Quasi-Newton Optimization

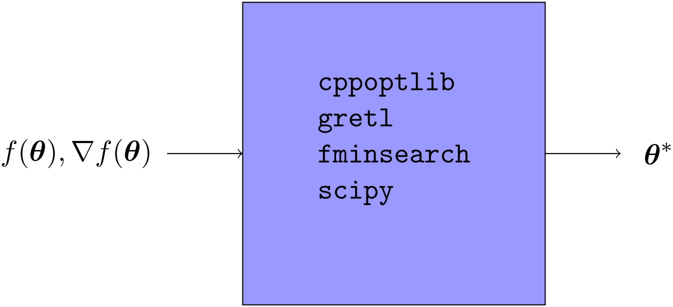
The Lifeblood of Model Selection

John Martin Jr.

August 18, 2017

The Optimization Black Box

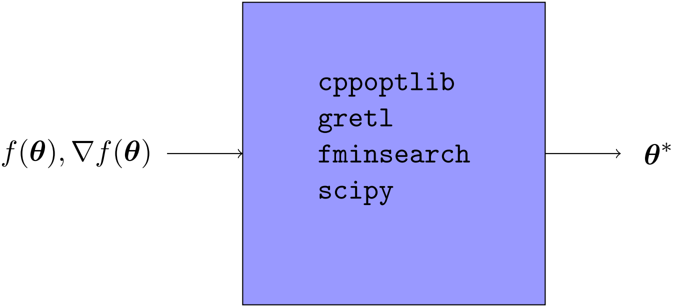
$$\boldsymbol{\theta}^* = \arg\min f(\boldsymbol{\theta})$$



What happens inside?

The Optimization Black Box

$$\boldsymbol{\theta}^* = \arg\min f(\boldsymbol{\theta})$$



What happens inside?

- Math?, Magic?

Optimization in Machine Learning

Objective Functions

- Marginal likelihood $f(\boldsymbol{\theta}) = p(\mathbf{y}|\mathbf{X}, \boldsymbol{\theta})$
- Entropy $f(\boldsymbol{\theta}) = \mathbf{E}[\log p(\mathbf{y}|\boldsymbol{\theta})]$
- Cross Entropy $f(\boldsymbol{\theta}) = \textsc{kl}[p(\mathbf{y}|\boldsymbol{\theta})||q(\boldsymbol{\theta})]$

Optimization in Machine Learning

Objective Functions

- Marginal likelihood $f(\boldsymbol{\theta}) = p(\mathbf{y}|\mathbf{X}, \boldsymbol{\theta})$
- Entropy $f(\boldsymbol{\theta}) = \mathbf{E}[\log p(\mathbf{y}|\boldsymbol{\theta})]$
- Cross Entropy $f(\boldsymbol{\theta}) = \textsc{kl}[p(\mathbf{y}|\boldsymbol{\theta})||q(\boldsymbol{\theta})]$

Non-Gradient Methods

- Analytical methods
- Nelder-Mead
- Convex programs

Optimization in Machine Learning

Objective Functions

- Marginal likelihood $f(\boldsymbol{\theta}) = p(\mathbf{y}|\mathbf{X}, \boldsymbol{\theta})$
- Entropy $f(\boldsymbol{\theta}) = \mathbf{E}[\log p(\mathbf{y}|\boldsymbol{\theta})]$
- Cross Entropy $f(\boldsymbol{\theta}) = \textsc{kl}[p(\mathbf{y}|\boldsymbol{\theta})||q(\boldsymbol{\theta})]$

Non-Gradient Methods

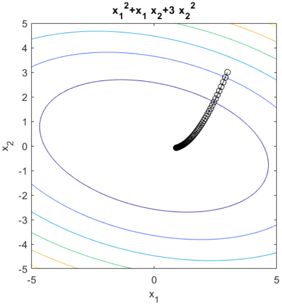
- Analytical methods
- Nelder-Mead
- Convex programs

Gradient Methods

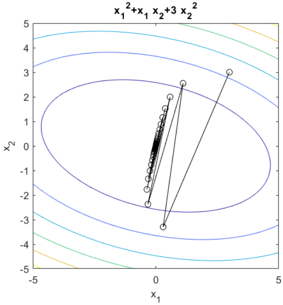
- Online, Batch (i.e. offline)
- First-order, Newton, Quasi-Newton

The Trouble with Gradients

$$\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k - \eta \nabla f(\boldsymbol{\theta}_k)$$



