g\left(\pm\frac{h}{\sqrt{3}}\right) = \pm\frac{h}{\sqrt{3}}\left(\frac{h^2}{3} - h^2\right) = \mp \frac{2}{3}\frac{1}{\sqrt{3}}h^3$

So, we see that $\left|e_2(\bar{x})\right| \le \frac{1}{3!}\cdot\frac{3}{8}\cdot\frac{2}{3}\frac{1}{\sqrt{3}}h^3$

$$= \frac{h^3}{24\sqrt{3}}$$

Consequently, if we want, say, 7 place accuracy, we should choose h such that

$$\frac{h^3}{24\sqrt{3}} < 5\cdot 10^{-8}$$

This yields $h \approx 0.0127619$

i.e., the number of table entries is about 79.

+ 1 = 80

(There are 79 intervals, so 80 data entries.)

Some interesting facts (taken from Forsythe et al.)

We might hope that as we increase the number of interpolation points we get a better approximation to $f(x)$ over its interval $[a,b]$ of definition.
It turns out that this need not be the case:

Conside the triangular array of points in (a,b):

Row 0: x_0^0

Row 1: $x_0^1 \quad x_1^1$

Row 2: $x_0^2 \quad x_1^2 \quad x_2^2$

$\vdots$

Row n: $x_0^n \quad x_1^n \quad x_2^n \quad \cdots \quad x_n^n$

$\vdots$

The points in each row are distinct.

This is called an interpolating array.
The array specifies the interpolation points to be used in the following interpolation scheme:

If we want an n^{th} order interpolating polynomial, then we interpolate $f(x)$ using Row n, that is, at the points $x_0^n, \dots, x_n^n$.

Denote this polynomial by $P_n(f)(x)$.

Faber's Theorem

For every interpolating array there exists a continuous function $g: [a,b] \to \mathbb{R}$ and there exists a point $x \in (a,b)$ such that

$P_n(g)(x)$ does <u>not</u> converge to $g(x)$ as $n \to \infty$.

In other words, for any preset interpolating schem there is some continuous function on which the scheme fails to produce good convergence.

Often this failure of convergence is very apparent with uniformly spaced interpolating points.

These observations suggest that we should be wary of higher-order interpolating polynomials. And we certainly shouldn't put much faith in extrapolation. — These worries are a reason why many prefer splines.

Fortunately, it turns out that for certain wisely-chosen interpolating arrays, the polynomials $P_n(f)(x)$ do converge to $f(x)$ at every $x \in (a,b)$ so long as f has a continuous first derivative on $[a,b]$.

One such interpolating array is given by the <u>Chebyshev points</u> for the interval $[a,b]$:

$$\text{Row } n: \quad x_i^n = \frac{1}{2}\left[a+b+(a-b)\cos\frac{2i+1}{2n+2}\pi\right], \; i=0,\dots,n$$

Indeed, it turns out that these points are nearly optimal, in the sense that the resulting polynomial $P_n(f)(x)$ nearly attains the minimum global error possible:

$$\min_{\substack{\text{polys } p(x) \\ \text{of degree } n}} \; \max_{x\in[a,b]} \left| f(x) - p(x) \right|$$

"Nearly" means that the error is some known small multiple of the best possible error (for $n \le 50$, the multiple is less than 4.5).

<u>One useful trick</u>: Interpolate using $n+1$ Chebyshev points. This yields a polynomial of degree n (or less). Now pick a new point and reinterpolate using these $n+2$ points. (By the recursion formula this is easier than generating the polynomial of degree $n+1$ (or less) on the $n+2$ different Chebyshev points of row $n+1$.) Often this degree $n+1$ poly will attain the min error possible for degree n polys.

