CONJUGATE GRADIENT METHODS

LONG CHEN

We shall present iterative methods for solving linear algebraic equation $Au = b$ based on Krylov subspaces. We derive conjugate gradient (CG) method developed by Hestenes and Stiefel in 1950s [1] for symmetric and positive definite matrix A and briefly mention the GMRES method [3] for general non-symmetric matrix systems.

1. Problem Setting and Gradient Methods

We introduce the conjugate gradient (CG) method for solving

$$Au = b. \tag{1}$$

where A is a symmetric and positive definite (SPD) operator defined on an N-dimensional Hilbert space $\mathbb{V}$ with inner product $(\cdot,\cdot)$, and its preconditioned version PCG. When $\mathbb{V} = \mathbb{R}^N$, A is an SPD matrix and $(\cdot,\cdot)$ is the ℓ_2-inner product of vectors.

Equation (1) can be derived from the following optimization problem

$$\min_{u\in\mathbb{V}} f(u) := \frac{1}{2}(Au,u) - (b,u). \tag{2}$$

As f is strongly convex, the global minimizer exists and unique and satisfies Euler equation $\nabla f(u) = 0$ which is exactly equation (1) .

We shall derive CG from the A-orthogonal projection to subspaces. We use $(\cdot,\cdot)$ for the standard inner product and $(\cdot,\cdot)_A$ for the inner product introduced by the SPD operator A:

$$(x,y)_A = (Ax,y) = (x,Ay) = x^\intercal Ay = y^\intercal Ax,$$

which induce a norm $\|x\|_A = \sqrt{(x,x)_A}$. When talking about orthogonality, we refer to the default $(\cdot,\cdot)$ inner product and emphasize the orthogonality in $(\cdot,\cdot)_A$ by *A-orthogonal* or *conjugate*. Given a vector $x \in \mathbb{V}$, the A-orthogonal projection of x to a subspace $S \subseteq \mathbb{V}$ is a vector in S, denoted by $\mathrm{Proj}_S^A x$, by the relation

$$(\mathrm{Proj}_S^A x, y)_A = (x,y)_A, \quad \forall y \in S.$$

The simplest iterative method for solving the minimization problem (2) is the gradient descent method in the form

$$u_{k+1} = u_k + \alpha_k r_k = u_k - \alpha_k \nabla f(u_k),$$

where α_k is the step size and the residual $r_k = b - Au_k = -\nabla f(u_k)$. In the so-called *steepest gradient descent* method, the step size is obtained by

$$\alpha_k = \arg\min_{\alpha} f(u_k + \alpha r_k). \tag{3}$$

Instead of looking at the minimization of the objective function, we shall consider the A-orthogonal projection of the current error $u - u_k$ to the subspace $S = \mathrm{span}\{r_k\}$; see Fig 1. That is we compute u_{k+1} by the identity

$$u_{k+1} - u_k = \mathrm{Proj}_S^A(u - u_k),$$

Date: November 12, 2020.

which implies

$$\alpha_k = \frac{(u-u_k, r_k)_A}{(r_k, r_k)_A} = \frac{(r_k, r_k)}{(Ar_k, r_k)}. \tag{4}$$

and the orthogonality

$$(r_{k+1}, r_k) = (u - u_{k+1}, r_k)_A = (u - u_k - \mathrm{Prof}_S^A(u-u_k), r_k)_A = 0. \tag{5}$$

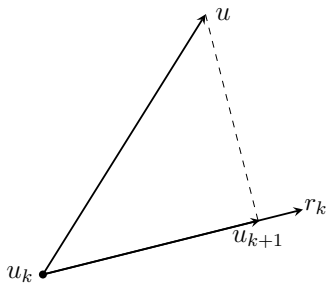

FIGURE 1. View the steepest gradient descent method as A-orthogonal projection.

