

CME 307 / MS&E 311: Optimization

Newton and quasi-Newton methods

Professor Udell

Management Science and Engineering
Stanford

September 22, 2025

Outline

Quadratic approximation

Newton's method

Quasi-Newton methods

BFGS

L-BFGS

Preconditioning

Variable metric methods

Minimize quadratic approximation

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

Suppose $f : \mathbb{R} \rightarrow \mathbb{R}$ is twice differentiable. For any $x \in \mathbb{R}$, approximate f about x :

$$\begin{aligned} f(x) &\approx f(x^{(k)}) + \nabla f(x^{(k)})^T (x - x^{(k)}) \\ &\quad + \frac{1}{2} (x - x^{(k)})^T \nabla^2 f(x^{(k)}) (x - x^{(k)}) \\ &\approx f(x^{(k)}) + \nabla f(x^{(k)})^T s + \frac{1}{2} s^T B_k s =: m_k(x) \end{aligned}$$

where $s = x - x^{(k)}$ is the **search direction** and $B_k \approx \nabla^2 f(x^{(k)})$ is the **Hessian approximation**.

If $B_k \succeq 0$, m_k is convex. to minimize,

$$B_k s + \nabla f(x^{(k)}) = 0$$

if B_k is invertible,

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

Why do we need $B_k \succ 0$?

$$x^{(k+1)} = \operatorname{argmin}_x m_k(x) = \operatorname{argmin}_x f(x) + \nabla f(x^{(k)})^T s + \frac{1}{2} s^T B_k s$$

Why do we need $B_k \succ 0$?

$$x^{(k+1)} = \operatorname{argmin}_x m_k(x) = \operatorname{argmin}_x f(x) + \nabla f(x^{(k)})^T s + \frac{1}{2} s^T B_k s$$

Q: What happens if B_k is indefinite?

Why do we need $B_k \succ 0$?

$$x^{(k+1)} = \operatorname{argmin}_x m_k(x) = \operatorname{argmin}_x f(x) + \nabla f(x^{(k)})^T s + \frac{1}{2} s^T B_k s$$

Q: What happens if B_k is indefinite?

A: Go in the direction of negative curvature; but not clear how far to go.

Why do we need $B_k \succ 0$?

$$x^{(k+1)} = \operatorname{argmin}_x m_k(x) = \operatorname{argmin}_x f(x) + \nabla f(x^{(k)})^T s + \frac{1}{2} s^T B_k s$$

Q: What happens if B_k is indefinite?

A: Go in the direction of negative curvature; but not clear how far to go.

Q: What happens if B_k is not invertible?

Why do we need $B_k \succ 0$?

$$x^{(k+1)} = \operatorname{argmin}_x m_k(x) = \operatorname{argmin}_x f(x) + \nabla f(x^{(k)})^T s + \frac{1}{2} s^T B_k s$$

Q: What happens if B_k is indefinite?

A: Go in the direction of negative curvature; but not clear how far to go.

Q: What happens if B_k is not invertible?

A: Not clear how far to go in flat directions.

Why do we need $B_k \succ 0$?

$$x^{(k+1)} = \operatorname{argmin}_x m_k(x) = \operatorname{argmin}_x f(x) + \nabla f(x^{(k)})^T s + \frac{1}{2} s^T B_k s$$

Q: What happens if B_k is indefinite?

A: Go in the direction of negative curvature; but not clear how far to go.

Q: What happens if B_k is not invertible?

A: Not clear how far to go in flat directions.

in practice

- **make it psd.** modify B_k to be positive definite

Why do we need $B_k \succ 0$?

$$x^{(k+1)} = \operatorname{argmin}_x m_k(x) = \operatorname{argmin}_x f(x) + \nabla f(x^{(k)})^T s + \frac{1}{2} s^T B_k s$$

Q: What happens if B_k is indefinite?

A: Go in the direction of negative curvature; but not clear how far to go.

Q: What happens if B_k is not invertible?