- $\eta = 0.001$
- Too slow



- $\eta = 0.03$
- Unstable

The Benefit of Hessians

Suppose the objective is quadratic

$$f(\boldsymbol{\theta}) = f(\boldsymbol{\theta}_k) + \nabla f(\boldsymbol{\theta}_k)^\top (\boldsymbol{\theta} - \boldsymbol{\theta}_k) + \frac{1}{2}(\boldsymbol{\theta} - \boldsymbol{\theta}_k)^\top \mathbf{H}_k (\boldsymbol{\theta} - \boldsymbol{\theta}_k).$$

The Benefit of Hessians

Suppose the objective is quadratic

$$f(\boldsymbol{\theta}) = f(\boldsymbol{\theta}_k) + \nabla f(\boldsymbol{\theta}_k)^\top(\boldsymbol{\theta} - \boldsymbol{\theta}_k) + \frac{1}{2}(\boldsymbol{\theta} - \boldsymbol{\theta}_k)^\top \mathbf{H}_k(\boldsymbol{\theta} - \boldsymbol{\theta}_k).$$

Differentiate f w.r.t $\boldsymbol{\theta}$, equate to zero, and solve for $\boldsymbol{\theta}$

$$\nabla f(\boldsymbol{\theta}) = \nabla f(\boldsymbol{\theta}_k) + \mathbf{H}_k(\boldsymbol{\theta} - \boldsymbol{\theta}_k),$$
$$\boldsymbol{\theta} = \boldsymbol{\theta}_k - \mathbf{H}_k^{-1}\nabla f(\boldsymbol{\theta}_k).$$

The Benefit of Hessians

Suppose the objective is quadratic

$$f(\boldsymbol{\theta}) = f(\boldsymbol{\theta}_k) + \nabla f(\boldsymbol{\theta}_k)^\top (\boldsymbol{\theta} - \boldsymbol{\theta}_k) + \frac{1}{2}(\boldsymbol{\theta} - \boldsymbol{\theta}_k)^\top \mathbf{H}_k (\boldsymbol{\theta} - \boldsymbol{\theta}_k).$$

Differentiate f w.r.t $\boldsymbol{\theta}$, equate to zero, and solve for $\boldsymbol{\theta}$

$$\nabla f(\boldsymbol{\theta}) = \nabla f(\boldsymbol{\theta}_k) + \mathbf{H}_k(\boldsymbol{\theta} - \boldsymbol{\theta}_k),$$
$$\boldsymbol{\theta} = \boldsymbol{\theta}_k - \mathbf{H}_k^{-1} \nabla f(\boldsymbol{\theta}_k).$$

Update step is $\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k + \mathbf{d}_k$.

The Benefit of Hessians

Suppose the objective is quadratic

$$f(\boldsymbol{\theta}) = f(\boldsymbol{\theta}_k) + \nabla f(\boldsymbol{\theta}_k)^\top (\boldsymbol{\theta} - \boldsymbol{\theta}_k) + \frac{1}{2}(\boldsymbol{\theta} - \boldsymbol{\theta}_k)^\top \mathbf{H}_k (\boldsymbol{\theta} - \boldsymbol{\theta}_k).$$

Differentiate f w.r.t $\boldsymbol{\theta}$, equate to zero, and solve for $\boldsymbol{\theta}$

$$\begin{aligned} \nabla f(\boldsymbol{\theta}) &= \nabla f(\boldsymbol{\theta}_k) + \mathbf{H}_k(\boldsymbol{\theta} - \boldsymbol{\theta}_k), \\ \boldsymbol{\theta} &= \boldsymbol{\theta}_k - \mathbf{H}_k^{-1} \nabla f(\boldsymbol{\theta}_k). \end{aligned}$$

Update step is $\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k + \mathbf{d}_k$.

Variants

- *Newton methods* use curvature as the step size $\eta_k = \mathbf{H}_k^{-1}$
- *Conjugate Gradient* methods solve $\mathbf{H}_k \mathbf{d}_k = -\nabla f(\boldsymbol{\theta}_k)$

Quasi-Newton Methods

Features

- Build a locally quadratic approximation to the objective
- Use the gradient to estimate the Hessian
- Never invert the Hessian directly
- Maintain and approximate the inverse Hessian $\mathbf{B}$
- Use line search to regularize approximation

Quasi-Newton Methods

Features

- Build a locally quadratic approximation to the objective
- Use the gradient to estimate the Hessian
- Never invert the Hessian directly
- Maintain and approximate the inverse Hessian $\mathbf{B}$
- Use line search to regularize approximation