It is straightforward to verify the step size obtained by (3) is the same as that in (4). So the residual vectors which is the negative of the gradient vectors in two consecutive steps of the steepest gradient descent method are orthogonal.

We then give convergence analysis of the steepest gradient descent method to show that it converges as the optimal Richardson method. Recall that in the Richardson method, the optimal choice $\alpha^* = 2/(\lambda_{\min}(A) + \lambda_{\max}(A))$ requires the information of eigenvalues of A, while in the gradient method, α_k is computed using the action of A only, cf. (4). The optimality is build into the optimization of the step size (so-called the exact line search).

Theorem 1.1. *Let u_k be the k-th iteration in the steepest gradient descent method with an initial guess u_0. We have*

$$\|u - u_k\|_A \leq \left(\frac{\kappa(A)-1}{\kappa(A)+1}\right)^k \|u - u_0\|_A, \tag{6}$$

where $\kappa(A) = \lambda_{\max(A)}/\lambda_{\min(A)}$ is the condition number of A.

Proof. In view of the A-orthogonality in (5), we have

$$\|u - u_{k+1}\|_A = \inf_{\alpha\in\mathbb{R}} \|u - (u_k + \alpha r_k)\|_A = \inf_{\alpha\in\mathbb{R}} \|(I - \alpha A)(u - u_k)\|_A.$$

Since $I - \alpha A$ is symmetric, $\|I - \alpha A\|_A = \rho(I - \alpha A)$. Consequently

$$\|u - u_{k+1}\|_A \leq \inf_{\alpha\in\mathbb{R}} \rho(I - \alpha A)\|u - u_k\|_A = \frac{\kappa(A)-1}{\kappa(A)+1}\|u - u_k\|_A.$$

The proof is completed by recursion on k. □

We now give geometric explanation. The level set of $\|u\|_A^2$ is an ellipsoid and the condition number of A is the ratio of the major and the minor axis of the ellipsoid. When $\kappa(A) \gg 1$, the ellipsoid is very skinny and the gradient method will take a long zig-zag

path (each turn is orthogonal) towards the solution while conjugate gradient methods will take a shorter one; see the figure below.

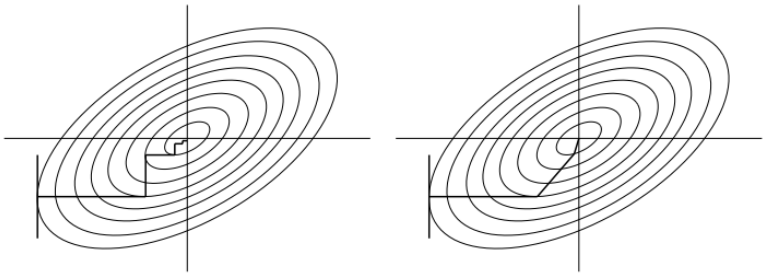

FIGURE 2. Steepest gradient descent vs. conjugate gradient directions.

2. CONJUGATE GRADIENT METHODS

By tracing back to the initial guess u_0, the $k+1$-th step of the steepest gradient descent method can be written as

$$u_{k+1} = u_0 + \alpha_0 r_0 + \alpha_1 r_1 + \cdots + \alpha_k r_k.$$

Let

$$\mathbb{V}_k = \mathrm{span}\{r_0, r_1, \cdots, r_k\}. \tag{7}$$

The correction $\sum_{i=0}^{k} \alpha_i r_i$ can be thought as an approximate solution of the residual equation

$$Ae_0 = r_0,$$

in $\mathbb{V}_k$ by computing the coefficients along each basis vector $\{r_0, r_1, \cdots, r_k\}$. It is not the best approximation of $e_0 = u - u_0$ in the subspace $\mathbb{V}_k$. Here the 'best' refers to the approximation in the A-norm and can be found by computing the A-orthogonal projection $\mathrm{Proj}^A_{V_k}(u - u_0) \in \mathbb{V}_k$. If using the basis (7), to compute the A-orthogonal projection, one needs to invert the Gram matrix $M = ((r_i, r_j)_A)$, while in the steepest descent gradient, only diagonal of M is inverted.