A: Not clear how far to go in flat directions.

in practice

- ▶ **make it psd.** modify B_k to be positive definite
- ▶ **Newton-CG.** use conjugate gradient to solve $B_k s = -\nabla f(x^{(k)})$. if you solve it, take the step; otherwise, CG gives a direction of negative curvature; take it!
See <https://arxiv.org/abs/1803.02924> for more details.

Why do we need $B_k \succ 0$?

$$x^{(k+1)} = \operatorname{argmin}_x m_k(x) = \operatorname{argmin}_x f(x) + \nabla f(x^{(k)})^T s + \frac{1}{2} s^T B_k s$$

Q: What happens if B_k is indefinite?

A: Go in the direction of negative curvature; but not clear how far to go.

Q: What happens if B_k is not invertible?

A: Not clear how far to go in flat directions.

in practice

- ▶ **make it psd.** modify B_k to be positive definite
- ▶ **Newton-CG.** use conjugate gradient to solve $B_k s = -\nabla f(x^{(k)})$. if you solve it, take the step; otherwise, CG gives a direction of negative curvature; take it!
See <https://arxiv.org/abs/1803.02924> for more details.
- ▶ **trust region method.** minimize nonconvex m_k over a ball

Trust region methods

suppose B_k is indefinite. solution to model problem is unbounded!

$$\operatorname{argmin}_x m_k(x) = \operatorname{argmin}_x f(x) + \nabla f(x^{(k)})^T s + \frac{1}{2} s^T B_k s$$

trust region method chooses $x^{(k+1)}$ to solve **trust region subproblem**

$$\begin{array}{ll} \text{minimize} & m_k(x) \\ \text{subject to} & \|x - x^{(k)}\| \leq \delta_k \end{array}$$

- ▶ limits step length to δ_k
- ▶ subproblem is nonconvex quadratically constrained quadratic program (QCQP)
- ▶ can solve with generalized eigenvalue solver

source: <https://www.math.uwaterloo.ca/~hwolkowi/henry/reports/previews.d/trsalgorithm10.pdf>

Which quadratic approximation?

- ▶ **Gradient descent.** use $B_k = \frac{1}{t}I$ for some $t > 0$.

$$s = -t\nabla f(x)$$

- ▶ **Newton's method.** use $B_k = \nabla^2 f(x)$.

$$s = -(\nabla^2 f(x))^{-1}\nabla f(x)$$

- ▶ **Quasi-Newton methods.** use $B_k \approx \nabla^2 f(x^{(k)})$.

$$s = -B_k^{-1}\nabla f(x)$$

if f is convex (and the appropriate derivatives exist) and $B_k \succeq 0$, we have global convergence as long as $m_k(x) \geq f(x)$ for all x . but how fast?

Outline

Quadratic approximation

Newton's method

Quasi-Newton methods

BFGS

L-BFGS

Preconditioning

Variable metric methods

Convergence rates

- ▶ **linear convergence.**

$$\lim_{k \rightarrow \infty} \frac{\|x^{(k)} - x^*\|}{\|x^{(k-1)} - x^*\|} = c \in (0, 1)$$

- ▶ **superlinear convergence.**

$$\lim_{k \rightarrow \infty} \frac{\|x^{(k)} - x^*\|}{\|x^{(k-1)} - x^*\|} = 0$$

- ▶ **quadratic convergence.**

$$\lim_{k \rightarrow \infty} \frac{\|x^{(k)} - x^*\|}{\|x^{(k-1)} - x^*\|^2} < M$$

Newton's method converges quadratically

Theorem (Local rate of convergence)

