15. Conjugate gradient method

- conjugate gradient method for linear equations
- complexity
- conjugate gradient method as iterative method
- applications in nonlinear optimization

Unconstrained quadratic minimization

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

with A symmetric positive definite and $n \times n$

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

Conjugate gradient method (CG)

- 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 subspaces

Definition: a sequence of subspaces

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

Properties

- subspaces are nested: $\mathcal{K}_0 \subseteq \mathcal{K}_1 \subseteq \mathcal{K}_2 \subseteq \cdots$
- dimensions increase by at most one: $\dim \mathcal{K}_{k+1} - \dim \mathcal{K}_k$ is zero or one
- if $\mathcal{K}_{k+1} = \mathcal{K}_k$, then $\mathcal{K}_i = \mathcal{K}_k$ for all $i \geq k$:

$$A^k b \in \operatorname{span}\{b, Ab, \ldots, A^{k-1}b\} \implies A^i b \in \operatorname{span}\{b, Ab, \ldots, A^{k-1}b\} \quad \text{for } i > k$$

Solution of $Ax = b$

Key property:

$$A^{-1}b \in \mathcal{K}_n$$

this holds even when $\mathcal{K}_n \neq \mathbf{R}^n$

- from Cayley–Hamilton theorem,

$$p(A) = A^n + a_1 A^{n-1} + \cdots + a_n I = 0$$

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

- multiplying on the right with $A^{-1}b$ shows

$$A^{-1}b = -\frac{1}{a_n}\left(A^{n-1}b + a_1 A^{n-2}b + \cdots + a_{n-1}b\right)$$

Krylov sequence

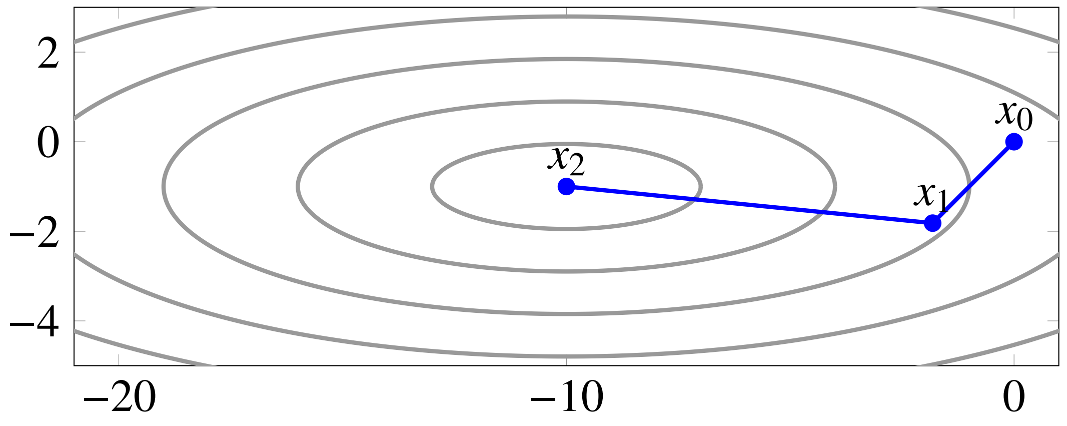
$$x_k = \underset{x \in \mathcal{K}_k}{\operatorname{argmin}} f(x), \quad k = 0, 1, \ldots$$

- from previous page, $x_n = A^{-1}b$
- CG method is a recursive method for computing the Krylov sequence $x_0, x_1, \ldots$
- we will see there is a simple two-term recurrence

$$x_{k+1} = x_k - t_k \nabla f(x_k) + s_k(x_k - x_{k-1})$$

Example

$$A = \begin{bmatrix} 1 & 0 \\ 0 & 10 \end{bmatrix}, \quad b = \begin{bmatrix} 10 \\ 10 \end{bmatrix}$$



Residuals of Krylov sequence

$$x_k = \operatorname*{argmin}_{x \in \mathcal{K}_k} f(x), \quad k = 0, 1, \ldots$$

- optimality conditions in definition of Krylov sequence:

$$x_k \in \mathcal{K}_k, \qquad \nabla f(x_k) = Ax_k - b \in \mathcal{K}_k^{\perp}$$

- hence, the residual $r_k = b - Ax_k$ satisfies

$$r_k \in \mathcal{K}_{k+1}, \qquad r_k \in \mathcal{K}_k^{\perp}$$

  the first property follows from $b \in \mathcal{K}_1$ and $x_k \in \mathcal{K}_k$

the (nonzero) residuals form an orthogonal basis for the Krylov subspaces:

$$\mathcal{K}_k = \operatorname{span}\{r_0, r_1, \ldots, r_{k-1}\}, \qquad r_i^T r_j = 0 \quad \text{for } i \neq j$$