If we can find an A-orthogonal basis, i.e.

$$\mathbb{V}_k = \mathrm{span}\{p_0, p_1, \cdots, p_k\}, \quad (p_i, p_j)_A = 0 \text{ for } i \neq j, \tag{8}$$

the projection can be found component by component

$$\mathrm{Proj}^A_{V_k}(u - u_0) = \sum_{i=0}^{k} \alpha_i p_i, \quad \alpha_i = \frac{(u - u_0, p_i)_A}{(p_i, p_i)_A}, \quad \text{for } i = 0, \cdots k.$$

Equivalently the corresponding Gram matrix $((p_i, p_j)_A)$ is diagonal.

Conjugate gradient method will construct an A-orthogonal basis by recrusion. Start from an initial guess u_0. Let $p_0 = r_0 = -\nabla f(u_0)$. For $k = 0, 1, 2, \ldots, n$, let $\mathbb{V}_k = \mathrm{span}\{p_0, p_1, \cdots, p_k\}$ be a subspace spanned by A-orthogonal basis, i.e. $(p_i, p_j)_A = 0$ for $i \neq j, i, j = 0, \ldots, k$.

CG consists of three steps:

(1) compute u_{k+1} by the A-orthogonal projection of $u - u_0$ to $\mathbb{V}_k$.
(2) add residual vector r_{k+1} to $\mathbb{V}_k$ to get $\mathbb{V}_{k+1}$.

(3) apply Gram-Schmit process to get $A-$orthogonal vector p_{k+1} .

We now briefly explain each step and present recursive formulae.

To simplify notation, we denote by $P_k = \mathrm{Proj}_{V_k}^A$. Given a subspace $\mathbb{V}_k$ spanned by an A-orthogonal basis $\{p_0, p_1, \cdots, p_k\}$, we compute u_{k+1} by

$$u_{k+1} - u_0 = P_k(u - u_0).$$

The projection can be computed without knowing u as $Au = b$ is known. By definition, $(u_{k+1} - u_0, p_i)_A = (P_k(u - u_0), p_i)_A = (u - u_0, p_i)_A = (A(u - u_0), p_i) = (r_0, p_i)$ for $i = 0, 1, \ldots, k$, which leads to the formulae

$$u_{k+1} = u_0 + \sum_{i=0}^{k} \alpha_i p_i = u_k + \alpha_k p_k,$$

where

$$\alpha_i = \frac{(u - u_0, p_i)_A}{(p_i, p_i)_A} = \frac{(r_0, p_i)}{(Ap_i, p_i)}.$$

With u_{k+1}, we can compute a new vector

$$r_{k+1} = b - Au_{k+1} = r_0 - \sum_{i=0}^{k} \alpha_i Ap_i = r_k - \alpha_k Ap_k.$$

If $r_{k+1} = 0$, which means $u_{k+1} = u$ is the solution, then we stop. Otherwise expand the subspace to a larger one $\mathbb{V}_{k+1} = \mathrm{span}\{p_0, p_1, \cdots, p_k, r_{k+1}\}$.

Then apply Gram-Schmit process to make new added vector r_{k+1} to be A-orthogonal to others. The new conjugate direction is

$$p_{k+1} = r_{k+1} + \sum_{i=0}^{k} \beta_i p_i, \quad \beta_i = -\frac{(r_{k+1}, p_i)_A}{(p_i, p_i)_A}.$$

The magic of CG algorithm is that only β_k is needed due to the orthogonality we shall explore now.