Suppose f is twice ctsly differentiable and $\nabla^2 f(x)$ is L -Lipschitz in a neighborhood of a strict local minimizer $x^ \in \operatorname{argmin} f(x)$. Then Newton's method converges to x^* quadratically near x^* .*

recall an operator F is L -Lipschitz if

$$\|F(x) - F(y)\| \leq L\|x - y\|$$

Taylor's theorem

since f is twice continuously differentiable,

$$\nabla f(y) - \nabla f(x) = \int_0^1 \nabla^2 f(x + t(y - x))(y - x) dt$$

source: <https://www.cambridge.org/core/books/optimization-for-data-analysis/C02C3708905D236AA354D1CE1739A6A2>

Newton's method converges quadratically (I)

proof: x^* is strict local min, so $\nabla f(x^*) = 0$ and $\nabla^2 f(x^*) \succ 0$.

$$\begin{aligned}x^{(k+1)} - x^* &= x^{(k)} + s^{(k)} - x^* \\&= x^{(k)} - x^* - B_k^{-1} \nabla f(x^{(k)}) \triangleright (\text{Newton's method}) \\&= (B^{(k)})^{-1} \left(B^{(k)}(x^{(k)} - x^*) - \nabla f(x^{(k)}) \right)\end{aligned}$$

by Taylor's theorem, $\nabla f(x^{(k)}) = \int_0^1 \nabla^2 f(x^* + t(x^{(k)} - x^*))(x^{(k)} - x^*) dt$, so

$$\begin{aligned}B^{(k)}(x^{(k)} - x^*) - \nabla f(x^{(k)}) &= \int_0^1 \left(\nabla^2 f(x^{(k)}) - \nabla^2 f(x^* + t(x^{(k)} - x^*)) \right) (x^{(k)} - x^*) dt \\ \|B^{(k)}(x^{(k)} - x^*) - \nabla f(x^{(k)})\| &\leq \int_0^1 \|\nabla^2 f(x^{(k)}) - \nabla^2 f(x^* + t(x^{(k)} - x^*))\| \|x^{(k)} - x^*\| dt \\ &\leq \int_0^1 L t \|x^{(k)} - x^*\|^2 dt \\ &\leq \frac{L}{2} \|x^{(k)} - x^*\|^2\end{aligned}$$

Newton's method converges quadratically (II)

now choose $r \in \mathbb{R}$ small enough that for $\|x^{(k)} - x^*\| \leq r$,

$$\|(\nabla^2 f(x^{(k)}))^{-1}\| \leq 2\|(\nabla^2 f(x^*))^{-1}\|,$$

which is possible since $\nabla^2 f(x^*) \succ 0$.

Newton's method converges quadratically (II)

now choose $r \in \mathbb{R}$ small enough that for $\|x^{(k)} - x^*\| \leq r$,

$$\|(\nabla^2 f(x^{(k)}))^{-1}\| \leq 2\|(\nabla^2 f(x^*))^{-1}\|,$$

which is possible since $\nabla^2 f(x^*) \succ 0$. then complete the proof:

$$\begin{aligned}\|x^{(k+1)} - x^*\| &\leq \frac{L}{2} \|(\nabla^2 f(x^{(k)}))^{-1}\| \|x^{(k)} - x^*\|^2 \\ &\leq \underbrace{L\|(\nabla^2 f(x^*))^{-1}\|}_{\text{constant}} \|x^{(k)} - x^*\|^2\end{aligned}$$

Outline

Quadratic approximation

Newton's method

Quasi-Newton methods

BFGS

L-BFGS

Preconditioning

Variable metric methods

Quasi-Newton methods

what's the problem with Newton's method? $\nabla^2 f(x)$ is

- ▶ expensive to compute
- ▶ expensive to invert
- ▶ not always positive definite

Quasi-Newton methods

what's the problem with Newton's method? $\nabla^2 f(x)$ is

- ▶ expensive to compute
- ▶ expensive to invert
- ▶ not always positive definite

quasi-Newton method: use a matrix $B_k \approx \nabla^2 f(x^{(k)})$ (or $H_k = B_k^{-1}$) that is

- ▶ easy to update
- ▶ easy to invert
- ▶ guaranteed to be positive definite

update B_k at each iteration to improve/maintain approximation

Quasi-Newton methods

what's the problem with Newton's method? $\nabla^2 f(x)$ is

- ▶ expensive to compute
- ▶ expensive to invert
- ▶ not always positive definite

quasi-Newton method: use a matrix $B_k \approx \nabla^2 f(x^{(k)})$ (or $H_k = B_k^{-1}$) that is

- ▶ easy to update
- ▶ easy to invert
- ▶ guaranteed to be positive definite

update B_k at each iteration to improve/maintain approximation

can still get superlinear convergence!

