Quasi-Newton methods

- variable metric methods
- quasi-Newton methods
- BFGS update
- limited-memory quasi-Newton methods

Newton method for unconstrained minimization

$$\text{minimize} \quad f(x)$$

f convex, twice continously differentiable

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

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

is steepest descent direction at x for quadratic norm

$$\|z\|_H = \left(z^T H z\right)^{1/2}$$

Quasi-Newton methods

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

for $k = 1, 2, \ldots$, until a stopping criterion is satisfied

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 Δx

Broyden-Fletcher-Goldfarb-Shanno (BFGS) update

BFGS update

$$H_k = H_{k-1} + \frac{yy^T}{y^T s} - \frac{H_{k-1} s s^T H_{k-1}}{s^T H_{k-1} s}$$

where

$$s = x^{(k)} - x^{(k-1)}, \qquad y = \nabla f(x^{(k)}) - \nabla f(x^{(k-1)})$$

inverse update

$$H_k^{-1} = \left(I - \frac{sy^T}{y^T s}\right) H_{k-1}^{-1} \left(I - \frac{ys^T}{y^T s}\right) + \frac{ss^T}{y^T s}$$

- note that $y^T s > 0$ for strictly convex f;
- cost of update or inverse update is $O(n^2)$ operations

Positive definiteness

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

proof: from inverse update formula,

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

- 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

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 $x^{(k)}$

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

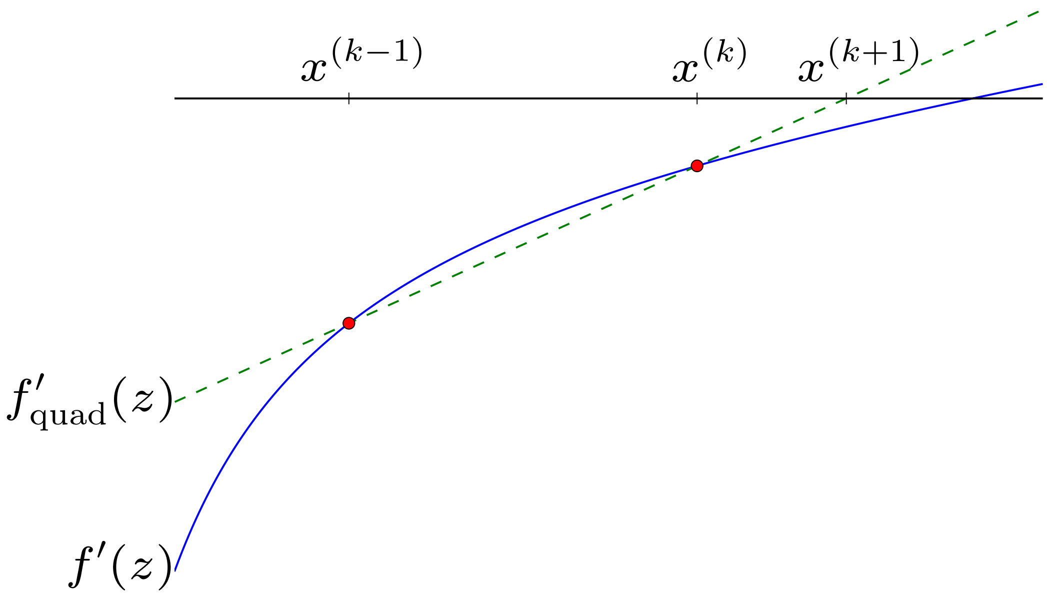
secant condition implies that gradient of f_{quad} agrees with f at $x^{(k-1)}$:

$$\begin{aligned} \nabla f_{\mathrm{quad}}(x^{(k-1)}) &= \nabla f(x^{(k)}) + H_k(x^{(k-1)} - x^{(k)}) \\ &= \nabla f(x^{(k-1)}) \end{aligned}$$

secant method

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

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



Convergence

global result

if f is strongly convex, BFGS with backtracking line search 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,