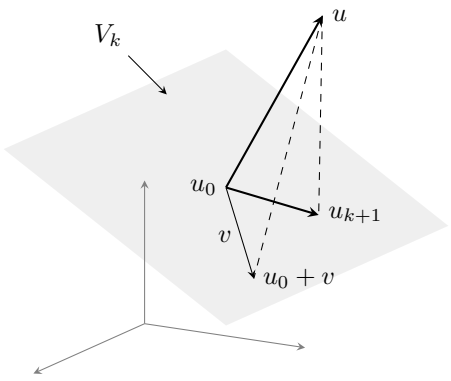

FIGURE 3. Projection of $u - u_0$ to the space $\mathbb{V}_k$.

Lemma 2.1. *The residual r_{k+1} is orthogonal to $\mathbb{V}_k$.*

Proof. We write

$$u - u_{k+1} = u - u_0 - (u_{k+1} - u_0) = (I - P_k)(u - u_0). \tag{9}$$

Therefore $u - u_{k+1} \perp_A \mathbb{V}_k$, which is equivalent to $r_{k+1} = A(u - u_{k+1}) \perp \mathbb{V}_k$. The orthogonality is thus proved; see Fig. 3 for an illustration. □

This lemma shows the advantage of the conjugate gradient method over the gradient method. The new residual is orthogonal to the whole space not only to one residual vector in the previous step.

In view of (9), we have the optimality

$$\|u - u_{k+1}\|_A = \inf_{v \in \mathbb{V}_k} \|u - (u_0 + v)\|_A,$$

which will be used later on to give convergence analysis of CG algorithm.

Since at every step, the residual vector is added to expand the subspace, it can be easily proved by induction that

$$\mathbb{V}_k = \operatorname{span}\{p_0, p_1, \cdots, p_k\} = \operatorname{span}\{r_0, r_1, \cdots, r_k\}.$$

We derive more efficient formulae for α_k and β_k.

Proposition 2.2.

$$\alpha_k = \frac{(r_k, r_k)}{(Ap_k, p_k)}.$$

Proof. We first use the A-orthogonality of basis $\{p_i\}$, i.e., $(p_k, Ap_i) = 0$ for $0 \le i \le k-1$, and the formulation $r_k = r_0 - \sum_{i=0}^{k-1} \alpha_i A p_i$ to get

$$(u - u_0, p_k)_A = (r_0, p_k) = (r_k, p_k).$$

Then we use the orthogonality $r_k \perp \mathbb{V}_{k-1}$ proved in Lemma 2.1, i.e., $(r_k, p_i) = 0$, for $0 \le i \le k-1$, and the formulae $p_k = r_k + \sum_{i=0}^{k-1} \beta_i p_i$ to get

$$(r_k, p_k) = (r_k, r_k).$$

Recall that α_k is the coefficient of $u - u_0$ corresponding to the basis p_k, then

$$\alpha_k = \frac{(u - u_0, p_k)_A}{(p_k, p_k)_A} = \frac{(r_k, p_k)}{(Ap_k, p_k)} = \frac{(r_k, r_k)}{(Ap_k, p_k)}.$$

□

This proposition shows if $r_k \neq 0$, then $\alpha_k \neq 0$ and r_{k+1} is linear independent to $\mathbb{V}_k$.

Lemma 2.3. *The residual r_{k+1} is A-orthogonal to $\mathbb{V}_{k-1}$.*

Proof. If the algorithm stops at the k-th step, then $r_{k+1} = 0$ and the statement is true. Otherwise the algorithm does not stop at the k-th step implies $r_i \neq 0$ and consequently $\alpha_i \neq 0$ for $i \le k-1$. By the recursive formula for the residual $r_{i+1} = r_i - \alpha_i A p_i$. As $\alpha_i \neq 0$, we get $Ap_i \in \operatorname{span}\{r_i, r_{i+1}\} \subset \mathbb{V}_k$ for $0 \le i \le k-1$. Since we have proved $r_{k+1} \perp \mathbb{V}_k$, we get $(r_{k+1}, p_i)_A = (r_{k+1}, Ap_i) = 0$ for $0 \le i \le k-1$, i.e. r_{k+1} is A-orthogonal to $\mathbb{V}_{k-1}$. □