Quasi-Newton Optimization

---

for $k = 1$ to convergence **do**
    $\mathbf{g}_k \leftarrow \nabla f(\boldsymbol{\theta}_k)$
    $\mathbf{d}_k \leftarrow -\mathbf{B}_k \mathbf{g}_k$
    Use line search for η_k
    $\boldsymbol{\theta}_{k+1} \leftarrow \boldsymbol{\theta}_k + \eta_k \mathbf{d}_k$
    Update $\mathbf{B}_{k+1}$

Broyden Fletcher Goldfarb and Shanno

Broyden[1], Fletcher [2], Goldfarb [3], Shanno [4]

BFGS Algorithm

- $\mathbf{B}_{k+1} = \arg\min ||\mathbf{B} - \mathbf{B}_k||_{\mathbf{w}}$ st.

$$\boldsymbol{\theta}_{k+1} - \boldsymbol{\theta}_k = \mathbf{B}_k(\mathbf{g}_{k+1} - \mathbf{g}_k)$$

- Update $\mathbf{B}$ recursively with Sherman-Morrison

BFGS Algorithm

- $\mathbf{B}_{k+1} = \arg\min ||\mathbf{B} - \mathbf{B}_k||_{\mathbf{w}}$ st.

$$\boldsymbol{\theta}_{k+1} - \boldsymbol{\theta}_k = \mathbf{B}_k(\mathbf{g}_{k+1} - \mathbf{g}_k)$$

- Update $\mathbf{B}$ recursively with Sherman-Morrison

BFGS

---

initialize $\mathbf{B}_1 = \mathbf{I}$
for $k = 1$ to convergence **do**
    $\mathbf{g}_k \leftarrow \nabla f(\boldsymbol{\theta}_k)$
    $\mathbf{d}_k \leftarrow -\mathbf{B}_k \mathbf{g}_k$
    Use line search for η_k
    $\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k + \eta_k \mathbf{d}_k$
    $\mathbf{y}_k = \mathbf{g}_{k+1} - \mathbf{g}_k$
    $\mathbf{s}_k = \boldsymbol{\theta}_{k+1} - \boldsymbol{\theta}_k$
    $\mathbf{B}_{k+1} = \left(\mathbf{I} - \frac{\mathbf{s}_k \mathbf{y}_k^\top}{\mathbf{s}_k^\top \mathbf{y}_k}\right) \mathbf{B}_k \left(\mathbf{I} - \frac{\mathbf{y}_k \mathbf{s}_k^\top}{\mathbf{s}_k^\top \mathbf{y}_k}\right) + \frac{\mathbf{s}_k \mathbf{s}_k^\top}{\mathbf{s}_k^\top \mathbf{y}_k}$

BFGS Algorithm Cont.

Line Search

- $\mathbf{H}_k$ is not the true Hessian, so look ahead for validity

BFGS Algorithm Cont.

Line Search

- $\mathbf{H}_k$ is not the true Hessian, so look ahead for validity

Wolfe Conditions

- Don't take too large a step: $f(\boldsymbol{\theta}_{k+1}) \leq f(\boldsymbol{\theta}_k) + c_1 \eta_k \mathbf{g}_k^\top \mathbf{d}_k$
- Don't take too small a step: $\mathbf{g}_{k+1}^\top \mathbf{d}_k \geq c_2 \mathbf{g}_k^\top \mathbf{d}_k$
- Where $0 < c_1 < c_2 < 1$

BFGS Algorithm Cont.

Line Search

- $\mathbf{H}_k$ is not the true Hessian, so look ahead for validity

Wolfe Conditions

- Don't take too large a step: $f(\boldsymbol{\theta}_{k+1}) \leq f(\boldsymbol{\theta}_k) + c_1 \eta_k \mathbf{g}_k^\top \mathbf{d}_k$
- Don't take too small a step: $\mathbf{g}_{k+1}^\top \mathbf{d}_k \geq c_2 \mathbf{g}_k^\top \mathbf{d}_k$
- Where $0 < c_1 < c_2 < 1$

Remarks

- If $\mathbf{B}$ is not truly convex, then it is rank deficient
- L-BFGS uses low-rank approximation of $\mathbf{B}$

References I

[1] Charles Broyden.
The convergence of a class of double-rank minimization algorithms 1. general considerations.
IMA Journal of Applied Mathematics, 6(1):76–90, 1970.

[2] Roger Fletcher.
A new approach to variable metric algorithms.
The Computer Journal, 13(3):317–322, 1970.

[3] Donald Goldfarb.
A family of variable-metric methods derived by variational means.
Mathematics of computation, 24(109):23–26, 1970.

[4] David Shanno.
Conditioning of quasi-newton methods for function minimization.
Mathematics of Computation, 24(111):647–656, 1970.