Quasi-Newton

Friday, September 29, 2023 2:09 PM

Newton for non-linear systems has 2 big difficulties

1- You compute an inverse at each step. if $J \in \mathbb{R}^{m \times m}$ cost of inverse is $O(m^3)$

2- You may not be able to create $J_n$. You may have to approximate it.

We will focus on problem 1.

Traditional Newton:

$$\begin{pmatrix} x_{n+1} \\ y_{n+1} \end{pmatrix} = \begin{pmatrix} x_n \\ y_n \end{pmatrix} - J_n^{-1} \begin{pmatrix} f(x_n, y_n) \\ g(x_n, y_n) \end{pmatrix}$$

Lazy Newton: compute $(J_0)^{-1}$ and use for all iterations

This avoids computing repeated inverses.

If Lazy Newton converges, the convergence is super linear. (at best)

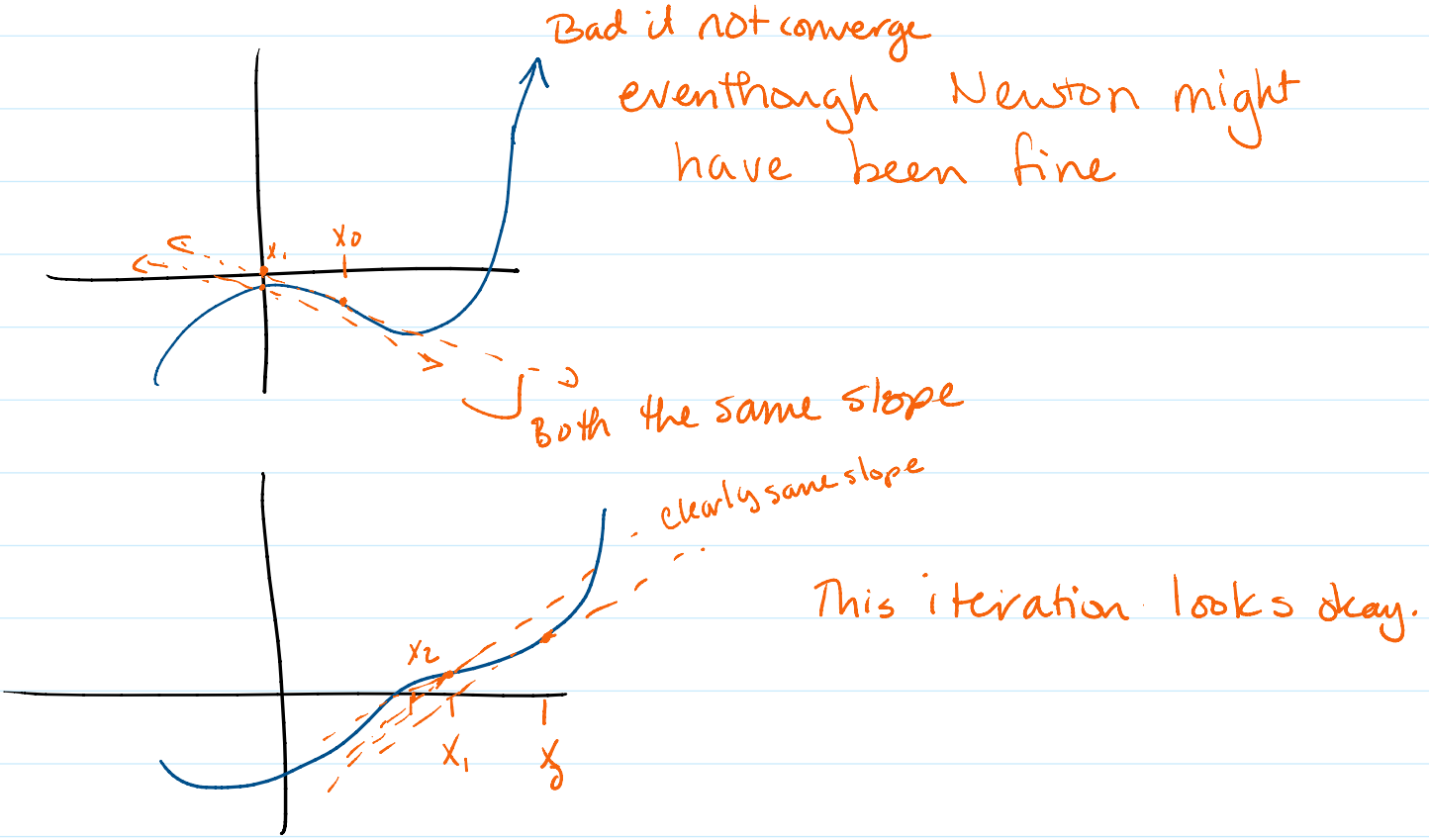
Option 1: Lazy Newton

let $\hat{J} = J(\bar{x}_0) \Rightarrow (\hat{J})^{-1} = (J(x_0))^{-1}$

& the iteration is

$$x_{n+1} = x_n - (\hat{J})^{-1} F(x_n)$$

## Graphically (1D)




**Broyden:** Rough extension of the secant method.

**IDEA:** Given $J_0 = J(x_0)$. Try to write $J(x_1) = J_1$ as a rank 1 update to $J_0$

**Note:** A rank one matrix can be expressed as an outer product of two vectors $x, y \in \mathbb{R}^{m(\times 1)}$

inner product $\quad x \cdot y = y^T x = \mathbb{R}$

outer product $\quad x y^T = \mathbb{R}^{m \times m}$

$m \times 1 \quad 1 \times m$

Our goal in Linear algebra:

Write $J_1 = J_0 + xy^T$ for some $x \& y \in \mathbb{R}^m$

Two question

1- Why is this a viable idea?

2- How do we find (create) $x \& y$?

Answer Question 1:

Thm 10.8 (Sherman-Morrison Formula)

let $A \in \mathbb{R}^{m \times m}$, & $x, y \in \mathbb{R}^m$

Given $A$, $A^{-1}$ & a rank 1 update $xy^T$

The inverse of $A + xy^T$ is given by the following