Recall the formulae for p_{k+1} obtained by Gram-Schmit process is

$$p_{k+1} = r_{k+1} + \sum_{i=0}^{k} \beta_i p_i, \quad \beta_i = -\frac{(r_{k+1}, p_i)_A}{(p_i, p_i)_A}.$$

Now as r_{k+1} is A-orthogonal to $p_0, \ldots, p_{k-1}$, all $\beta_i = 0$ for $i = 0, \ldots, k-1$ and left with a three-term formula

$$p_{k+1} = r_{k+1} + \beta_k p_k, \quad \beta_k = -\frac{(r_{k+1}, p_k)_A}{(p_k, p_k)_A}.$$

We can use the simplified formula of α_k to simply β_k.

Proposition 2.4.

$$\beta_k = \frac{(r_{k+1}, r_{k+1})}{(r_k, r_k)}.$$

Proof. We use the recursion $r_{k+1} = r_k - \alpha_k A p_k$ and the orthogonality $r_{k+1} \perp r_k$ to get

$$(r_{k+1}, r_{k+1}) = -\alpha_k (r_{k+1}, A p_k).$$

Then using the formula we proved in Proposition 2.2 $\alpha_k (A p_k, p_k) = (r_k, r_k)$, to get

$$\beta_k = -\frac{(r_{k+1}, p_k)_A}{(p_k, p_k)_A} = -\frac{(r_{k+1}, A p_k)}{(A p_k, p_k)} = \frac{(r_{k+1}, r_{k+1})}{(r_k, r_k)}.$$

□

Note that the correction vector $u_{k+1} - u_0 \in \mathbb{V}_k$. The space we are looking for u_{k+1} is in the affine space $u_0 + \mathbb{V}_k$. As $f(v) - f(u) = \frac{1}{2}\|v - u\|_A^2$, we obtain another optimality

$$f(u_{k+1}) = \inf_{v \in u_0 + \mathbb{V}_k} f(v).$$

Namely at the k-th step of CG, it is to find the minimum in the affine space $u_0 + \mathbb{V}_k$.

Exercise 2.5. *Show that*

$$\|u - u_{k+1}\|_A = \inf_{\alpha \in \mathbb{R}} \|u - (u_k + \alpha p_k)\|_A.$$

That is u_{k+1} can be found by the steepest descent method starting from u_k and moving along the searching direction p_k which gives the formulae $\alpha_k = (r_k, p_k)_A / (p_k, p_k)_A$.

We summarize recursive formula of CG below. Starting from an initial guess u_0 and $p_0 = r_0$, for $k = 0, 1, 2, \cdots$, we use three recursive formulae to compute

$$\begin{aligned}
u_{k+1} &= u_k + \alpha_k p_k, & \alpha_k &= \frac{(u - u_0, p_k)_A}{(p_k, p_k)_A} = \frac{(r_k, r_k)}{(A p_k, p_k)}, \\
r_{k+1} &= r_k - \alpha_k A p_k, & & \\
p_{k+1} &= r_{k+1} + \beta_k p_k, & \beta_k &= -\frac{(r_{k+1}, p_k)_A}{(p_k, p_k)_A} = \frac{(r_{k+1}, r_{k+1})}{(r_k, r_k)}.
\end{aligned}$$

The method is called *conjugate gradient* (CG) method since the conjugate direction is obtained by a correction of the gradient (the residual) direction.