BFGS

BFGS is the most popular quasi-Newton method. idea:

- ▶ take step with length $\alpha_k > 0$ chosen by line search

$$x^{(k+1)} = x^{(k)} + \alpha_k(-B_k^{-1}\nabla f(x^{(k)})) =: x^{(k)} + s^{(k)}$$

- ▶ define $p = x - x^{(k+1)}$. new model will be

$$m_{k+1}(x) = f(x^{(k+1)}) + \nabla f(x^{(k+1)})^T p + \frac{1}{2}p^T B_{k+1}p$$

want gradient of m_{k+1} to match f at $x^{(k)}$ and $x^{(k+1)}$:

- ▶ match at $x^{(k+1)}$ by construction
- ▶ match at $x^{(k)}$ if

$$\nabla f(x^{(k)}) = \nabla m_{k+1}(x^{(k)} - x^{(k+1)}) = \nabla f(x^{(k+1)}) + B_{k+1}(x^{(k)} - x^{(k+1)})$$

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

$$y^{(k)} = B_{k+1}s^{(k)} \triangleright (\text{secant equation})$$

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

Secant equation

$$y^{(k)} = B_{k+1}s^{(k)}$$

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

- ▶ need $s^{(k)T}y^{(k)} > 0$ (otherwise B_{k+1} is not positive definite)
- ▶ (*) if f is strongly convex, then $s^{(k)T}y^{(k)} > 0$ for all k
(pf on next slide)

Secant equation

$$y^{(k)} = B_{k+1}s^{(k)}$$

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

- ▶ need $s^{(k)T}y^{(k)} > 0$ (otherwise B_{k+1} is not positive definite)
- ▶ (*) if f is strongly convex, then $s^{(k)T}y^{(k)} > 0$ for all k
(pf on next slide)
- ▶ for nonconvex f , can enforce $s^{(k)T}y^{(k)} > 0$ by using a line search that satisfies the **Wolfe conditions**: for search direction $p^{(k)} = -B_k^{-1}\nabla f(x^{(k)})$, constants $c_1, c_2 \in (0, 1)$,

$$f(x^{(k)} + \alpha p^{(k)}) - f(x^{(k)}) \geq \alpha c_1 \nabla f(x^{(k)})^T p^{(k)} \triangleright (\text{Armijo})$$

$$\nabla f(x^{(k)} + \alpha p^{(k)})^T p^{(k)} \geq c_2 \nabla f(x^{(k)})^T p^{(k)} \triangleright (\text{Curvature condition})$$

(but BFGS is not guaranteed to converge for nonconvex f even with exact linesearch

<https://www.ime.usp.br/~walterfm/orientacao/bfgs.pdf>)

Proof of (*)

Lemma (*)

if f is strongly convex, then $y^{(k)T} s^{(k)} > 0$ for all k

Proof of (*)

Lemma (*)

if f is strongly convex, then $y^{(k)T} s^{(k)} > 0$ for all k

proof: for f μ -strongly convex, for any $v, w \in \mathbb{R}^n$,

$$\begin{aligned} f(v) &\geq f(w) + \nabla f(w)^T (v - w) + \frac{\mu}{2} \|v - w\|^2 \\ f(w) &\geq f(v) + \nabla f(v)^T (w - v) + \frac{\mu}{2} \|w - v\|^2 \\ 0 &\geq (\nabla f(v) - \nabla f(w))^T (v - w) + \mu \|v - w\|^2 \\ \implies (y^{(k)})^T s^{(k)} &\geq \mu \|s^{(k)}\|^2 > 0 \end{aligned}$$

where we have set $v = x^{(k+1)}$, $w = x^{(k)}$ and used $s^{(k)} = x^{(k+1)} - x^{(k)}$,
 $y^{(k)} = \nabla f(x^{(k+1)}) - \nabla f(x^{(k)})$.