Conjugate directions

the “steps” $v_i = x_i - x_{i-1}$ in the Krylov sequence (defined for $i \geq 1$) satisfy

$$v_i^T A v_j = 0 \quad \text{for } i \neq j, \qquad v_i^T A v_i = v_i^T r_{i-1}$$

(proof on next page)

- the vectors v_i are *conjugate:* orthogonal for inner product $\langle v, w \rangle = v^T A w$
- in particular, if $v_i \neq 0$, it is linearly independent of $v_1, \ldots, v_{i-1}$

the (nonzero) vectors v_i form a *conjugate basis* for the Krylov subspaces:

$$\mathcal{K}_k = \operatorname{span}\{v_1, v_2, \ldots, v_k\}, \qquad v_i^T A v_j = 0 \quad \text{for } i \neq j$$

Proof of properties on page 15.7

- assume $j < i$; we show that Av_i and v_j are orthogonal ($v_i^T A v_j = 0$):

$$v_j = x_j - x_{j-1} \in \mathcal{K}_j \subseteq \mathcal{K}_{i-1}$$

and

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

- the expression $v_i^T A v_i = v_i^T r_{i-1}$ follows from the fact that $t = 1$ minimizes

$$f(x_{i-1} + t v_i) = f(x_{i-1}) + \frac{1}{2} t^2 (v_i^T A v_i) - t(v_i^T r_{i-1}),$$

since $x_i = x_{i-1} + v_i$ minimizes f over the entire subspace $\mathcal{K}_i$

Conjugate vectors

it will be convenient to work with a sequence of scaled vectors $p_k = v_k/\alpha_k$ with

$$\alpha_k = \frac{v_k^T r_{k-1}}{\|r_{k-1}\|_2^2}$$

- the scaling factor α_k was chosen to satisfy

$$p_k^T r_{k-1} = \|r_{k-1}\|_2^2$$

- using $v_k^T A v_k = v_k^T r_{k-1}$ (page 15.7), we can express α_k as

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

- in this notation, the Krylov sequence and residuals satisfy

$$x_k = x_{k-1} + \alpha_k p_k, \qquad r_k = r_{k-1} - \alpha_k A p_k$$

Recursion for p_k

the vectors p_1, p_2, $\ldots$, can be computed recursively as $p_1 = r_0$,

$$p_{k+1} = r_k - \frac{p_k^T A r_k}{p_k^T A p_k} p_k, \quad k = 1, 2, \ldots \tag{1}$$

(proof on next page)

- this can be further simplified using

$$r_k = r_{k-1} - \frac{\|r_{k-1}\|_2^2}{p_k^T A p_k} A p_k \qquad \Longrightarrow \qquad \|r_k\|_2^2 = -\frac{r_k^T A p_k}{p_k^T A p_k} \|r_{k-1}\|_2^2$$

- substituting in the recursion for p_{k+1} gives

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

Proof of (1): $p_{k+1} \in \mathcal{K}_{k+1} = \operatorname{span}\{p_1, p_2, \ldots, p_k, r_k\}$, so we can express it as

$$p_{k+1} = \gamma_1 p_1 + \cdots + \gamma_{k-1} p_{k-1} + \beta p_k + \delta r_k$$

- $\delta = 1$: take inner product with r_k and use

$$r_k^T p_{k+1} = \|r_k\|_2^2, \qquad r_k^T p_1 = \cdots = r_k^T p_k = 0 \quad (r_k \in \mathcal{K}_k^\perp)$$

- $\gamma_1 = \cdots = \gamma_{k-1} = 0$: take inner products with Ap_j for $j \leq k-1$, and use

$$p_j^T A p_i = 0 \quad \text{for } j \neq i, \qquad p_j^T A r_k = 0$$

  (second equality because $Ap_j \in \mathcal{K}_{j+1} \subseteq \mathcal{K}_k$ and $r_k \in \mathcal{K}_k^\perp$)

- hence, $p_{k+1} = r_k + \beta p_k$; inner product with Ap_k shows that

$$\beta = -\frac{p_k^T A r_k}{p_k^T A p_k}$$

Conjugate gradient algorithm

define $x_0 = 0$, $r_0 = b$, and repeat for $k = 0, 1, \ldots$ until r_k is sufficiently small:

1. if $k = 0$, take $p_1 = r_0$; otherwise, take

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

2. compute

$$\alpha = \frac{\|r_k\|_2^2}{p_{k+1}^T A p_{k+1}}, \qquad x_{k+1} = x_k + \alpha p_{k+1}, \qquad r_{k+1} = r_k - \alpha A p_{k+1}$$

main computation per iteration is matrix-vector product Ap_{k+1}

Outline

- conjugate gradient method for linear equations
- **complexity**
- conjugate gradient method as iterative method
- applications in nonlinear optimization

