Announcements

- exam 3
- use link in exam to go to Jupyterlab NOT relate... / lab
-

Review

$$x_{k+1} = x_k - \frac{f'(x_k)}{f''(x_k)} \quad \text{(opt 1D)}$$

$$\left(x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)} \quad \text{for solving} \right)$$

$$\vec{x}_{k+1} = \vec{x}_k - H_f(x_k)^{-1} \nabla f$$

Goals

$f : \mathbb{R}^n \to \mathbb{R}$

- nonlin lsq
- interpolation

Nonlinear Least Squares: Setup

What if the f to be minimized is actually a 2-norm?

$$f(\boldsymbol{x}) = \|\boldsymbol{r}(\boldsymbol{x})\|_2, \qquad \boldsymbol{r}(\boldsymbol{x}) = \boldsymbol{y} - \boldsymbol{a}(\boldsymbol{x})$$

$$\varphi(\vec{x}) = \frac{1}{2}\vec{r}(\vec{x})^T\vec{r}(x)$$

$$\frac{\partial}{\partial x_i}\varphi = \frac{1}{2}\sum_{j=1}\frac{\partial}{\partial x_i}\left(r_j(\vec{x})\right)^2 = \sum_j\left(\frac{\partial}{\partial x_i}r_j(\vec{x})\right)r_j(\vec{x})$$

$$\nabla\varphi = J_{\vec{r}}^T(\vec{x})r(\vec{x})$$

Gauss-Newton

For brevity: $J := J_r(\boldsymbol{x})$.

$$H_\varphi(\vec{x}) = J_r^T J_r + \underbrace{\sum_i r_i H_{r_i}}_{\substack{\text{painful} \\ \text{maybe small}}}$$

$$\tilde{H}(\vec{x}) = J_r^T J_r$$

$$\vec{x}_{k+1} = \vec{x}_k - \tilde{H}^{-1} \nabla \varphi = \vec{x}_k - \underbrace{(J^T J)^{-1} J^T r}$$

equiv. to solving $J^T J \vec{s} = J^T r \rightsquigarrow J \vec{s} \approx \vec{r}$

Gauss-Newton: Observations?

Demo: Gauss-Newton [cleared]

Observations?

Gauss-Newton: Observations?

Demo: Gauss-Newton [cleared]

Observations?

- Newton on its own is still only locally convergent
- Gauss-Newton is clearly similar
- It's worse because the step is only approximate
  $\rightarrow$ Much depends on the starting guess.

Levenberg-Marquardt

If Gauss-Newton on its own is poorly conditioned, can try Levenberg-Marquardt:

$$(J^T J + \mu_k I) s_k = J^T r$$

Regularization term

lsq for:

$$\begin{pmatrix} J \\ \sqrt{\mu_k} I \end{pmatrix} s \cong \begin{pmatrix} r \\ 0 \end{pmatrix}.$$

Outline

Interpolation: Setup

Given: $(x_i)_{i=1}^N$, $(y_i)_{i=1}^N$
Wanted: Function f so that $f(x_i) = y_i$

How is this not the same as function fitting? (from least squares)

Interpolation: Setup

Given: $(x_i)_{i=1}^N$, $(y_i)_{i=1}^N$
Wanted: Function f so that $f(x_i) = y_i$

How is this not the same as function fitting? (from least squares)

It's very similar–but the key difference is that we are asking for *exact equality*, not just minimization of a residual norm.
$\rightarrow$ Better error control, error not dominated by residual

Idea: There is an *underlying function* that we are approximating from the known point values.

Error here: Distance from that underlying function

Interpolation: Setup (II)

Given: $(x_i)_{i=1}^N$, $(y_i)_{i=1}^N$
Wanted: Function f so that $f(x_i) = y_i$

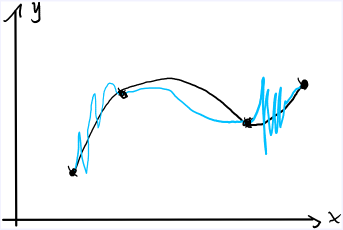
Does this problem have a unique answer?

