2. Quasi-Newton methods

• variable metric methods

• quasi-Newton methods

• BFGS update

• limited-memory quasi-Newton methods
Newton method for unconstrained minimization

\[
\text{minimize } f(x)
\]

\(f\) convex, twice contiuously differentiable

\textbf{Newton method}

\[
x^+ = x - t \nabla^2 f(x)^{-1} \nabla f(x)
\]

- advantages: fast convergence, affine invariance
- disadvantages: requires second derivatives, solution of linear equation

can be too expensive for large scale applications
Variable metric methods

\[ x^+ = x - tH^{-1}\nabla f(x) \]

\( H \succ 0 \) is approximation of the Hessian at \( x \), chosen to:

- avoid calculation of second derivatives
- simplify computation of search direction

'Variable metric' interpretation (EE236B, lecture 10, page 11)

\[ \Delta x = -H^{-1}\nabla f(x) \]

is steepest descent direction at \( x \) for quadratic norm

\[ \|z\|_H = (z^T H z)^{1/2} \]
Quasi-Newton methods

**given** starting point $x^{(0)} \in \text{dom } f$, $H_0 \succ 0$

1. compute quasi-Newton direction $\Delta x = -H_{k-1}^{-1} \nabla f(x^{(k-1)})$

2. determine step size $t$ (e.g., by backtracking line search)

3. compute $x^{(k)} = x^{(k-1)} + t\Delta x$

4. compute $H_k$

- different methods use different rules for updating $H$ in step 4
- can also propagate $H_k^{-1}$ to simplify calculation of $\Delta x$
Broyden-Fletcher-Goldfarb-Shanno (BFGS) update

**BFGS update**

\[ H_k = H_{k-1} + \frac{yy^T}{y^Ts} - \frac{H_{k-1}ss^TH_{k-1}}{s^TH_{k-1}s} \]

where

\[ s = x^{(k)} - x^{(k-1)}, \quad y = \nabla f(x^{(k)}) - \nabla f(x^{(k-1)}) \]

**Inverse update**

\[ H_{k-1}^{-1} = \left( I - \frac{sy^T}{y^Ts} \right) H_{k-1}^{-1} \left( I - \frac{ys^T}{y^Ts} \right) + \frac{ss^T}{y^Ts} \]

- note that \( y^Ts > 0 \) for strictly convex \( f \); see page 1-9
- cost of update or inverse update is \( O(n^2) \) operations
Positive definiteness

if $y^T s > 0$, BFGS update preserves positive definiteness of $H_k$

**Proof:** from inverse update formula,

$$v^T H_k^{-1} v = \left( v - \frac{s^T v}{s^T y} \right)^T H_k^{-1} \left( v - \frac{s^T v}{s^T y} \right) + \frac{(s^T v)^2}{y^T s}$$

• if $H_{k-1} \succ 0$, both terms are nonnegative for all $v$

• second term is zero only if $s^T v = 0$; then first term is zero only if $v = 0$

this ensures that $\Delta x = -H_k^{-1} \nabla f(x^k)$ is a descent direction
Secant condition

the BFGS update satisfies the secant condition $H_k s = y$, i.e.,

$$H_k(x^{(k)} - x^{(k-1)}) = \nabla f(x^{(k)}) - \nabla f(x^{(k-1)})$$

**Interpretation:** define second-order approximation at $x^{(k)}$

$$f_{\text{quad}}(z) = f(x^{(k)}) + \nabla f(x^{(k)})^T (z - x^{(k)}) + \frac{1}{2}(z - x^{(k)})^T H_k (z - x^{(k)})$$

secant condition implies that gradient of $f_{\text{quad}}$ agrees with $f$ at $x^{(k-1)}$:

$$\nabla f_{\text{quad}}(x^{(k-1)}) = \nabla f(x^{(k)}) + H_k (x^{(k-1)} - x^{(k)})$$

$$= \nabla f(x^{(k-1)})$$
Secant method

for $f : \mathbb{R} \to \mathbb{R}$, BFGS with unit step size gives the secant method

$$x^{(k+1)} = x^{(k)} - \frac{f'(x^{(k)})}{H_k}, \quad H_k = \frac{f'(x^{(k)}) - f'(x^{(k-1)})}{x^{(k)} - x^{(k-1)}}$$

Quasi-Newton methods
Convergence

Global result

if $f$ is strongly convex, BFGS with backtracking line search (EE236B, lecture 10-6) converges from any $x^{(0)}$, $H_0 \succ 0$