$$(A + xy^T)^{-1} = A^{-1} - \frac{A^{-1} x y^T A^{-1}}{1 + y^T A^{-1} x}$$

Why is this powerful?

- No new inverses are needed
- Cost of applying the "inverse" of $A + xy^T$ is $O(m^2)$

$$(A + xy^T)^{-1} w = A^{-1} w - \frac{A^{-1} x \, y^T A^{-1} w}{1 + y^T A^{-1} x}$$

Compute circle items once. Then only do matrix vector multiplies

Proof: (one direction here. You can do the other direction at home)

$$\left(A^{-1} - \frac{A^{-1} x y^T A^{-1}}{1 + y^T A^{-1} x}\right)(A + xy^T)$$

$$= \underbrace{A^{-1} A}_{I} - \frac{A^{-1} x y^T \overbrace{A^{-1} A}^{I}}{1 + y^T A^{-1} x} + A^{-1} x y^T$$

$$- \frac{A^{-1} x y^T A^{-1} x y^T}{1 + y^T A^{-1} x}$$

$$= I + A^{-1} x y^T - \left(A^{-1} x y^T + A^{-1} x \overbrace{y^T A^{-1} x}^{=\text{constant}} y^T\right)$$

$$= I + A^{-1}xy^T - \left(\frac{A^{-1}xy^T + A^{-1}xy^T A^{-1}xy^T}{1 + y^T A^{-1}x}\right)$$

$$= I + A^{-1}xy^T - A^{-1}xy^T \left(\frac{1 + y^T A^{-1}x}{1 + y^T A^{-1}x}\right)$$

$A^{-1}xy^T = 0$

$= 1$

$= I$

//

So the Sherman-Morrison Formula works but what are the update vectors?

We have a vector function $\bar{F}(\bar{x})$

Define our approximate Jacobian at step n+1

$$\hat{G}_{n+1} = \hat{G}_n + \underbrace{[F(x_{n+1}) - F(x_n) - \hat{G}_n(x_{n+1} - x_n)]}_{x} \underbrace{\frac{(x_{n+1} - x_n)^T}{\|x_{n+1} - x_n\|_2^2}}_{y^T}$$

$x$: Approx of "slope"

$y^T$: Direction

Numerical Example

Consider the task of approximating the root of the follow nonlinear system of equations

$$\begin{cases} 3x - \cos(yz) - \frac{1}{2} = 0 \\ x - 81(y+0.1)^2 + \sin(z) + 1.06 = 0 \\ e^{-xy} + 20z + \frac{10\pi - 3}{3} = 0 \end{cases}$$

Initial guess $x_0 = \begin{pmatrix} 0.1 \\ 0.1 \\ -0.1 \end{pmatrix}$

Code posted on canvas.