Notation

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

Optimal value

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

Suboptimality at x

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

Relative error measure

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

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

Error after k steps

- $x_k \in \mathcal{K}_k = \operatorname{span}\{b, Ab, \ldots, A^{k-1}b\}$, so x_k can be expressed as

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

where $p(\lambda) = \sum_{i=1}^{k} c_i \lambda^{i-1}$ is a polynomial of degree $k-1$ or less

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

$$2(f(x_k) - f^\star) = \inf_{x \in \mathcal{K}_k} \|x - x^\star\|_A^2 = \inf_{\deg p < k} \|(p(A) - A^{-1})b\|_A^2$$

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

Error and spectrum of A

- eigenvalue decomposition of A

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

- define $d = Q^T b$

the expression on the previous page simplifies to

$$\begin{aligned} 2(f(x_k) - f^\star) &= \inf_{\deg p<k} \|(p(A) - A^{-1})\, b\|_A^2 \\ &= \inf_{\deg p<k} \|(p(\Lambda) - \Lambda^{-1})\, d\|_\Lambda^2 \\ &= \inf_{\deg p<k} \sum_{i=1}^n \frac{(\lambda_i p(\lambda_i) - 1)^2\, d_i^2}{\lambda_i} \\ &= \inf_{\deg q\leq k,\ q(0)=1} \sum_{i=1}^n \frac{q(\lambda_i)^2\, d_i^2}{\lambda_i} \end{aligned}$$

Error bounds

Absolute error

$$\begin{aligned} f(x_k) - f^\star &\leq \left(\sum_{i=1}^{n} \frac{d_i^2}{2\lambda_i}\right) \inf_{\deg q \leq k,\ q(0)=1} \max_{i=1,\ldots,n} q(\lambda_i)^2 \\ &= \frac{1}{2}\|x^\star\|_A^2 \inf_{\deg q \leq k,\ q(0)=1} \max_{i=1,\ldots,n} q(\lambda_i)^2 \end{aligned}$$

the equality follows from $\sum_i d_i^2/\lambda_i = b^T A^{-1} b = \|x^\star\|_A^2$

Relative error

$$\tau_k = \frac{\|x_k - x^\star\|_A^2}{\|x^\star\|_A^2} \leq \inf_{\deg q \leq k,\ q(0)=1} \max_{i=1,\ldots,n} q(\lambda_i)^2$$

Convergence rate and spectrum of A

- if A has m distinct eigenvalues $\gamma_1, \ldots, \gamma_m$, CG terminates in m steps:

$$q(\lambda) = \frac{(-1)^m}{\gamma_1 \cdots \gamma_m}(\lambda - \gamma_1) \cdots (\lambda - \gamma_m)$$

  satisfies $\deg q = m$, $q(0) = 1$, $q(\lambda_1) = \cdots = q(\lambda_n) = 0$; therefore $\tau_m = 0$

- if eigenvalues are clustered in m groups, then τ_m is small

  can find $q(\lambda)$ of degree m, with $q(0) = 1$, that is small on spectrum

- if $x^\star$ is a linear combination of m eigenvectors, CG terminates in m steps

  take q of degree m with $q(\lambda_i) = 0$ where $d_i \neq 0$; then

$$\sum_{i=1}^{n} \frac{q(\lambda_i)^2 d_i^2}{\lambda_i} = 0$$

Other bounds

we omit the proofs of the following results

- 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 $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n$

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

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

Outline

- conjugate gradient method for linear equations
- complexity
- **conjugate gradient method as iterative method**
- applications in nonlinear optimization

Conjugate gradient method as iterative method

In exact arithmetic

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

In practice

- due to rounding errors, CG method can take many more than n steps (or fail)
- CG is now used as an iterative method
- with luck (good spectrum of A), good approximation in small number of steps
- attractive if matrix-vector products are inexpensive

Preconditioning

- make change of variables $y = Bx$ with B nonsingular, and apply CG to

$$B^{-T}AB^{-1}y = B^{-T}b$$

- if spectrum of $B^{-T}AB^{-1}$ is clustered, PCG converges fast
- trade-off between enhanced convergence, cost of extra computation
- the matrix $C = B^TB$ is called the *preconditioner*

Examples

- diagonal $C = \mathbf{diag}(A_{11}, A_{22}, \ldots, A_{nn})$
- incomplete or approximate Cholesky factorization of A
- good preconditioners are often application-dependent

Naive implementation

apply algorithm of page 15.12 to $\tilde{A}y = \tilde{b}$ where $\tilde{A} = B^{-T}AB^{-1}$ and $\tilde{b} = B^{-T}b$

Algorithm:

define $y_0 = 0$, $\tilde{r}_0 = \tilde{b}$, and repeat for $k = 0, 1, \ldots$ until $\tilde{r}_k$ is sufficiently small:

1. if $k = 0$, take $\tilde{p}_1 = \tilde{r}_0$; otherwise, take

$$\tilde{p}_{k+1} = \tilde{r}_k + \frac{\|\tilde{r}_k\|_2^2}{\|\tilde{r}_{k-1}\|_2^2}\tilde{p}_k$$

2. compute

$$\alpha = \frac{\|\tilde{r}_k\|_2^2}{\tilde{p}_{k+1}^T \tilde{A}\tilde{p}_{k+1}}, \qquad y_{k+1} = y_k + \alpha\tilde{p}_{k+1}, \qquad \tilde{r}_{k+1} = \tilde{r}_k - \alpha\tilde{A}\tilde{p}_{k+1}$$

Improvements

- instead of y_k, $\tilde{p}_k$ compute iterates and steps in original coordinates

$$x_k = B^{-1} y_k, \qquad p_k = B^{-1} \tilde{p}_k$$

- compute residuals in original coordinates:

$$r_k = B^T \tilde{r}_k = b - A x_k$$

- compute squared residual norms as

$$\|\tilde{r}_k\|_2^2 = r_k^T C^{-1} r_k$$

- extra work per iteration is solving one equation to compute $C^{-1} r_k$

Preconditioned conjugate gradient algorithm

define $x_0 = 0$, $r_0 = b$, and repeat for $k = 0, 1, \ldots$ until r_k is sufficiently small:

1. solve the equation $Cs_k = r_k$
2. if $k = 0$, take $p_1 = s_0$; otherwise, take

$$p_{k+1} = s_k + \frac{r_k^T s_k}{r_{k-1}^T s_{k-1}} p_k$$

3. compute

$$\alpha = \frac{r_k^T s_k}{p_{k+1}^T A p_{k+1}}, \qquad x_{k+1} = x_k + \alpha p_{k+1}, \qquad r_{k+1} = r_k - \alpha A p_{k+1}$$

Outline

- conjugate gradient method for linear equations
- complexity
- conjugate gradient method as iterative method
- **applications in nonlinear optimization**

Applications in optimization

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 methods

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

Nonlinear conjugate gradient

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

f convex and differentiable

Modifications needed to extend linear CG algorithm of page 15.12

- replace $r_k = b - Ax_k$ with $-\nabla f(x_k)$
- determine step size α by line search

Fletcher–Reeves CG algorithm

CG algorithm of page 15.12 modified to minimize non-quadratic convex f

Algorithm: choose x_0 and repeat for $k = 0, 1, \ldots$ until $\nabla f(x_k)$ is sufficiently small:

1. if $k = 0$, take $p_1 = -\nabla f(x_0)$; otherwise, take

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

2. update $x_{k+1} = x_k + \alpha_k p_{k+1}$ where

$$\alpha_k = \underset{\alpha}{\operatorname{argmin}}\, f(x_k + \alpha p_{k+1})$$

Some observations

Interpretation

- first iteration is a gradient step
- general update is gradient step with momentum term

$$x_{k+1} = x_k - \alpha_k \nabla f(x_k) + \frac{\alpha_k \beta_k}{\alpha_{k-1}} (x_k - x_{k-1})$$

- it is common to restart the algorithm periodically by taking a gradient step

Line search

- with exact line search, reduces to linear CG for quadratic f
- exact line search in computation of α_{k-1} implies that $\nabla f(x_k)^T p_k = 0$
- therefore p_{k+1} is a descent direction at x_k:

$$\begin{aligned} \nabla f(x_k)^T p_{k+1} &= -\|\nabla f(x_k)\|_2^2 + \beta_k \nabla f(x_k)^T p_k \\ &= -\|\nabla f(x_k)\|_2^2 \\ &< 0 \end{aligned}$$

Variations

Polak–Ribière: compute β_k from

$$\beta_k = \frac{\nabla f(x_k)^T \left(\nabla f(x_k) - \nabla f(x_{k-1})\right)}{\|\nabla f(x_{k-1})\|_2^2}$$

Hestenes–Stiefel: compute β_k from

$$\beta_k = \frac{\nabla f(x_k)^T \left(\nabla f(x_k) - \nabla f(x_{k-1})\right)}{p_k^T \left(\nabla f(x_k) - \nabla f(x_{k-1})\right)}$$

formulas are equivalent for quadratic f and exact line search

References

- S. Boyd, *Lecture slides and notes for EE364b, Convex Optimization II*, lectures on the conjugate gradient method.
- G. H. Golub and C. F. Van Loan, *Matrix Computations* (1996), chapter 10.
- C. T. Kelley, *Iterative Methods for Linear and Nonlinear Equations* (1995), chapter 2.
- J. Nocedal and S. J. Wright, *Numerical Optimization* (2006), chapter 5.
- H. A. van der Vorst, *Iterative Krylov Methods for Large Linear Systems* (2003).