

12. Conjugate gradient method

- Krylov sequence
- conjugate gradient method for linear equations
- convergence analysis
- preconditioned conjugate gradient method
- nonlinear conjugate gradient method

12-1

Unconstrained quadratic minimization

$$\text{minimize } f(x) = \frac{1}{2}x^T Ax - b^T x$$

with $A \in \mathbf{S}_{++}^n$

- equivalent to solving $Ax = b$
- residual $r = b - Ax$ is negative gradient at x : $r = -\nabla f(x)$

conjugate gradient method

- invented by Hestenes and Stiefel around 1951
- the most widely used iterative method for solving $Ax = b$, with $A \succ 0$
- can be extended to non-quadratic unconstrained minimization

Krylov subspace

$$\mathcal{K}_0 = \{0\}, \quad \mathcal{K}_k = \text{span}\{b, Ab, \dots, A^{k-1}b\} \quad \text{for } k \geq 1$$

- a sequence of nested subspaces: $\mathcal{K}_0 \subseteq \mathcal{K}_1 \subseteq \mathcal{K}_2 \subseteq \dots$
- if $\mathcal{K}_{k+1} = \mathcal{K}_k$, then $\mathcal{K}_i = \mathcal{K}_k$ for all $i \geq k$
- $A^{-1}b \in \mathcal{K}_n$ (even when $\mathcal{K}_n \neq \mathbf{R}^n$)

this follows from Cayley-Hamilton theorem:

$$A^n + a_1A^{n-1} + \dots + a_{n-1}A + a_nI = 0$$

where $\det(\lambda I - A) = \lambda^n + a_1\lambda^{n-1} + \dots + a_{n-1}\lambda + a_n$; therefore

$$A^{-1}b = -\frac{1}{a_n} (A^{n-1}b + a_1A^{n-2}b + \dots + a_{n-1}b)$$

Krylov sequence

CG algorithm is a recursive method for computing the *Krylov sequence*

$$x^{(k)} = \underset{x \in \mathcal{K}_k}{\text{argmin}} f(x), \quad k \geq 0$$

- $x^{(n)} = A^{-1}b$
- optimality condition for $x^{(k)}$ is

$$x^{(k)} \in \mathcal{K}_k, \quad \nabla f(x^{(k)}) = Ax^{(k)} - b \in \mathcal{K}_k^\perp$$

either $x^{(k)} = A^{-1}b$, or the residual $r_k = b - Ax^{(k)}$ is orthogonal to \mathcal{K}_k

- we will see there is a simple two-term recurrence

$$x^{(k+1)} = x^{(k)} + a_k r_k + b_k (x^{(k)} - x^{(k-1)})$$

Residuals of Krylov sequence

- the residuals $r_i = b - Ax^{(i)}$ span the Krylov subspaces:

$$\mathcal{K}_k = \text{span}\{r_0, r_1, \dots, r_{k-1}\}$$

this follows from $r_i \in \mathcal{K}_i^\perp$ and $r_i = b - Ax^{(i)} \in \mathcal{K}_{i+1}$

- the residuals are mutually orthogonal:

$$r_i^T r_j = 0 \text{ for } i \neq j$$

if $j < i$, this follows from $r_j \in \mathcal{K}_i$ and $r_i \in \mathcal{K}_i^\perp$

Conjugate directions

the vectors $v_i = x^{(i)} - x^{(i-1)}$ satisfy

$$v_i^T Av_j = 0 \text{ for } i \neq j, \quad v_i^T Av_i = v_i^T r_{i-1}$$

proof

- (assume $j < i$) $v_j = x^{(j)} - x^{(j-1)} \in \mathcal{K}_j \subseteq \mathcal{K}_{i-1}$ and

$$Av_i = A(x^{(i)} - x^{(i-1)}) = -r_i + r_{i-1} \in \mathcal{K}_{i-1}^\perp$$

- follows from the fact that $t = 1$ is the minimum of

$$f(x^{(i-1)} + tv_i) = f(x^{(i-1)}) + \frac{1}{2}t^2 v_i^T Av_i - tv_i^T r_{i-1}$$