We present the algorithm of the conjugate gradient method as follows.

```
function u = CG(A,b,u,tol)
tol = tol*norm(b);
k = 1;
r = b - A*u;
p = r;
r2 = r'*r;
while sqrt(r2) ≥ tol && k<length(b)
    Ap = A*p;
    alpha = r2/(p'*Ap);
```

```
    u = u + alpha*p;
    r = r - alpha*Ap;
    r2old = r2;
    r2 = r'*r;
    beta = r2/r2old;
    p = r + beta*p;
    k = k + 1;
end
```

Several remarks on the realization are listed below:

- The most time consuming part is the matrix-vector multiplication `A*p`. There is no need to form the matrix A explicitly. A subroutine to compute the matrix-vector multiplication is enough. This is an attractive feature of Krylov subspace methods.
- The error is measured by the relative error of the residual $\|r\| \leq \text{tol}\|b\|$.
- A maximum iteration step is given to avoid a large loop.

3. Convergence Analysis of Conjugate Gradient Method

Recall the two bases of $\mathbb{V}_k$:

$$\mathbb{V}_k = \text{span}\{r_0, r_1, \cdots, r_k\} = \text{span}\{p_0, p_1, \cdots, p_k\}.$$

We now give another basis of $\mathbb{V}_k$ which is important for the convergence analysis.

Lemma 3.1.

$$\mathbb{V}_k = \text{span}\{r_0, Ar_0, \cdots, A^k r_0\}.$$

Proof. We prove it by induction. For $k = 0$, it is trivial. Suppose the statement holds for $k = i$. We prove it holds for $k = i + 1$ by noting that

$$\mathbb{V}_{i+1} = \mathbb{V}_i + \text{span}\{r_{i+1}\} = \mathbb{V}_i + \text{span}\{r_i, Ap_i\} = \mathbb{V}_i + \text{span}\{Ap_i\} = \mathbb{V}_i + A\mathbb{V}_i.$$

□

The space $\text{span}\{r_0, Ar_0, \cdots, A^k r_0\}$ is called *Krylov subspace*. The CG method belongs to a large class of Krylov subspace iterative methods for solving linear algebraic equation.

Theorem 3.2. *Let A be SPD and let u_k be the kth iteration in the CG method with an initial guess u_0. Then*

$$\|u - u_k\|_A = \inf_{v \in \mathbb{V}_{k-1}} \|u - (u_0 + v)\|_A, \tag{10}$$

$$\|u - u_k\|_A = \inf_{\substack{p_k \in \mathcal{P}_k, \\ p_k(0)=1}} \|p_k(A)(u - u_0)\|_A, \tag{11}$$

$$\|u - u_k\|_A \leq \inf_{\substack{p_k \in \mathcal{P}_k, \\ p_k(0)=1}} \sup_{\lambda \in \sigma(A)} |p_k(\lambda)| \, \|u - u_0\|_A. \tag{12}$$

Proof. The first identity is from the fact $u - u_k = (I - P_{k-1})(u - u_0)$. For $v \in \mathbb{V}_{k-1}$, it can be expanded as

$$v = \sum_{i=0}^{k-1} c_i A^i r_0 = \sum_{i=1}^{k} c_{i-1} A^i (u - u_0).$$

Let $p_k(x) = 1 - \sum_{i=1}^{k} c_{i-1} x^i$. Then

$$u - (u_0 + v) = p_k(A)(u - u_0).$$

The identity (11) then follows from (10).

Since A^i is symmetric in the A-inner product, we have

$$\|p_k(A)\|_A = \rho(p_k(A)) = \sup_{\lambda\in\sigma(A)} |p_k(\lambda)|,$$

which leads to the estimate (12). □

The polynomial $p_k \in \mathcal{P}_k$ with constraint $p_k(0) = 1$ will be called the *residual polynomial*. Various convergence results of CG method can be obtained by choosing specific residual polynomials.

