3. Conjugate gradient method

- conjugate gradient method for linear equations
- convergence analysis
- 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 $A \in \mathbf{S}^n_{++}$

- equivalent to solving linear equation $Ax = b$
- residual $r = b - Ax$ is 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 nested subspaces ($\mathcal{K}_0 \subseteq \mathcal{K}_1 \subseteq \mathcal{K}_2 \subseteq \cdots$)

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

if $\mathcal{K}_{k+1} = \mathcal{K}_k$, then $\mathcal{K}_i = \mathcal{K}_k$ for all $i \geq k$

Key property: $A^{-1}b \in \mathcal{K}_n$ (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 $p(\lambda) = \det(\lambda I - A) = \lambda^n + a_1\lambda^{n-1} + \cdots + a_{n-1}\lambda + a_n$

- therefore

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

Krylov sequence

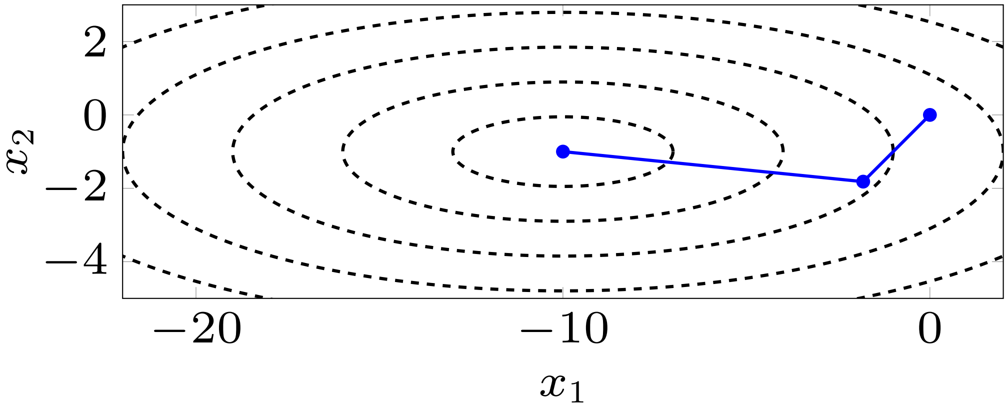
$$x^{(k)} = \operatorname*{argmin}_{x \in \mathcal{K}_k} f(x), \quad k \geq 0$$

- from previous page, $x^{(n)} = A^{-1}b$
- CG is a recursive method for computing the Krylov sequence $x^{(0)}$, $x^{(1)}$, ...
- we will see there is a simple two-term recurrence

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

Example

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



Residuals of Krylov sequence

- 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:

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

Conjugate directions

the 'steps' $v_i = x^{(i)} - x^{(i-1)}$ in the Krylov sequence 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 independent of $v_1, \ldots, v_{i-1}$

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

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

Proof of properties on page 3-6 (assume $j < i$)

- $v_i^T A v_j = 0$ because

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

  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

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

  (since $x^{(i)} = x^{(i-1)} + v_i$ minimizes f over the entire subspace $\mathcal{K}_i$)

Conjugate vectors

instead of v_i, we will work a sequence p_i of scaled vectors v_i:

$$p_i = \frac{\|r_{i-1}\|_2^2}{v_i^T r_{i-1}} \, v_i$$

- scaling factor is chosen to satisfy $r_{i-1}^T p_i = \|r_{i-1}\|_2^2$; equivalently,

$$-\nabla f(x^{(i-1)})^T p_i = \|\nabla f(x^{(i-1)})\|_2^2$$

- using $v_i^T A v_i = v_i^T r_{i-1}$ (page 3-6), we can write the scaling factor as

$$\frac{\|r_{i-1}\|_2^2}{v_i^T r_{i-1}} = \frac{\|r_{i-1}\|_2^2}{v_i^T A v_i} = \frac{p_i^T A p_i}{\|r_{i-1}\|_2^2}$$

- with this notation we can write the update as

$$x^{(i)} = x^{(i-1)} + \alpha p_i, \qquad \alpha = \frac{\|r_{i-1}\|_2^2}{p_i^T A p_i}$$

Recursion for p_k

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

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

- $\gamma_1 = \cdots = \gamma_{k-2} = 0$: take inner products with Ap_j for $j \leq k-2$, and use

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

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

- $\delta = 1$: take inner product with r_{k-1} and use $r_{k-1}^T p_k = \|r_{k-1}\|_2^2$