Local convergence

if $f$ is strongly convex and $\nabla^2 f(x)$ is Lipschitz continuous, local convergence is superlinear: for sufficiently large $k$,

$$\|x^{(k+1)} - x^*\|_2 \leq c_k \|x^{(k)} - x^*\|_2 \to 0$$

where $c_k \to 0$

(cf., quadratic local convergence of Newton method)
Example

\[
\text{minimize} \quad c^T x - \sum_{i=1}^{m} \log(b_i - a_i^T x)
\]

\(n = 100, m = 500\)

- cost per Newton iteration: \(O(n^3)\) plus computing \(\nabla^2 f(x)\)
- cost per BFGS iteration: \(O(n^2)\)

Quasi-Newton methods
Square root BFGS update

to improve numerical stability, propagate $H_k$ in factored form $H_k = L_k L_k^T$

- if $H_{k-1} = L_{k-1} L_{k-1}^T$ then $H_k = L_k L_k^T$ with

\[ L_k = L_{k-1} \left( I + \frac{\alpha \tilde{y} - \tilde{s}}{\tilde{s}^T \tilde{s}} \right) \]

where

\[ \tilde{y} = L_{k-1}^{-1} y, \quad \tilde{s} = L_{k-1} s, \quad \alpha = \left( \frac{\tilde{s}^T \tilde{s}}{y^T s} \right)^{1/2} \]

- if $L_{k-1}$ is triangular, cost of reducing $L_k$ to triangular form is $O(n^2)$
Optimality of BFGS update

\[ X = H_k \] solves the convex optimization problem

\[
\begin{align*}
\text{minimize} & \quad \text{tr}(H_{k-1}^{-1}X) - \log \det(H_{k-1}^{-1}X) - n \\
\text{subject to} & \quad Xs = y
\end{align*}
\]

- cost function is nonnegative, equal to zero only if \( X = H_{k-1} \)
- also known as relative entropy between densities \( N(0, X), N(0, H_{k-1}) \)

optimality result follows from KKT conditions: \( X = H_k \) satisfies

\[
X^{-1} = H_{k-1}^{-1} - \frac{1}{2}(sv^T + \nu s^T), \quad Xs = y, \quad X \succ 0
\]

with

\[
\nu = \frac{1}{s^Ty} \left( 2H_{k-1}^{-1}y - \left( 1 + \frac{y^TH_{k-1}^{-1}y}{y^Ts} \right)s \right)
\]
Davidon-Fletcher-Powell (DFP) update

switch $H_{k-1}$ and $X$ in objective on previous page

\[
\begin{align*}
\text{minimize} & \quad \text{tr}(H_{k-1}X^{-1}) - \log \det(H_{k-1}X^{-1}) - n \\
\text{subject to} & \quad Xs = y
\end{align*}
\]

- minimize relative entropy between $N(0, H_{k-1})$ and $N(0, X)$
- problem is convex in $X^{-1}$ (with constraint written as $s = X^{-1}y$)
- solution is ‘dual’ of BFGS formula

\[
H_k = \left( I - \frac{ys^T}{s^Ty} \right) H_{k-1} \left( I - \frac{sy^T}{s^Ty} \right) + \frac{yy^T}{s^Ty}
\]

(known as DFP update)

predates BFGS update, but is less often used
Limited memory quasi-Newton methods

main disadvantage of quasi-Newton method is need to store $H_k$ or $H_k^{-1}$

**Limited-memory BFGS (L-BFGS):** do not store $H_k^{-1}$ explicitly

- instead we store the $m$ (e.g., $m = 30$) most recent values of

$$s_j = x^{(j)} - x^{(j-1)}, \quad y_j = \nabla f(x^{(j)}) - \nabla f(x^{(j-1)})$$

- we evaluate $\Delta x = H_k^{-1} \nabla f(x^{(k)})$ recursively, using

$$H_j^{-1} = \left( I - \frac{s_j y_j^T}{y_j^T s_j} \right) H_{j-1}^{-1} \left( I - \frac{y_j s_j^T}{y_j^T s_j} \right) + \frac{s_j s_j^T}{y_j^T s_j}$$

for $j = k, k - 1, \ldots, k - m + 1$, assuming, for example, $H_{k-m}^{-1} = I$

- cost per iteration is $O(nm)$; storage is $O(nm)$
References
