Nonlinear least-squares, Gauss-Newton, Levenberg-Marquardt, and connections…

Tuesday, September 23, 2025 2:19 PM

Solve: $\left.\begin{array}{l} f_1(\vec{x}) = 0 \\ \vdots \\ f_m(\vec{x}) = 0 \end{array}\right\}$ m equations

i.e. $F(\vec{x}) = \vec{0}$, $F(\vec{x}) = \begin{bmatrix} f_1(\vec{x}) \\ \vdots \\ f_m(\vec{x}) \end{bmatrix}$

$F: \mathbb{R}^n \to \mathbb{R}^m$

$x \in \mathbb{R}^n$

n variables/ "unknowns"

So far we took $m=n$… what if we relax that?

$m<n$ often there are multiple solutions

$m>n$ often there's no solution } let's focus on this

No solution… next best thing is often the least-squares solution:

$$\star \quad \min_{x \in \mathbb{R}^n} \left(f(\vec{x}) := \frac{1}{2} \sum_{i=1}^m f_i(\vec{x})^2 \right) \qquad f: \mathbb{R}^n \to \mathbb{R}$$

Optimization: $\min_{x \in \mathbb{R}^n} f(\vec{x})$

canonical methods: ① gradient descent,

$$\vec{x}^{(k+1)} = \vec{x}^{(k)} - \underbrace{\eta}_{\text{scalar stepsize}} \cdot \nabla f(\vec{x}^{(k)})$$

② Newton's method (for minimization)

$$\vec{x}^{(k+1)} = \vec{x}^{(k)} - \nabla^2 f(\vec{x}^{(k)})^{-1} \cdot \nabla f(\vec{x}^{(k)})$$

Apply these to our least-squares problem

J_F or J or $J(\vec{x})$ is the Jacobian of F

$$J = \begin{bmatrix} -\nabla f_1^T- \\ \vdots \\ -\nabla f_m^T- \end{bmatrix}$$

① gradient descent,

$$\nabla f(\vec{x}) = \frac{1}{\cancel{2}} \sum_{i=1}^m \cancel{2} \cdot f_i(\vec{x}) \cdot \nabla f_i(\vec{x}) = J^T \cdot F(\vec{x}) = \begin{bmatrix} | & & | \\ \nabla f_1 & \cdots & \nabla f_m \\ | & & | \end{bmatrix} \cdot \begin{bmatrix} f_1 \\ \vdots \\ f_m \end{bmatrix}$$

So $\vec{x}^{(k+1)} = \vec{x}^{(k)} - \eta \cdot J(\vec{x}^{(k)})^T \cdot F(\vec{x}^{(k)})$

②Newton's method (for minimization)

$$\nabla^2 f(\vec{x}) = \sum_{i=1}^m \underbrace{f_i(\vec{x})}_{\text{scalar}} \cdot \underbrace{\nabla^2 f_i(\vec{x})}_{n \times n \text{ matrix}} + \underbrace{\underbrace{\nabla f_i(\vec{x})}_{n \times 1 \text{ vector}} \cdot \underbrace{\nabla f_i(\vec{x})^T}_{1 \times n \text{ vector}}}_{= J^T \cdot J} \in \mathbb{R}^{n \times n}$$

so $\vec{x}^{(k+1)} = \vec{x}^{(k)} - \nabla^2 f(\vec{x}^{(k)})^{-1} \cdot J(\vec{x}^{(k)})^T F(\vec{x}^{(k)})$

③ (new) "Gauss-Newton", $\vec{x}^{(k+1)} = \vec{x}^{(k)} - \eta \cdot (J^T J)^{-1} J^T \cdot F(\vec{x}^{(k)})$

motivation 1: do Newton (for minimization) but approximate $\nabla^2 f(\vec{x})$ with just this term!

Saves needing to find $\nabla^2 f_i$, and you already needed J anyway

motivation 2: $\vec{x}^{(k+1)} = \operatorname{argmin}_{\vec{x} \in \mathbb{R}^n} \frac{1}{2} \sum_{i=1}^m \underbrace{\left(f_i(\vec{x}^{(k)}) + \nabla f_i(\vec{x}^{(k)})^T \cdot (\vec{x} - \vec{x}^{(k)}) \right)^2}_{\text{linearize inside the square}}$

Nonlinear least-squares, p. 2

Tuesday, September 23, 2025 5:17 PM

3) Levenberg-Marquardt (not correctly described in our book)

is a **robust** version of Gauss-Newton, suitable for real problems.

Common in software for nonlinear least-squares (just don't confuse with linear least-squares methods)

Comparison

let $J = J_F(\vec{x}) \in \mathbb{R}^{m \times n}$, $F: \mathbb{R}^n \to \mathbb{R}^m$. Solve $\boxed{F(\vec{x}) = \vec{0}}$

$m \geq n$, nonlinear least squares, define $f(\vec{x}) = \frac{1}{2}\sum_{i=1}^{m} f_i(\vec{x})^2$

(1) Gradient descent

$$\vec{x} \leftarrow \vec{x} - \underset{\text{positive stepsize}}{\eta} \cdot \underbrace{J^T \cdot F(\vec{x})}_{\nabla f}$$

(2) Newton (for optimization)

$$\vec{x} \leftarrow \vec{x} - \nabla^2 f(\vec{x})^{-1} \cdot J^T F(\vec{x})$$

(3) Gauss-Newton

$$\vec{x} \leftarrow \vec{x} - (J^T J)^{-1} J^T F(\vec{x})$$

if $m \geq n$ and J has rank n then $(J^T J)^{-1} J^T =: J^{+}$ the Moore-Penrose pseudoinverse. "pinv" in Matlab / np.linalg, but better to use np.linalg.lstsq

$m = n$, directly solve $F(\vec{x}) = \vec{0}$

(a) Fixed point iteration

$$\vec{x} \leftarrow \vec{x} - \eta \cdot F(\vec{x})$$

η: positive stepsize, chosen to (hopefully) make contractive

(b). Newton's Method (for root-finding) aka Newton-Raphson

$$\vec{x} \leftarrow \vec{x} - \underbrace{J^{-1}}_{n \times n \text{ so inverse makes sense}} \cdot F(\vec{x})$$

if $m = n$ and J invertible, Gauss-Newton is Newton (root-finding)

Since $(J^T J)^{-1} J^T = J^{-1} J^{-T} J^T = J^{-1}$

...and equivalent to Newton for optimization if F is affine

Nonlinear least-squares:
under mild assumptions, a "solution" always exists, but might not be a solution to $F(\vec{x}) = 0$.

root-finding: root may not exist! i.e. equations could be incompatible / inconsistent

Also, have issues of local min vs. global min

Both approaches:

- might need to initialize close
- may have singular or ill-conditioned matrices to invert
- (2),(3), (b) scale $O(n^3)$ w/ dimension n
  (1),(a) often better in high dimensions