COMP 330: Optimization–Newton's Method

Chris Jermaine and Kia Teymourian

Rice University

Alternatives to Gradient Descent

Gradient descent great

- Easy to use
- Widely applicable
- But convergence can be slow

Can we do better? Sure!

Second-Order Methods

Class of iterative optimization methods

- Use not only first partial derivatives
- But second as well
- Speeds convergence
- Cost: more complexity
- Cost: quadratic in number of variables

Newton's Method

Classic second order method for optimization

- Comes from Newton's method for finding zero of a function $F()$
- How does it work?

$\theta \leftarrow$ `intial guess;`
`while` θ `keeps changing,` **`do:`**
  θ `\leftarrow` θ `- \frac{F(\theta)}{F'(\theta)};`

What About for Optimization?

In data science, don't want a zero

- Want a max/min
- So just find the root (zero) of the derivative

Algorithm becomes:

$\theta \leftarrow$ `intial guess;`
`while` θ `keeps changing,` **`do`**`:`
  $\theta \leftarrow \theta - \frac{F'(\theta)}{F''(\theta)}$`;`

Multi-Variate Newton's Method

- Say we have:

$$F_1(\theta_1, \theta_2, \ldots, \theta_m)$$
$$F_2(\theta_1, \theta_2, \ldots, \theta_m)$$
$$\ldots$$
$$F_m(\theta_1, \theta_2, \ldots, \theta_m)$$

- We want $\Theta = \langle \theta_1, \theta_2, \ldots, \theta_m \rangle$ such that:

$$F_1(\theta_1, \theta_2, \ldots, \theta_m) = 0$$
$$F_2(\theta_1, \theta_2, \ldots, \theta_m) = 0$$
$$\ldots$$
$$F_m(\theta_1, \theta_2, \ldots, \theta_m) = 0$$

- How to do this?

Multi-Variate Newton's Method

Turns out it's not so difficult...

- Won't do the derivation (relies on multi-variate Taylor expansion)
- Define $F(\Theta)$ to be the vector:

$$\langle F_1(\theta_1, \theta_2, \ldots, \theta_m), F_2(\theta_1, \theta_2, \ldots, \theta_m), \ldots, F_m(\theta_1, \theta_2, \ldots, \theta_m)\rangle$$

- Define the "Jacobian" of $F_1, F_2, \ldots, F_m$ to be:

$$J_F = \begin{pmatrix} \frac{\partial F_1}{\partial \theta_1} & \frac{\partial F_1}{\partial \theta_2} & \frac{\partial F_1}{\partial \theta_3} & \cdots \\ \frac{\partial F_2}{\partial \theta_1} & \frac{\partial F_2}{\partial \theta_2} & \frac{\partial F_2}{\partial \theta_3} & \cdots \\ \frac{\partial F_3}{\partial \theta_1} & \frac{\partial F_3}{\partial \theta_2} & \frac{\partial F_3}{\partial \theta_3} & \cdots \\ \cdots & \cdots & \cdots & \cdots \end{pmatrix}$$

- Note: this is a matrix of functions!
- So $J_F(\Theta)$ is a matrix of scalars

Multi-Variate Newton's Method

Multi-Variate Newton's is simply:

$\Theta \leftarrow$ `intial guess;`
`while` Θ `keeps changing,` **`do:`**
  $\Theta \leftarrow \Theta - J_F^{-1}(\Theta)F(\Theta)$`;`

What About Multi-Variate Optimization

Difference: we don't have a system of equations to solve

- Just have $F()$, which we want to max
- That is, want Θ such that:

$$\frac{\partial F}{\partial \theta_1}(\Theta) = 0$$
$$\frac{\partial F}{\partial \theta_2}(\Theta) = 0$$
$$\cdots$$
$$\frac{\partial F}{\partial \theta_m}(\Theta) = 0$$

- So just let F_1 be $\frac{\partial F}{\partial \theta_1}$, F_2 be $\frac{\partial F}{\partial \theta_1}$, etc.
- That is, we want Θ such that $\nabla F(\Theta) = 0$
- Can then use exactly the same alg as before!

What About Multi-Variate Optimization

Then this:

$\Theta \leftarrow$ `intial guess;`
`while` Θ `keeps changing,` **`do`**`:`
  $\Theta \leftarrow \Theta - J_F^{-1}(\Theta)F(\Theta)$`;`

Becomes this:

$\Theta \leftarrow$ `intial guess;`
`while` Θ `keeps changing,` **`do`**`:`
  $\Theta \leftarrow \Theta - J_{\nabla F}^{-1}(\Theta)\nabla F(\Theta)$`;`

Thet's it!

One Last Thing

We have:

$\Theta \leftarrow$ `intial guess;`
while Θ `keeps changing,` **do**:
  $\Theta \leftarrow \Theta - J_{\nabla F}^{-1}(\Theta)\nabla F(\Theta)$;

- The matrix of functions $J_{\nabla F}$ is typically called the "Hessian" of F
- Entries are:

$$H_F = \begin{pmatrix} \frac{\partial F}{\partial \theta_1^2} & \frac{\partial F}{\partial \theta_1 \partial \theta_2} & \frac{\partial F}{\partial \theta_1 \partial \theta_3} & \cdots \\ \frac{\partial F}{\partial \theta_1 \partial \theta_2} & \frac{\partial F}{\partial \theta_2^2} & \frac{\partial F}{\partial \theta_2 \partial \theta_3} & \cdots \\ \frac{\partial F}{\partial \theta_1 \partial \theta_3} & \frac{\partial F}{\partial \theta_2 \partial \theta_3} & \frac{\partial F}{\partial \theta_3^2} & \cdots \\ \cdots & \cdots & \cdots & \cdots \end{pmatrix}$$

Pros and Cons of Newton's

Pro: convergence is quadratic; that is, error decreases quadratically

Pro: hundreds/thousands of iters (gradient descent) becomes tens

Con: more complicated than gradient descent!

Con: quatratic cost each iter (linear gradient descent)

Questions?