Conjugate vectors

to simplify notation we scale the vectors $v_i = x^{(i)} - x^{(i-1)}$ as

$$p_i = \frac{\|r_{i-1}\|_2^2}{v_i^T r_{i-1}} v_i \quad (p_i = 0 \text{ if } v_i = 0)$$

- $p_i^T r_{i-1} = \|r_{i-1}\|_2^2$
- $p_i^T A p_j = 0$ for $i \neq j$, so if $p_i \neq 0$, it is independent of $\{p_1, \dots, p_{i-1}\}$
- the conjugate vectors span the Krylov subspace:

$$\mathcal{K}_k = \text{span}\{p_1, p_2, \dots, p_k\}$$

- by optimizing $f(x^{(k-1)} + \alpha p_k)$ over α we can express $x^{(k)}$ as

$$x^{(k)} = x^{(k-1)} + \alpha p_k \quad \text{where} \quad \alpha = \frac{p_k^T r_{k-1}}{p_k^T A p_k}$$

Recursion for p_k

$\mathcal{K}_k = \text{span}\{p_1, p_2, \dots, p_{k-1}, r_{k-1}\}$, so we can express p_k as

$$p_1 = \beta r_0, \quad p_k = \delta r_{k-1} + \beta p_{k-1} + \sum_{i=1}^{k-2} \gamma_i p_i \quad (k > 1)$$

- taking inner products with $A p_i$ for $i \leq k-2$ gives $\gamma_i = 0$

$$p_i^T A p_k = 0, \quad p_i^T A r_{k-1} = 0 \quad (\text{because } A p_i \in \mathcal{K}_{i+1} \subseteq \mathcal{K}_{k-1})$$

- inner product with r_{k-1} gives $\delta = 1$ (from $r_{k-1}^T p_k = \|r_{k-1}\|_2^2$)
- inner product with $A p_{k-1}$ gives

$$\beta = -\frac{p_{k-1}^T A r_{k-1}}{p_{k-1}^T A p_{k-1}}$$

Basic conjugate gradient algorithm

$$x^{(0)} = 0, r_0 = b$$

for $k = 1, 2, \dots$

1. return $x^{(k-1)}$ if $\|r_{k-1}\|_2 \leq \epsilon \|b\|_2$

2. if $k = 1$, $p_k = r_0$; otherwise

$$p_k = r_{k-1} + \beta p_{k-1} \quad \text{where} \quad \beta = -\frac{p_{k-1}^T A r_{k-1}}{p_{k-1}^T A p_{k-1}}$$

3. compute

$$x^{(k)} = x^{(k-1)} + \alpha p_k \quad \text{where} \quad \alpha = \frac{\|r_{k-1}\|_2^2}{p_k^T A p_k}$$

$$\text{and } r_k = b - A x^{(k)}$$

improvements

- step 3: compute residual recursively:

$$r_k = r_{k-1} - \alpha A p_k$$

- step 2: simplify the expression for β by using

$$r_{k-1} = r_{k-2} - \frac{\|r_{k-2}\|_2^2}{p_{k-1}^T A p_{k-1}} A p_{k-1}$$

inner product with r_{k-1} gives

$$\beta = \frac{\|r_{k-1}\|_2^2}{\|r_{k-2}\|_2^2}$$

this reduces number of matrix multiplications to one per iteration

Conjugate gradient algorithm

$$x^{(0)} = 0, r_0 = b$$

for $k = 1, 2, \dots$

1. return $x^{(k-1)}$ if $\|r_{k-1}\|_2 \leq \epsilon \|b\|_2$

2. if $k = 1$, $p_1 = r_0$; else

$$p_k = r_{k-1} + \frac{\|r_{k-1}\|_2^2}{\|r_{k-2}\|_2^2} p_{k-1}$$

3. compute $\alpha = \|r_{k-1}\|_2^2 / p_k^T A p_k$ and

$$x^{(k)} = x^{(k-1)} + \alpha p_k, \quad r_k = r_{k-1} - \alpha A p_k$$

Analysis of Krylov sequence

$$\text{minimize } f(x) = \frac{1}{2} x^T A x - b^T x$$

- optimal value is

$$f(x^*) = -\frac{1}{2} b^T A^{-1} b = -\frac{1}{2} \|x^*\|_A^2$$

- suboptimality at x is

$$f(x) - f^* = \frac{1}{2} \|x - x^*\|_A^2$$

- relative error measure

$$\tau = \frac{f(x) - f^*}{f(0) - f^*} = \frac{\|x - x^*\|_A^2}{\|x^*\|_A^2}$$

here, $\|u\|_A = (u^T A u)^{1/2}$ is A -weighted norm

error after k steps in the Krylov sequence

- $x^{(k)} \in \mathcal{K}_k = \text{span}\{b, Ab, A^2b, \dots, A^{k-1}b\}$, so it can be expressed as

$$x^{(k)} = \sum_{i=1}^k \gamma_i A^{i-1}b = p(A)b$$

where $p(s) = \sum_{i=1}^k \gamma_i s^{i-1}$, a polynomial of degree $k - 1$ or less

- $x^{(k)}$ minimizes $f(x)$ over \mathcal{K}_k ; hence

$$\begin{aligned} 2(f(x^{(k)}) - f^*) &= \inf_{x \in \mathcal{K}_k} \|x - x^*\|_A^2 \\ &= \inf_{\deg p < k} \|(p(A) - A^{-1})b\|_A^2 \end{aligned}$$

we now use the eigenvalue decomposition of A to bound this quantity

simplification using eigenvalue decomposition of A

$$A = Q\Lambda Q^T = \sum_{i=1}^n \lambda_i q_i q_i^T \quad (Q^T Q = I, \quad \Lambda = \mathbf{diag}(\lambda_1, \dots, \lambda_n))$$

with $\bar{b} = Q^T b$, error expression simplifies to

$$\begin{aligned} \|(p(A) - A^{-1})b\|_A^2 &= \|(p(\Lambda) - \Lambda^{-1})\bar{b}\|_\Lambda^2 \\ &= \sum_{i=1}^n (\lambda_i p(\lambda_i) - 1)^2 \bar{b}_i^2 / \lambda_i \\ 2(f(x^{(k)}) - f^*) &= \inf_{\deg p < k} \sum_{i=1}^n (\lambda_i p(\lambda_i) - 1)^2 \bar{b}_i^2 / \lambda_i \\ &= \inf_{\deg q \leq k, q(0)=1} \sum_{i=1}^n q(\lambda_i)^2 \bar{b}_i^2 / \lambda_i \end{aligned}$$

bounds on error

- absolute error

$$\begin{aligned} f(x^{(k)}) - f^* &\leq \left(\sum_{i=1}^n \frac{\bar{b}_i^2}{2\lambda_i} \right) \inf_{\deg q \leq k, q(0)=1} \left(\max_{i=1, \dots, n} q(\lambda_i)^2 \right) \\ &= \frac{1}{2} \|x^*\|_A^2 \inf_{\deg q \leq k, q(0)=1} \left(\max_{i=1, \dots, n} q(\lambda_i)^2 \right) \end{aligned}$$

$$\left(\sum_i \bar{b}_i^2 / \lambda_i = b^T A^{-1} b = \|x^*\|_A^2 \right)$$

- relative error

$$\tau_k = \frac{\|x - x^*\|_A^2}{\|x^*\|_A^2} \leq \min_{\deg q \leq k, q(0)=1} \left(\max_{i=1, \dots, n} q(\lambda_i)^2 \right)$$