Interpolation: Setup (II)

Given: $(x_i)_{i=1}^N$, $(y_i)_{i=1}^N$
Wanted: Function f so that $f(x_i) = y_i$

Does this problem have a unique answer?

No–there are infinitely many functions that satisfy the problem as stated:



Interpolation: Importance

Why is interpolation important?

Interpolation: Importance

Why is interpolation important?

It brings all of calculus within range of numerical operations.

- Why?
  Because calculus works on functions.
- How?
  1. Interpolate (go from discrete to continuous)
  2. Apply calculus
  3. Re-discretize (evaluate at points)

Making the Interpolation Problem Unique

Interpolation basis $\{\varphi_i\}_{i=1}^{n}$

$$p_{n-1}(x) = \sum_{i=1}^{n} \alpha_i \varphi_i(x)$$

degree of the polynomial if it is one.

data points

$$y_i = p_{n-1}(x_i) = \sum_{j=1}^{n} \alpha_j \varphi_j(x_i)$$

$(i=1\ldots n)$

generalized Vandermonde matrix

$$V = \left(\varphi_j(x_i)\right)_{i,j=1}^{n}$$

i: row, j: col

$$V = \left(\begin{array}{c} \xrightarrow{\text{functions}} \\ \downarrow \text{ data points} \end{array} \right)$$

"un-generalized" if φ_i are the monomials

$$V\vec{\alpha} = \vec{y}$$

V (coefficients) = (values at nodes)

Existence/Sensitivity

Solution to the interpolation problem: Existence? Uniqueness?

Existence / uniqueness of the linear system

Sensitivity?

sensitivity of the solve: $\kappa(V)$

Lebesgue constant:

$$\max_{x\in[a,b]} |p_{n-1}(x)| \le \Lambda \, \|y\|_\infty$$

Lebesgue const:

- depends on (x_i), (φ_i)
- same for all polynomial bases

Modes and Nodes (aka Functions and Points)

Both function basis and point set are under our control. What do we pick?

Ideas for basis functions:

- Monomials $1, x, x^2, x^3, x^4, \ldots$
- Functions that make $V = I \to$ 'Lagrange basis'
- Functions that make V triangular $\to$ 'Newton basis'
- *Splines* (piecewise polynomials)
- *Orthogonal polynomials*
- Sines and cosines
- 'Bumps' (*'Radial Basis Functions'*)

Ideas for points:

- Equispaced
- *'Edge-Clustered'* (so-called Chebyshev/Gauss/... nodes)

Specific issues:

- Why *not* monomials on equispaced points?
  Demo: Monomial interpolation [cleared]
- Why not equispaced?
  Demo: Choice of Nodes for Polynomial Interpolation [cleared]

Lagrange Interpolation

Find a basis so that $V = I$, i.e.

$$\varphi_j(x_i) = \begin{cases} 1 & i = j, \\ 0 & \text{otherwise.} \end{cases}$$

$x_1 \quad , x_2 \quad , x_3$

$$\varphi_1(x) = \frac{(x-x_2)(x-x_3)}{(x_1-x_2)(x_1-x_3)}$$

$$\varphi_2(x) = \frac{(x-x_1)(x-x_3)}{(x_2-x_1)(x_2-x_3)}$$

$$\varphi_3(x) = \frac{(x-x_1)(x-x_2)}{(x_3-x_1)(x_3-x_2)}$$

Lagrange Polynomials: General Form

$$\varphi_j(x) = \frac{\prod_{k=1, k\neq j}^{m}(x - x_k)}{\prod_{k=1, k\neq j}^{m}(x_j - x_k)}$$

Write down the Lagrange interpolant for nodes $(x_i)_{i=1}^{m}$ and values $(y_i)_{i=1}^{m}$.

$$p_{n-1}(x) = \sum_{j=1}^{n} y_j \varphi_j(x)$$

Newton Interpolation

Find a basis so that V is triangular.

Why not Lagrange/Newton?