Corollary 3.3. *Let A be SPD with eigenvectors $\{\phi\}_{i=1}^N$. Let $b \in \operatorname{span}\{\phi_{i_1}, \phi_{i_2}, \cdots, \phi_{i_k}\}$, $k \leq N$. Then the CG method with $u_0 = 0$ will find the solution $Au = b$ in at most k iterations. In particular, for a given $b \in \mathbb{R}^N$, the CG algorithm will find the solution $Au = b$ within N iterations. Namely CG can be viewed as a direct method.*

Proof. Let $b = \sum_{l=1}^k b_l\phi_{i_l}$. Then $u = \sum_{l=1}^k u_l\phi_{i_l}$ with $u_l = b_l/\lambda_{i_l}$. Choose the polynomial $p(x) = \prod_{l=1}^k(1 - x/\lambda_{i_l})$. Then p is a residual polynomial of order k and $p(0) = 1, p(\lambda_{i_l}) = 0$. and

$$p(A)u = p(A)\sum_{l=1}^k u_l\phi_{i_l} = \sum_{l=1}^k u_l p(\lambda_{i_l})\phi_{i_l} = 0.$$

The assertion is then from the identity (11). □

Remark 3.4. CG method can be also applied to symmetric and positive semi-definite matrix A. Let $\{\phi_i\}_{i=1}^k$ be the eigenvectors associated to $\lambda_{\min}(A) = 0$. Then from Corollary 3.3, if $b \in \operatorname{range}(A) = \operatorname{span}\{\phi_{k+1}, \phi_{k+2}, \cdots, \phi_N\}$, then the CG method with $u_0 \in \operatorname{range}(A)$ will find a solution $Au = b$ within $N - k - 1$ iterations.

CG is invented as a direct method but it is more effective to use as an iterative method. The rate of convergence depends crucially on the distribution of eigenvalues of A and could converge to the solution within certain tolerance in steps $k \ll N$.

Theorem 3.5. *Let u_k be the k-th iteration of the CG method with u_0. Then*

$$\|u - u_k\|_A \leq 2\left(\frac{\sqrt{\kappa(A)} - 1}{\sqrt{\kappa(A)} + 1}\right)^k \|u - u_0\|_A, \tag{13}$$

Proof. Introduce $T_k(x)$, the Chebyshev polynomial of degree k,

$$T_k(x) = \begin{cases} \cos(k \cdot \arccos x) & \text{if} \quad |x| \leq 1 \\ \cosh(k \cdot \operatorname{arccosh} x) & \text{if} \quad |x| \geq 1 \end{cases}$$

To show $T_k(x)$ is indeed a polynomial of x, we can denote by $\theta = \arccos x$ and use

$$(\cos\theta + \mathrm{i}\sin\theta)^k = (e^{\mathrm{i}\theta})^k = e^{\mathrm{i}k\theta} = \cos k\theta + \mathrm{i}\sin k\theta.$$

On the left hand side, the real part will contain $(\sin\theta)^{2\ell} = (1 - \cos^2\theta)^\ell = (1 - x^2)^\ell$ which is a polynomial of x. For $|x| \geq 1$, verification is similar.

Let $a = \lambda_{\min}(A)$ and $b = \lambda_{\max}(A)$. We use the transformation $x \mapsto \frac{b+a-2x}{b-a}$ to change the interval $[a, b]$ to $[1, -1]$ and can use Chebyshev polynomial to define a residual polynomial

$$p_k(x) = \frac{T_k((b + a - 2x)/(b - a))}{T_k((b + a)/(b - a))}.$$

The denominator is introduced to satisfy the condition $p_k(0) = 1$. For $x \in [a, b]$, the transformed variable

$$\left| \frac{b + a - 2x}{b - a} \right| \leq 1.$$

Hence the numerator is $\cos k\theta$ and $|\cos k\theta| \leq 1$ which leads to the bound

$$\inf_{\substack{p_k \in \mathcal{P}_k, \\ p_k(0)=1}} \sup_{\lambda \in \sigma(A)} |p_k(\lambda)| \leq \left[T_k \left(\frac{b + a}{b - a} \right) \right]^{-1}.$$