Convergence rate and spectrum of A

- if A has k distinct eigenvalues $\gamma_1, \dots, \gamma_k$, CG terminates in k steps

$$q(\lambda) = \frac{(-1)^k}{\gamma_1 \cdots \gamma_k} (\lambda - \gamma_1) \cdots (\lambda - \gamma_k)$$

has degree k , $q(0) = 1$, $q(\lambda_i) = 0$ for all i ; therefore $\tau_k = 0$

- if eigenvalues are clustered in k groups, then τ_k is small
can find $q(\lambda)$ of degree k , with $q(0) = 1$, that is small on spectrum
- if x^* is a linear combination of k eigenvectors, termination in k steps
take q of degree k with $q(\lambda_i) = 0$ where $\bar{b}_i \neq 0$; then

$$\sum_{i=1}^n q(\lambda_i)^2 \bar{b}_i^2 / \lambda_i = 0$$

other bounds (without proof)

- in terms of condition number $\kappa = \lambda_{\max}/\lambda_{\min}$

$$\tau_k \leq 2 \left(\frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1} \right)^k$$

derived by taking for q a Chebyshev polynomial on $[\lambda_{\min}, \lambda_{\max}]$

- in terms of sorted eigenvalues $\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_n$

$$\tau_k \leq \left(\frac{\lambda_k - \lambda_n}{\lambda_k + \lambda_n} \right)^2$$

derived by taking q with roots at $\lambda_1, \dots, \lambda_{k-1}$ and $(\lambda_1 + \lambda_n)/2$

Conjugate gradient method as iterative method

in exact arithmetic

- CG was originally proposed as a direct (non-iterative) method
- in theory, convergence in at most n steps

in practice

- due to rounding errors, can take $\gg n$ steps (or fail)
- CG is now used as an iterative method
- with luck (good spectrum of A), good approximation in $\ll n$ steps

Efficient matrix-vector products

cost per iteration

- each iteration requires a matrix-vector product Ax
- for general dense A , cost per iteration is $O(n^2)$, so cost of n iterations is comparable to cost of LU factorization
- CG is attractive if cost of product Ax is much less than n^2

examples

- sparse A
- structured (*e.g.*, sparse) plus low rank
- products of easy-to-multiply matrices
- fast transforms (FFT, wavelet, . . .)
- inverses of lower/upper triangular (by forward/backward substitution)
- fast Gauss transform, for $A_{ij} = \exp(-\|v_i - v_j\|^2/\sigma^2)$ (via multipole)

Shifting

- can start CG iteration at nonzero $x^{(0)}$, $r_0 = b - Ax^{(0)}$
- equivalent to CG for $Az = b - Ax^{(0)}$ with starting point $z^{(0)} = 0$
- good for 'warm start', *i.e.*, solving $Ax = b$ starting from a good initial guess (*e.g.*, the solution of another system $\tilde{A}x = \tilde{b}$, with $A \approx \tilde{A}$, $b \approx \tilde{b}$)

Preconditioned conjugate gradient algorithm

preconditioner

- apply CG after linear change of coordinates $x = Ty$, $\det T \neq 0$
- use CG to solve $T^T ATy = T^T b$; then set $x^* = T^{-1}y^*$
- T or $M = TT^T$ is called *preconditioner*

implementation

- in naive implementation, each iteration requires multiplies by T and T^T (and A); also need to compute $x^* = T^{-1}y^*$ at end
- can re-arrange computation so each iteration requires one multiply by M (and A), and no final solve $x^* = T^{-1}y^*$

called *preconditioned conjugate gradient* (PCG) algorithm

Choice of preconditioner

- if spectrum of $T^T AT$ (which is the same as the spectrum of MA) is clustered, PCG converges fast
- extreme case: $M = A^{-1}$
- trade-off between enhanced convergence, and extra cost of multiplication by M at each step
- goal is to find M that is cheap to multiply, and approximate inverse of A (or at least has a more clustered spectrum than A)