<u>Another useful approach</u>:

The <u>expanded Chebyshev points</u> yield an interpolating error within 0.02 of the best possible error.

$$\text{Row } n: \quad \hat{x}_i^n = \frac{1}{2}\left[a+b+(a-b)\frac{\cos\frac{2i+1}{2n+2}\pi}{\cos\frac{\pi}{2n+2}}\right], \; i=0,\dots,n.$$

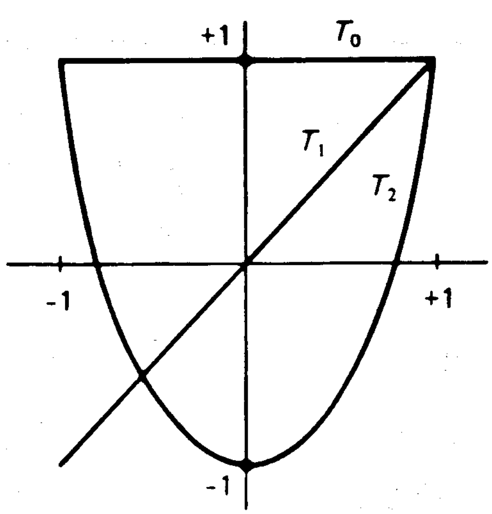
The Chebyshev points on $[-1,1]$, $\{x_i^n\}$ are the zeros of the Chebyshev polynomial $T_{n+1}(x)$ of degree $n+1$.

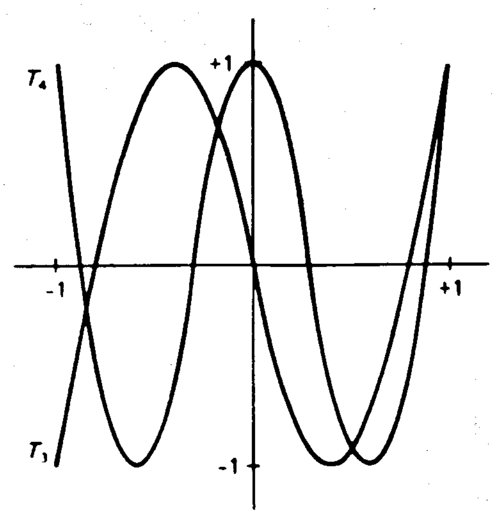
$T_k(x)$ is defined by: $T_k(\cos\theta) = \cos k\theta$.

So:

$$\begin{aligned}
T_0(x) &= 1 \\
T_1(x) &= x \\
T_2(x) &= 2x^2 - 1 \\
T_3(x) &= 4x^3 - 3x \\
T_4(x) &= 8x^4 - 8x^2 + 1 \\
T_5(x) &= 16x^5 - 20x^3 + 5x \\
T_6(x) &= 32x^6 - 48x^4 + 18x^2 - 1 \\
T_7(x) &= 64x^7 - 112x^5 + 56x^3 - 7x
\end{aligned}$$

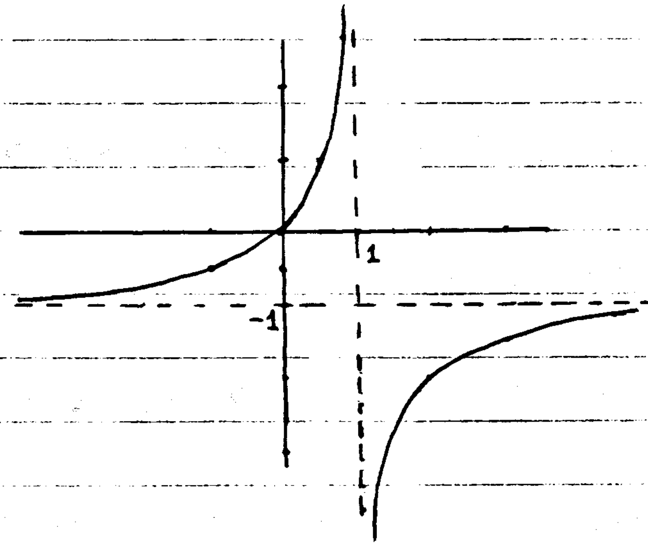
Note the uneven spacing of the zeros. Also note that the zeros (i.e., the interpolating points) are different for different n.