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

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

Basic conjugate gradient algorithm

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

For $k = 1, 2, \ldots$

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

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

2. compute

$$\alpha = \frac{\|r_{k-1}\|_2^2}{p_k^T A p_k}, \qquad x^{(k)} = x^{(k-1)} + \alpha p_k, \qquad r_k = b - Ax^{(k)}$$

if r_k is sufficiently small, return $x^{(k)}$

Improvements

Step 2: compute residual recursively:

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

Step 1: simplify the expression for β by using

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

taking inner product with r_{k-1} gives

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

this reduces number of matrix-vector products to one per iteration (product Ap_k)

Conjugate gradient algorithm

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

For $k = 1, 2, \ldots$

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

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

2. compute

$$\alpha = \frac{\|r_{k-1}\|_2^2}{p_k^T A p_k}, \qquad x^{(k)} = x^{(k-1)} + \alpha p_k, \qquad r_k = r_{k-1} - \alpha A p_k$$

if r_k is sufficiently small, return $x^{(k)}$

Outline

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

Notation

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

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

Error after k steps

- $x^{(k)} \in \mathcal{K}_k = \operatorname{span}\{b, Ab, A^2b, \ldots, A^{k-1}b\}$, 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 some polynomial of degree $k-1$ or less

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

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

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 $d = Q^Tb$

expression on previous page simplifies to

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

Error bounds

Absolute error

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

(equality follows from $\sum_i d_i^2/\lambda_i = b^TA^{-1}b = \|x^\star\|_A^2$)

Relative error

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

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 λ_1, …, λ_{k-1} and $(\lambda_1 + \lambda_n)/2$

Outline

- conjugate gradient method for linear equations
- convergence analysis
- **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

define $\tilde{A} = B^{-T}AB^{-1}$ and apply algorithm of page 3-12 to $\tilde{A}y = B^{-T}b$

Initialize: $y^{(0)} = 0$, $\tilde{r}_0 = B^{-T}b$

For $k = 1, 2, \ldots$

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

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

2. define $\tilde{A} = B^{-T}AB^{-1}$ and compute

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

if $\tilde{r}_k$ is sufficiently small, return $B^{-1}y^{(k)}$

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:

$$r_k = B^T\tilde{r}_k = b - Ax^{(k)}$$

- compute squared residual norms as

$$\|\tilde{r}_{k-1}\|_2^2 = r_{k-1}^T C^{-1} r_{k-1}$$

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

Preconditioned conjugate gradient algorithm

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

For $k = 1, 2, \ldots$

1. solve the equation $Cs_k = r_{k-1}$

2. if $k = 1$, take $p_k = s_k$; otherwise, take

$$p_k = s_k + \beta p_{k-1} \quad \text{where} \quad \beta = \frac{r_{k-1}^T s_k}{r_{k-2}^T s_{k-1}}$$

3. compute

$$\alpha = \frac{r_{k-1}^T s_k}{p_k^T A p_k}, \qquad x^{(k)} = x^{(k-1)} + \alpha p_k, \qquad r_k = r_{k-1} - \alpha A p_k$$

if r_k is sufficiently small, return $x^{(k)}$

Outline

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

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

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

(f convex and differentiable)

Modifications needed to extend linear CG algorithm of page 3-12

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

Fletcher-Reeves CG algorithm

CG algorithm of page 3-12 modified to minimize non-quadratic convex f

Initialize: choose $x^{(0)}$

For $k = 1, 2, \ldots$

1. if $k = 1$, take $p_1 = -\nabla f(x^{(0)})$; otherwise, take

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

2. update $x^{(k)} = x^{(k-1)} + \alpha_k p_k$ where

$$\alpha_k = \underset{\alpha}{\operatorname{argmin}}\, f(x^{(k-1)} + \alpha p_k)$$

   if $\nabla f(x^{(k)})$ is sufficiently small, return $x^{(k)}$

Some observations

Interpretation

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

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

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

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

Variations

Polak-Ribière: in step 1, 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 2-5) with $H_{k-1} = I$:

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

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

- $\nabla f(x^{(k)})^T s = 0$ 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 $m = 1$

References

- S. Boyd, Lecture notes for EE364b, Convex Optimization II.
- G. H. Golub and C. F. Van Loan, *Matrix Computations* (1996), chapter 10.
- J. Nocedal and S. J. Wright, *Numerical Optimization* (2006), chapter 5.
- H. A. van der Vorst, *Iterative Krylov Methods for Large Linear Systems* (2003).