BFGS update

- ▶ $B_{k+1} \in \mathbf{S}_+^n$ has $n(n+1)/2$ degrees of freedom
- ▶ secant equation gives n -dimensional linear system for $B_{k+1} \implies$ many solutions!
- ▶ BFGS update chooses rank 2 update

$$B_{k+1} = B_k + \frac{y^{(k)}y^{(k)T}}{y^{(k)T}s^{(k)}} - \frac{B_k s^{(k)}s^{(k)T} B_k}{s^{(k)T} B_k s^{(k)}}$$

- ▶ equivalently, can update the inverse Hessian approximation $H_k = B_k^{-1}$:

$$H_{k+1} = (I - \rho^{(k)} s^{(k)} y^{(k)T}) H_k (I - \rho^{(k)} y^{(k)} s^{(k)T})^T + \rho^{(k)} s^{(k)} s^{(k)T}$$

where $\rho^{(k)} = \frac{1}{y^{(k)T} s^{(k)}}$ (uses Sherman-Morrison-Woodbury)

- ▶ each iteration uses $O(n^2)$ flops

Sherman Morrison Woodbury formula

Lemma

Sherman-Morrison-Woodbury formula for a matrix $H = A + UCV$ (where dimensions match)

$$H^{-1} = A^{-1} - A^{-1}U(C^{-1} + VA^{-1}U)^{-1}VA^{-1}$$

can derive from formula for 2x2 (block) matrix inverse

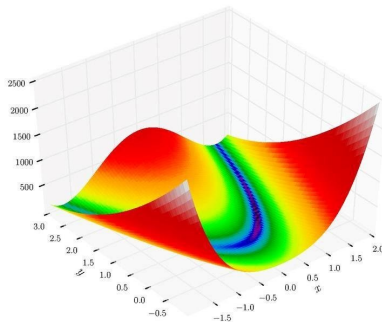
special case: $H = A + uv^T$ for $u, v \in \mathbb{R}^n$:

$$H^{-1} = A^{-1} - \frac{A^{-1}uv^T A^{-1}}{1 + v^T A^{-1}u}$$

also called **matrix inversion lemma** or any subset of names

BFGS convergence

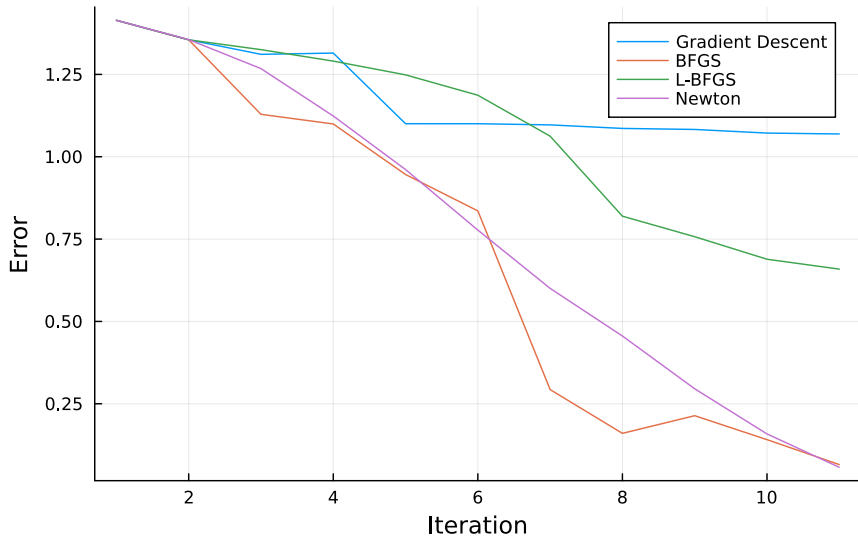
demo: try on Rosenbrock function $f(x, y) = (1 - x)^2 + 100(y - x^2)^2$



