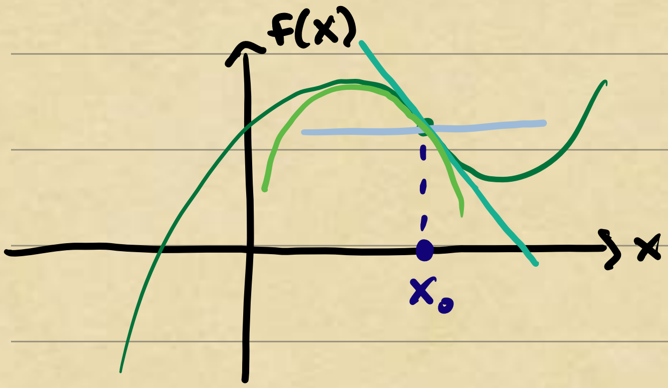
Intro to Linear Approximation

Idea: build complex functions from simple ones.

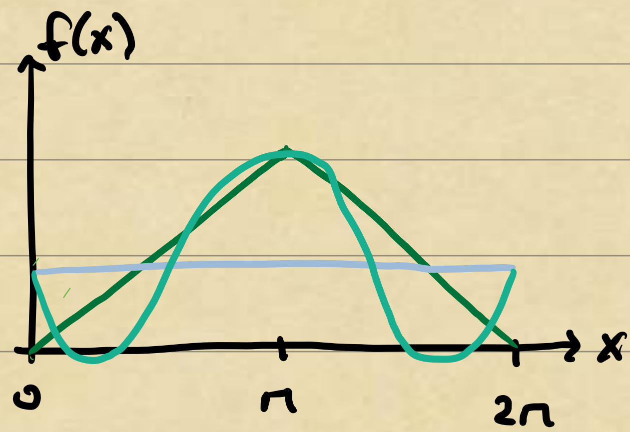
Example 1: Taylor Polynomials



$$f(x) \approx f(x_0) + f'(x_0)(x-x_0) + \frac{1}{2} f''(x_0)(x-x_0)^2 + \cdots + \frac{1}{n!} f^{(n)}(x_0)(x-x_0)^n$$

Build up $f(x)$ near x_0 using monomials $(x-x_0)^j$.

Example 2: Fourier Series



$$f(x) \approx \frac{a_0}{2} + a_1 \cos(x) + a_2 \cos(2x) + \cdots + a_n \cos(nx)$$

discuss coeffs later

Build up $f(x)$ using trigonometric functions.

In both examples, $f(x)$ is built up using linear combinations of simpler functions.

$$f(x) \approx c_1 e_1(x) + c_2 e_2(x) + \cdots + c_n e_n(x)$$

Q: Does this remind you of a math. structure?

$\Rightarrow$ vector spaces!

We can use tools from linear algebra to develop both the theory of these approximations and practical algorithms.

Example 3: "Best" Approximation

Suppose we have a "dictionary" of functions and we want to find the "best" approximation to a function $f:[0,1]\to\mathbb{R}$ by combining

$$(*) \qquad f(x) \approx c_1 e_1(x) + c_2 e_2(x) + \cdots + c_n e_n(x).$$

Let's rewrite (*) in a suggestive form:

$$\begin{array}{c} 0 \\ \uparrow \\ x \\ \downarrow \\ 1 \end{array} \begin{bmatrix} | \\ f(x) \\ | \end{bmatrix} \approx \begin{bmatrix} | & | & & | \\ e_1(x) & e_2(x) & \cdots & e_n(x) \\ | & | & & | \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_n \end{bmatrix}$$

"Quasimatrix"

<u>Q:</u> What linear algebra task does this look like?

=> Rectangular Least-Squares!

We need a few more tools before we solve (*), but we can solve a related practical problem right away.

<u>Example 4: Regression Models</u>

We often only have samples of f:

$$f_1 = f(x_1),\ f_2 = f(x_2),\ \ldots,\ f_m = f(x_n)$$

Evaluating (*) at the sample points gives

$$(\text{NL}) \quad \begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ f_m \end{bmatrix} \approx \begin{bmatrix} e_1(x_1) & e_2(x_1) & \cdots & e_n(x_1) \\ e_1(x_2) & e_2(x_2) & \cdots & e_n(x_2) \\ \vdots & \vdots & & \vdots \\ e_1(x_m) & e_2(x_m) & \cdots & e_n(x_m) \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_m \end{bmatrix}$$

$$\underline{f} \approx \underline{E} \quad \underline{c}$$

Q: Under what conditions is it possible to fit the data exactly (interpolation)?

⇒ If $\underline{f}$ is in the column space of E, i.e., can be written as a linear combo of the columns of E.

⇒ If $n = m = \text{rank}(E)$, i.e., E is square with linearly independent columns.

Q: How well does the interpolant approx. f?

⇒ Need to incorporate ideas from analysis. Roughly, depends on how "close" f is to $\text{span}\{e_1, \dots, e_n\}$.

In many data-driven settings, we have $m \gg n$ and interpolating the data is a questionable goal.

Q: What should we do if we can't?

"optimize"! $\Rightarrow \underline{c} = \underset{\underline{x}\in\mathbb{R}^n}{\operatorname{argmin}} \overbrace{\|\underline{f} - E\underline{x}\|}^{\text{size of residual}}$

Q: What do we mean by $\|\cdot\|$? I.e., how should we measure size?