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

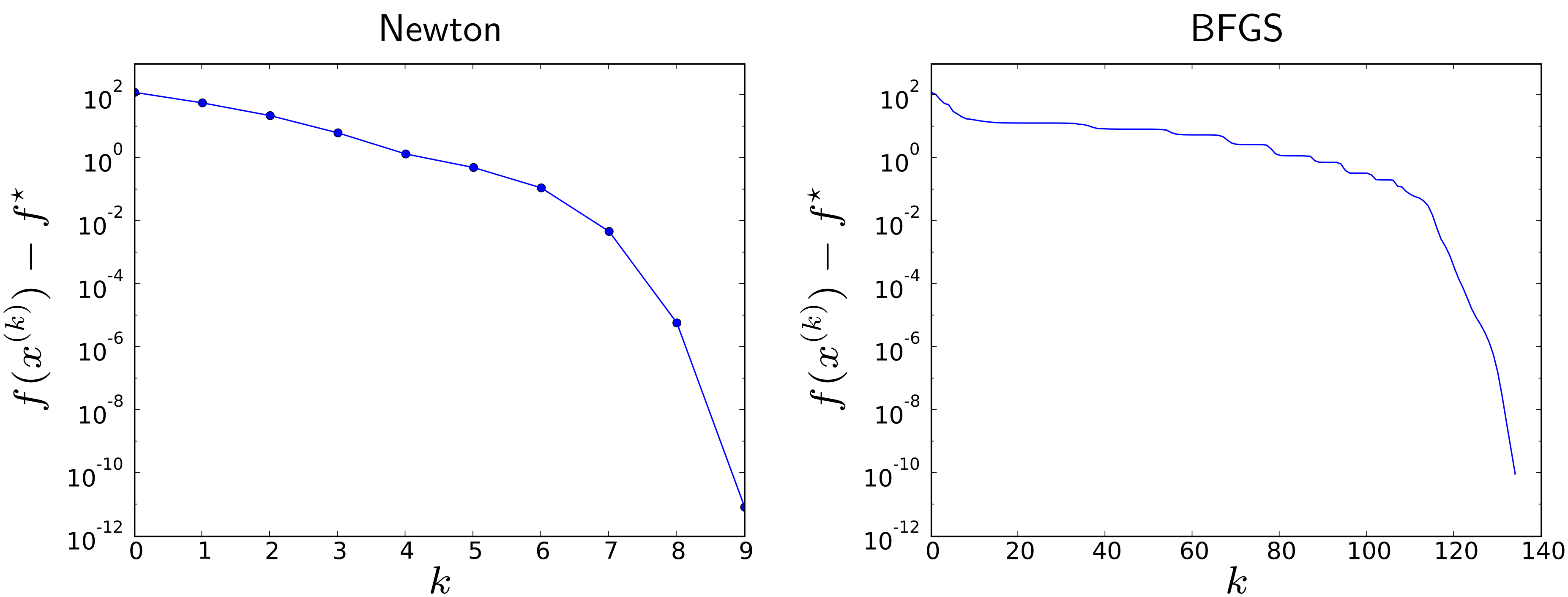
where $c_k \to 0$

($cf.$, 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: $O(n^2)$

Square root BFGS update

to improve numerical stability, can propagate H_k in factored form

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

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

where

$$\tilde{y} = L_{k-1}^{-1}y, \qquad \tilde{s} = L_{k-1}s, \qquad \alpha = \left(\frac{\tilde{s}^T\tilde{s}}{y^Ts}\right)^{1/2}$$

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

Optimality of BFGS update

$X = H_k$ solves the convex optimization problem

$$\begin{array}{ll} \text{minimize} & \mathbf{tr}(H_{k-1}^{-1}X) - \log\det(H_{k-1}^{-1}X) - n \\ \text{subject to} & Xs = y \end{array}$$

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

optimality result follows from KKT conditions: $X = H_k$ satisfies

$$X^{-1} = H_{k-1}^{-1} - \frac{1}{2}(s\nu^T + \nu s^T), \qquad Xs = y, \qquad X \succ 0$$

with

$$\nu = \frac{1}{s^T y}\left(2H_{k-1}^{-1}y - \left(1 + \frac{y^T H_{k-1}^{-1} y}{y^T s}\right) s\right)$$

Davidon-Fletcher-Powell (DFP) update

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

$$\begin{array}{ll} \text{minimize} & \mathbf{tr}(H_{k-1}X^{-1}) - \log\det(H_{k-1}X^{-1}) - n \\ \text{subject to} & Xs = y \end{array}$$

- minimize relative entropy between $\mathcal{N}(0, H_{k-1})$ and $\mathcal{N}(0, X)$
- 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

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

- 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

- J. Nocedal and S. J. Wright, *Numerical Optimization* (2006), chapters 6 and 7
- J. E. Dennis and R. B. Schnabel, *Numerical Methods for Unconstrained Optimization and Nonlinear Equations* (1983)