<https://github.com/stanford-cme-307/demos/blob/main/qn.jl>

BFGS in practice

Error vs Iteration



Limited memory quasi-Newton methods

main disadvantage of quasi-Newton method: need to store H or B

Limited-memory BFGS (L-BFGS): don't store B explicitly!

- ▶ instead, store the m (say, $m = 30$) most recent values of

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

- ▶ evaluate $\delta x = B_k \nabla f(x^{(k)})$ recursively, using

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

assuming $B_{k-m} = I$

Limited memory quasi-Newton methods

main disadvantage of quasi-Newton method: need to store H or B

Limited-memory BFGS (L-BFGS): don't store B explicitly!

- ▶ instead, store the m (say, $m = 30$) most recent values of

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

- ▶ evaluate $\delta x = B_k \nabla f(x^{(k)})$ recursively, using

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

assuming $B_{k-m} = I$

- ▶ advantage: for each update, just apply rank 1 + diagonal matrix to vector!
- ▶ cost per update is $O(n)$; cost per iteration is $O(mn)$
- ▶ storage is $O(mn)$

L-BFGS: interpretations

- ▶ only remember curvature of Hessian on active subspace

$$S_k = \text{span}\{s_k, \dots, s_{k-m}\}$$

- ▶ hope: locally, $\nabla f(x^{(k)})$ will approximately lie in active subspace

$$\nabla f(x^{(k)}) = g^S + g^{S^\perp}, \quad g^S \in S_k, \quad g^{S^\perp} \text{ small}$$

- ▶ L-BFGS assumes $B_k \sim I$ on S^\perp , so $B_k g^{S^\perp} \approx g^{S^\perp}$;
if g^{S^\perp} is small, it shouldn't matter much.

Outline

Quadratic approximation

Newton's method

Quasi-Newton methods

BFGS

L-BFGS

Preconditioning

Variable metric methods

Three perspectives

- ▶ precondition the function
- ▶ change the quadratic approximation
- ▶ change the metric

Three perspectives

- ▶ precondition the function
- ▶ change the quadratic approximation
- ▶ change the metric

three names:

- ▶ preconditioned
- ▶ quasi-Newton
- ▶ variable metric

Recap: convergence analysis for gradient descent

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

recall: we say (twice-differentiable) f is μ -strongly convex and L -smooth if

$$\mu I \preceq \nabla^2 f(x) \preceq LI$$

recall: if f is μ -strongly convex and L -smooth, gradient descent converges linearly

$$f(x^{(k)}) - p^* \leq \frac{Lc^k}{2} \|x^{(0)} - x^*\|^2,$$

where $c = \left(\frac{\kappa-1}{\kappa+1}\right)^2$, $\kappa = \frac{L}{\mu} \geq 1$ is condition number
 \implies want $\kappa \approx 1$

Recap: convergence analysis for gradient descent

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

recall: we say (twice-differentiable) f is μ -strongly convex and L -smooth if

$$\mu I \preceq \nabla^2 f(x) \preceq LI$$

recall: if f is μ -strongly convex and L -smooth, gradient descent converges linearly

$$f(x^{(k)}) - p^* \leq \frac{Lc^k}{2} \|x^{(0)} - x^*\|^2,$$

where $c = \left(\frac{\kappa-1}{\kappa+1}\right)^2$, $\kappa = \frac{L}{\mu} \geq 1$ is condition number
 \implies want $\kappa \approx 1$

idea: can we minimize another function with $\kappa \approx 1$ whose solution will tell us the minimizer of f ?