Some generic preconditioners

- diagonal: $M = \mathbf{diag}(1/A_{11}, \dots, 1/A_{nn})$
- incomplete/approximate Cholesky factorization: use $M = \hat{A}^{-1}$, where $\hat{A} = \hat{L}\hat{L}^T$ is an approximation of A with cheap Cholesky factorization
 - compute Cholesky factorization of \hat{A} , $\hat{A} = \hat{L}\hat{L}^T$
 - at each iteration, compute $Mz = \hat{L}^{-T}\hat{L}^{-1}z$ via forward/backward substitution
- examples
 - \hat{A} is central k -wide band of A
 - \hat{L} obtained by sparse Cholesky factorization of A , ignoring small elements in A , or refusing to create excessive fill-in

Applications in optimization

nonlinear conjugate gradient methods

- extend linear CG method to nonquadratic functions
- local convergence similar to linear CG
- limited global convergence theory

inexact and truncated Newton methods

- use conjugate gradient method to compute (approximate) Newton step
- less reliable than exact Newton methods, but handle very large problems

Nonlinear conjugate gradient

minimize $f(x)$

(f convex and differentiable)

modifications needed to extend linear CG algorithm of page 12–11

- replace $r_k = b - Ax^{(k)}$ with $-\nabla f(x^{(k)})$
- determine α by line search

Fletcher-Reeves CG algorithm

CG algorithm of page 12–11 modified to minimize non-quadratic convex f

given $x^{(0)}$

for $k = 1, 2, \dots$

1. return $x^{(k-1)}$ if $\|\nabla f(x^{(k-1)})\|_2 \leq \epsilon$
2. if $k = 1$, $p_1 = -\nabla f(x^{(0)})$; else

$$p_k = -\nabla f(x^{(k-1)}) + \beta p_{k-1} \quad \text{where} \quad \beta = \frac{\|\nabla f(x^{(k-1)})\|_2^2}{\|\nabla f(x^{(k-2)})\|_2^2}$$

3. update $x^{(k)} = x^{(k-1)} + \alpha p_k$ where $\alpha = \operatorname{argmin}_t f(x^{(k-1)} + tp_k)$

some observations

- first iteration is a gradient step; practical implementations restart the algorithm by taking a gradient step, for example, every n iterations
- update is gradient step with momentum term

$$x^{(k)} = x^{(k-1)} - \alpha_k \nabla f(x^{(k-1)}) + \beta_k (x^{(k-1)} - x^{(k-2)})$$

- with exact line search, reduces to linear CG for quadratic f

line search

- exact line search in step 3 implies $\nabla f(x^{(k)})^T p_k = 0$
- therefore p_k is a descent direction at $x^{(k+1)}$: from step 2,

$$\nabla f(x^{(k-1)})^T p_k = -\|\nabla f(x^{(k-1)})\|_2^2 < 0$$

Variations

Polak-Ribière: in step 2, compute β from

$$\beta = \frac{\nabla f(x^{(k-1)})^T (\nabla f(x^{(k-1)}) - \nabla f(x^{(k-2)}))}{\|\nabla f(x^{(k-2)})\|_2^2}$$

Hestenes-Stiefel

$$\beta = \frac{\nabla f(x^{(k-1)})^T (\nabla f(x^{(k-1)}) - \nabla f(x^{(k-2)}))}{p_{k-1}^T (\nabla f(x^{(k-1)}) - \nabla f(x^{(k-2)}))}$$

formulas are equivalent for quadratic f and exact line search

Interpretation as restarted BFGS method

BFGS update (page 11-5) with $H_{k-1} = I$:

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

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

- $\nabla f(x^{(k)})^T s = 0$ if $x^{(k)}$ is determined by exact line search
- quasi-Newton step in iteration k is

$$-H_k^{-1} \nabla f(x^{(k)}) = -\nabla f(x^{(k)}) + \frac{y^T \nabla f(x^{(k)})}{y^T s} s$$

this is the Hestenes-Stiefel update

nonlinear CG can be interpreted as L-BFGS with $m = 1$