Polynomial interpolation does not work well for functions with poles. In such cases, it is often beneficial to use <u>rational function approximations.</u> (A rational function is a ratio of polynomials.)

<u>Ex</u> Consider $f(x) = \frac{x}{1-x}$



Suppose we wish to interpolate this function at the three points $x_0 = -1,\ x_1 = 0,\ x_2 = 2$.

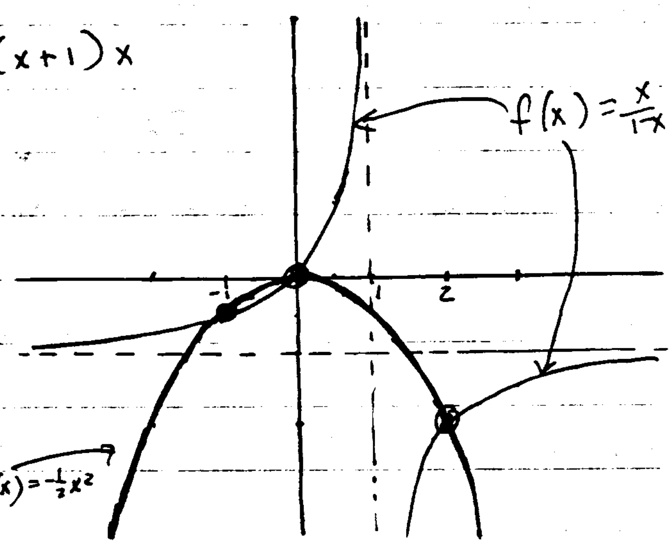
First, let's consider using the interpolating polynomial $p_2(x)$.

x_i	$f[\ ]$	$f[\ ,\ ]$	$f[\ ,\ ,\ ]$
$x_0 = -1$	$-\frac{1}{2}$		
		$\frac{1}{2}$	
$x_1 = 0$	0		$-\frac{1}{2}$
		-1	
$x_2 = 2$	-2		

so $p_2(x) = -\frac{1}{2} + \frac{1}{2}(x+1) - \frac{1}{2}(x+1)x$
$\qquad\quad = -\frac{1}{2}x^2$

The match isn't great in the range $[0,2]$, which crosses the pole:

(And, of course, we shouldn't trust extrapolating outside the range $[-1,2]$.)



Suppose we use the same three points to interpolate using a rational function. This allows us to use rational functions of the form

$$\frac{a_0 + a_1 x}{b_0 + b_1 x} \quad \text{or} \quad \frac{a_0}{b_0 + b_1 x + b_2 x^2}$$

[There are really only 3 unknowns in each of these, since, e.g., only the ratio $\frac{a_0}{b_0}$ matters, not each of a_0 and b_0.]

If we have reason to believe the pole is first order, we would use

$$R(x) = \frac{a_0 + a_1 x}{b_0 + b_1 x}.$$

Now we want $R(x_i) = f(x_i)$, $i = 0, 1, 2$.

So:

$$\frac{a_0 - a_1}{b_0 - b_1} = -\frac{1}{2} \qquad \frac{a_0}{b_0} = 0 \qquad \frac{a_0 + 2a_1}{b_0 + 2b_1} = -2$$

We can arbitrarily set $b_0 = 1$. Then $a_0 = 0$ and

$$\frac{-a_1}{1 - b_1} = -\frac{1}{2} \qquad\qquad \frac{2a_1}{1 + 2b_1} = -2$$

$$\Downarrow \qquad\qquad\qquad \Downarrow$$

$$2a_1 = 1 - b_1 \qquad\qquad 2a_1 = -2 - 4b_1$$

So $1 - b_1 = -2 - 4b_1 \Rightarrow b_1 = -1$

$\therefore$ $a_1 = 1$

So $R(x) = \dfrac{0 + 1x}{1 - 1x} = \dfrac{x}{1 - x}$, which happens to be exact