$\Rightarrow$ We need to introduce the idea of a normed vector space

$\Rightarrow$ The choice of <u>which</u> norm strongly influences the features of the minimizer, our "best" fit.

$\Rightarrow$ Most norms lead to nonlinear approx. but special norms lead to linear algebra.

To control the accuracy of our approximations, we will typically need to add more functions to our dictionary and/or gather more data/samples.

Key Ideas (Analysis): limits, smoothness, convergence.

Q: How does Taylor approximation improve as we add more terms, i.e., as n increases?

Remainder Formula (Taylor Series)

Taylor Polynomial $\quad f_n(x) = f(x_0) + f'(x_0)(x-x_0) + \cdots + \frac{1}{n!} f^{(n)}(x_0)(x-x_0)^n$

Claim: If f has $n+1$ continuous derivatives in a neighborhood $I_\delta = [x_0-\delta, x_0+\delta]$ of x_0, then $(n\geq 1)$

(R) $$f(x) - f_n(x) = \frac{1}{n!}\int_{x_0}^{x} f^{(n+1)}(t)(x-t)^n\,dt, \quad x \in I_\delta.$$

The remainder formula allows us to estimate

max error

local improvement

(E) $$\sup_{x\in I_\delta} |f(x) - f_n(x)| \leq \frac{\delta^{n+1}}{(n+1)!} \sup_{x\in I_\delta} |f^{(n+1)}(x)|.$$

"size" of local fluctuations

pf| The proof combines the fundamental theorem of calculus with integration-by-parts.

Key Idea 1: Fundamental thm of calc. (F.T.C.)

$$f(x) = f(x_0) + \int_{x_0}^{x} f'(t)\,dt$$

Key Idea 2: Integration-by-parts. (I.B.P.)

$$f(x) = f(x_0) + \left[t f'(t)\right]_{x_0}^{x} - \int_{x_0}^{x} f''(t)\, t\, dt$$

$$= f(x_0) + x f'(x) - x_0 f'(x_0) - \int_{x_0}^{x} f''(t)\, t\, dt$$

Now, we apply F.T.C. again to simplify.

$$x f'(x) = x f'(x_0) + x \int_{x_0}^{x} f''(t)\, dt$$

$$= f(x_0) + f'(x_0) x + \int_{x_0}^{x} f''(t)(x-t)\, dt$$

Now, repeatedly apply I.B.P. to integral remainder. E.g., for the $n=2$ case we proceed to calculate

$$\int_{x_0}^{x} f''(t)(x-t)\, dt = \frac{1}{2} f''(x_0)(x-x_0)^2 + \frac{1}{2}\int_{x_0}^{x} f^{(3)}(t)(x-t)^2\, dt$$

$$f(x) = f(x_0) + f'(x_0)(x-x_0) + \frac{1}{2} f''(x_0)(x-x_0)^2 + \frac{1}{2}\int_{x_0}^{x} f^{(3)}(t)(x-t)^3\, dt$$

To prove the general case, proceed by induction. Suppose that (R) holds for $n=k \geq 1$, we'll show that it must also hold for $k+1$.

Since we already know it holds for $n=1$, this implies it also holds for $n=2,3,4,\ldots$

$$f(x) = f_k(x) + \frac{1}{k!}\int_{x_0}^{x} f^{(k+1)}(t)(x-t)^k\,dt$$

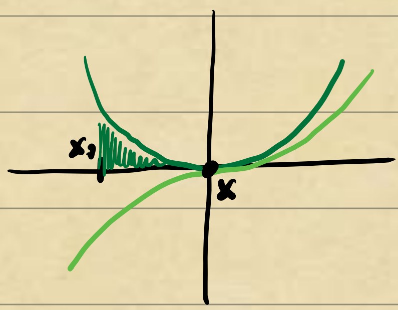
$$\int_{x_0}^{x} f^{(k+1)}(t)(x-t)^k\,dt = \frac{1}{k+1} f^{(k+1)}(x_0)(x-x_0)^{k+1} + \frac{1}{k+1}\int_{x_0}^{x} f^{(k+2)}(t)(x-t)^{k+1}\,dt$$

$$\Rightarrow \quad f(x) = f_{k+1}(x) + \frac{1}{(k+1)!}\int_{x_0}^{x} f^{(k+2)}(t)(x-t)^{k+1}\,dt$$

This establishes the remainder formula. Now, let's estimate the size of the remainder.

$$\sup_{x\in I_\delta} |f(x) - f_n(x)| \leq \frac{1}{n!}\sup_{x\in I_\delta}\left|\int_{x_0}^{x} f^{(n+1)}(t)(x-t)^n\,dt\right|$$

$$\leq \frac{1}{n!}\sup_{t\in I_\delta}|f^{(n+1)}(t)|\int_{x_0}^{x}|x-t|^n\,dt$$



$$\leq \frac{1}{n!}\sup_{x\in I_\delta}|f^{(n+1)}(x)|\,\frac{|x-x_0|^{n+1}}{n+1}$$

$$\leq \frac{\delta^{n+1}}{(n+1)!}\sup_{x\in I_\delta}|f^{(n+1)}(x)|.$$

HW!

Q: What is the analogous estimate for Fourier?