Preconditioning

for $D \succ 0$, the two problems

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

have solutions related by $x^* = Dz^*$

Preconditioning

for $D \succ 0$, the two problems

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

have solutions related by $x^* = Dz^*$

- ▶ gradient of $f(Dz)$ is $D^T \nabla f(Dz)$
- ▶ the second derivative (Hessian) of $f(Dz)$ is $D^T \nabla^2 f(Dz) D$

Preconditioning

for $D \succ 0$, the two problems

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

have solutions related by $x^* = Dz^*$

- ▶ gradient of $f(Dz)$ is $D^T \nabla f(Dz)$
- ▶ the second derivative (Hessian) of $f(Dz)$ is $D^T \nabla^2 f(Dz) D$

a gradient step on $f(Dz)$ with step-size $t > 0$ is

$$\begin{aligned} z^+ &= z - t D^T \nabla f(Dz) \\ Dz^+ &= Dz - t D D^T \nabla f(Dz) \\ x^+ &= x - t D D^T \nabla f(x) \end{aligned}$$

Preconditioning

for $D \succ 0$, the two problems

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

have solutions related by $x^* = Dz^*$

- ▶ gradient of $f(Dz)$ is $D^T \nabla f(Dz)$
- ▶ the second derivative (Hessian) of $f(Dz)$ is $D^T \nabla^2 f(Dz) D$

a gradient step on $f(Dz)$ with step-size $t > 0$ is

$$\begin{aligned} z^+ &= z - t D^T \nabla f(Dz) \\ Dz^+ &= Dz - t D D^T \nabla f(Dz) \\ x^+ &= x - t D D^T \nabla f(x) \end{aligned}$$

from prev analysis, gd on z converges fastest if

$$\begin{aligned} D^T \nabla^2 f(Dz) D &\approx I \\ D &\approx (\nabla^2 f(Dz))^{-1/2} \end{aligned}$$

Approximate inverse Hessian

$B = DD^T$ is called the **approximate inverse Hessian**

can fix B or update it at every iteration:

- ▶ if B is constant: called **preconditioned** method
(e.g., preconditioned conjugate gradient)
- ▶ if B is updated: called **(quasi)-Newton** method

how to choose B ? want

- ▶ $B \approx \nabla^2 f(x)^{-1}$
- ▶ easy to compute (and update) B
- ▶ fast to multiply by B

Outline

Quadratic approximation

Newton's method

Quasi-Newton methods

BFGS

L-BFGS

Preconditioning

Variable metric methods

Variable metric

definition of the gradient:

$$f(x + s) = f(x) + \langle \nabla f(x), s \rangle + \frac{1}{2} \langle s, \nabla^2 f(x) s \rangle + o(s^3)$$

wrt Euclidean inner product $\langle u, v \rangle = u^T v$

Variable metric

definition of the gradient:

$$f(x + s) = f(x) + \langle \nabla f(x), s \rangle + \frac{1}{2} \langle s, \nabla^2 f(x) s \rangle + o(s^3)$$

wrt Euclidean inner product $\langle u, v \rangle = u^T v$

now define new inner product $\langle u, v \rangle_A = u^T A v$ for some matrix $A \in \mathbf{S}_{++}^n$.

compute the gradient and Hessian wrt this inner product:

$$\begin{aligned} f(x + h) &= f(x) + \langle \nabla f(x), s \rangle + \frac{1}{2} \langle s, \nabla^2 f(x) s \rangle + o(s^3) \\ &= f(x) + \langle A^{-1} \nabla f(x), s \rangle_A + \frac{1}{2} \langle s, A^{-1} \nabla^2 f(x) s \rangle_A + o(s^3) \end{aligned}$$

so the gradient and Hessian wrt the new inner product is

$$\nabla_A f(x) = A^{-1} \nabla f(x), \quad \nabla_A^2 f(x) = \frac{1}{2} [A^{-1} \nabla^2 f(x) + \nabla^2 f(x) A^{-1}]$$