CSC338 Lecture 11

Computation Problem: Unconstrained optimization
$f: \mathbb{R} \to \mathbb{R}$, find a local minimum of $f(x)$.

① Golden Section Search: If f is unimodal on $[a,b]$ then we can iteratively shrink the interval in which x^* lies in.
only requires evaluation of f, not derivatives.
However, requires the unimodal property. Always converges.

② Newton's Method: Idea – approximate $f(x)$ using a Quadratic function, and find a critical point of the approximation.

$$x_{k+1} = x_k - \frac{f'(x)}{f''(x)}.$$

Requires derivative and Second derivative.

Does not always converge. Does not always converge to a min.

Today $f: \mathbb{R}^n \to \mathbb{R}$, find local minimum of $f(\underline{x})$.

1) Newton's method
2) Gradient Descent

Newton's method for $f: \mathbb{R}^n \to \mathbb{R}$

Recall, for $f: \mathbb{R} \to \mathbb{R}$, we have the Taylor Series Expansion

$$f(x+h) = f(x) + f'(x)h + \frac{1}{2}f''(x)h^2 + O(h^3)$$

This result extends to $f: \mathbb{R}^n \to \mathbb{R}$

$$f(\underline{x}+\underline{s}) = f(\underline{x}) + \underbrace{\nabla f(\underline{x})^T}_{1\times n}\underbrace{\underline{s}}_{n\times 1} + \frac{1}{2}\underbrace{\underline{s}^T}_{1\times n}\underbrace{H_f(\underline{x})}_{n\times n}\underbrace{\underline{s}}_{n\times 1}$$

Gradient: $\begin{bmatrix} \frac{\partial f}{\partial x_1} \\ \vdots \\ \frac{\partial f}{\partial x_n} \end{bmatrix}$ (n×1)

Hessian.

Idea: In each iteration, we have an estimate $\underline{x}_k$ of a minimum, approximate $f(\underline{x})$ with

$$f_k(\underline{x}) = f(\underline{x}_k) + \nabla f(\underline{x}_k)^T\underline{s} + \frac{1}{2}\underline{s}^T H_f(\underline{x}_k)\underline{s}$$

$(\underline{x}_k + \underline{s})$

And find $\underline{s}$ that (locally) minimizes $f_k(\underline{x}_k + \underline{s})$.

Critical points of f_k has:

$$\nabla_{\underline{s}} f_k(\underline{x}_k + \underline{s}) = 0 \quad \text{(like } f'(x)=0\text{)}$$

$$\Rightarrow \underbrace{H_f(\underline{x}_k)}_{n\times n}\underbrace{\underline{s}_k}_{n\times 1} = -\underbrace{\nabla f(\underline{x}_k)}_{n\times 1} \quad \text{(like } f''(x)h = -f(x)\text{)}$$

Newton's method update rule:

$$\underline{x}_{k+1} = \underline{x}_k + \underline{s}_k \qquad \left(\underline{s}_k = -\left(H_f(\underline{x}_k)\right)^{-1} \nabla f(x_k) \right.$$

One major disadvantage of Newton's method is that computing the Hessian $H_f(\underline{x})$ is expensive. $H_f(\underline{x}) \in \mathbb{R}^{n \times n}$

eg// $f(\underline{x}) = x_1^4 + x_1^2 x_2 + x_1^2 + 2x_2^3 + x_2$

we wish to find a min, starting with $\underline{x}_0 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$

$$\nabla f(\underline{x}) = \begin{bmatrix} 4x_1^3 + 2x_1 x_2 + 2x_1 \\ x_1^2 + 4x_2 + 1 \end{bmatrix} \qquad H_f(\underline{x}) = \begin{bmatrix} 12x_1 + 2x_2 + 2 & 2x_1 \\ 2x_1 & 4 \end{bmatrix}$$

$$\nabla f(\underline{x}_0) = \begin{bmatrix} 8 \\ 6 \end{bmatrix} \qquad H_f(\underline{x}_0) = \begin{bmatrix} 16 & 2 \\ 2 & 4 \end{bmatrix}$$

Solve $\underline{s}_0 = \begin{bmatrix} -2/5 \\ -4/5 \end{bmatrix}$

Reading contour Plots:

A curve in the contour plot shows points with the same $f(x)$ value.

Steepest descent (gradient Descent)

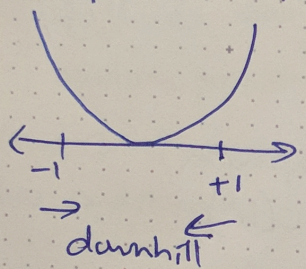
We'll discuss a variation of the algorithm from your textbook. This version is typically used in machine learning.

Key Idea: ~~the negative gradient~~

The gradient of a differentiable function points uphill
∴ negative gradient ∴ ∴ ∴ downhill

Toward $\underline{x}$ with lower values of $f(\underline{x})$.

Example in 1D: $f(x) = x^2 : \mathbb{R} \to \mathbb{R}$



$f'(-1) = 2(-1) = -2$

⇒ negative grad points to (2) to the right

$f'(1) = 2(1) = 2$

⇒ negative gradient points to the left

Example in 2D: $f(\underline{x}) = x_1^4 + x_1x_2 + x_1^2 + 2x_2^2 + x_2$

At $\underline{x} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$ $\nabla f(\underline{x}) = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$ So downhill is in the direction $\begin{bmatrix} 0 \\ -1 \end{bmatrix}$

$\underline{x} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$ $\nabla f(\underline{x}) = \begin{bmatrix} 8 \\ 6 \end{bmatrix}$ So ∴ ∴ ∴ ∴ ∴ $\begin{bmatrix} -8 \\ -6 \end{bmatrix}$

Note: The gradient is always perpendicular to the contour!

Why does this work?

Using Taylor's Thm, we can locally approximate f using

$$f(\underline{x} + \underline{s}) = f(\underline{x}) + \nabla f(\underline{x})^T \underline{s}$$

If ~~$\nabla f(\underline{x}) = 0$~~ $\nabla f(\underline{x}) \neq 0$, then if we let $\underline{s} = -\nabla f(\underline{x}) \cdot c$ ← for some c small

move $\underline{x}$ towards $-\nabla f(\underline{x})$

Then $f(\underline{x} + \underline{s}) = f(x) + ($Something negative$)$

It turns out that $-\nabla f(\underline{x})$ is, locally, the direction of the steepest descent.

Steepest Descent for finding a minima of $f: \mathbb{R}^n \to \mathbb{R}$

Start from initial guess $\underline{x}_0$, and update

$$\underline{x}_{k+1} = \underline{x}_k - \alpha_k \nabla f(\underline{x}_k) \qquad \alpha_k \in \mathbb{R}, \ \alpha_k \geq 0$$

- How to find (or choose) α_k?

  Textbook: use a line search: find α_k ~~so that~~ to minimize

  $$g(\alpha_k) = f(\underline{x}_k - \alpha_k \nabla f(\underline{x}_k)) \quad : \mathbb{R} \to \mathbb{R}.$$

  Gradient Descent: Set α_k to a constant α called the learning rate

Steepest descent is usually reliable: can always make progress provided that the gradient is nonzero.

However this method is myopic in its view of function's behavior

↑ only take into account local information

eg $f(\underline{x}) = 0.5 x_1^2 + 2.5 x_2^2$ $\quad \nabla f(\underline{x}) = \begin{bmatrix} x_1 \\ 5x_2 \end{bmatrix}$

... slow slide 29 & 30.

Comparison:

Steepest Descent: $\underline{x}_{k+1} = \underline{x}_k - \alpha_k \nabla f(\underline{x}_k)$ ← like assuming $H_f(\underline{x}_k) = \alpha_k I$.

Newton: $\underline{x}_{k+1} = \underline{x}_k - H_f(\underline{x}_k)^{-1} \nabla f(\underline{x}_k)$

Quasi-Newton: $\underline{x}_{k+1} = \underline{x}_k - \alpha_k B_k^{-1} \nabla f(\underline{x}_k)$

where B_k is an approximation of the Hessian matrix. (e.g. like secant method)

$\Rightarrow$ BFGS.

Revisit hw grade prediction: predict HW3 given HW1, HW2 grades.

$$A = \begin{bmatrix} a_1^{(1)} & a_2^{(1)} \\ a_1^{(2)} & a_2^{(2)} \\ \vdots & \\ a_1^{(73)} & a_2^{(73)} \end{bmatrix} \quad \underline{b} = \begin{bmatrix} b^{(1)} \\ b^{(2)} \\ \vdots \\ b^{(73)} \end{bmatrix}$$

↑ colum = one hw ← row = one student

Problem find $\underline{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$ to minimize $\|A\underline{x} - \underline{b}\|_2$

We can treat this as a non-linear optimization problem:

(see slides for rest of problem) ~~(see next 4 to pages)~~