We set

$$\frac{b + a}{b - a} = \cosh\sigma = \frac{\mathrm{e}^{\sigma} + \mathrm{e}^{-\sigma}}{2}.$$

Solving this equation for e^{σ}, we have

$$\mathrm{e}^{\sigma} = \frac{\sqrt{\kappa(A)} + 1}{\sqrt{\kappa(A)} - 1}$$

with $\kappa(A) = b/a$. We then obtain

$$T_k \left(\frac{b + a}{b - a} \right) = \cosh(k\sigma) = \frac{\mathrm{e}^{k\sigma} + \mathrm{e}^{-k\sigma}}{2} \geq \frac{1}{2} \mathrm{e}^{k\sigma} = \frac{1}{2} \left(\frac{\sqrt{\kappa(A)} + 1}{\sqrt{\kappa(A)} - 1} \right)^{k},$$

which complete the proof. □

The estimate (13) shows that CG is in general better than the gradient method. Furthermore if the condition number of A is close to one, CG iteration will converge very fast. In some scenario, even if $\kappa(A)$ is large, the iteration will perform well if the majority of eigenvalues are clustered in a few small intervals.

Corollary 3.6. *Assume that $\sigma(A) = \sigma_0(A) \cup \sigma_1(A)$ and l is the number of elements in $\sigma_0(A)$. Then*

$$\|u - u_k\|_A \leq 2M \left(\frac{\sqrt{b/a} - 1}{\sqrt{b/a} + 1} \right)^{k-l} \|u - u_0\|_A, \tag{14}$$

where

$$a = \min_{\lambda \in \sigma_1(A)} \lambda,\ b = \max_{\lambda \in \sigma_1(A)} \lambda, \quad \textit{and}\ \ M = \max_{\lambda \in \sigma_1(A)} \prod_{\mu \in \sigma_0(A)} |1 - \lambda/\mu|.$$

Proof. Exercise. □

This result shows that if there are only few (say 2 or 3) small eigenvalues and others are well conditioned (in the sense that the so-called *effective condition number* b/a is not too large), then after few steps, the convergence rate of CG is governed by the effective condition number b/a.

4. Preconditioned Conjugate Gradient method

Let B be an SPD matrix. We shall apply the Gram-Schmidt process to the subspace

$$\mathbb{V}_k = \mathrm{span}\{Br_0, Br_1, \cdots, Br_k\},$$

to get another A-orthogonal basis. The three steps of CG are still the same except that we add Br_{k+1} instead of r_{k+1}.

(1) compute u_{k+1} by the A-orthogonal projection of $u - u_0$ to $\mathbb{V}_k$.

(2) add residual vector Br_{k+1} to $\mathbb{V}_k$ to get $\mathbb{V}_{k+1}$.
(3) apply Gram-Schmit process to get $A-$orthogonal vector p_{k+1} .

Starting from an initial guess u_0 and $p_0 = Br_0$, for $k = 0, 1, 2, \cdots$, we use three recrusive formulae to compute

$$
\begin{aligned}
u_{k+1} &= u_k + \alpha_k p_k, \qquad & \alpha_k &= \frac{(u-u_0, p_k)_A}{(p_k,p_k)_A} = \frac{(Br_k, r_k)}{(Ap_k, p_k)},\\
r_{k+1} &= r_k - \alpha_k A p_k, & & \\
p_{k+1} &= Br_{k+1} + \beta_k p_k, \qquad & \beta_k &= -\frac{(Br_{k+1}, p_k)_A}{(p_k,p_k)_A} = \frac{(Br_{k+1}, r_{k+1})}{(Br_k, r_k)}.
\end{aligned}
$$

We need to justify p_{k+1} computed above is A-orthogonal to $\mathbb{V}_k$. Proof of the following lemma is almost identical to that of Lemma 2.1 and 2.3.

Lemma 4.1. *Br_{k+1} is A-orthogonal to $\mathbb{V}_{k-1}$ and p_{k+1} is A-orthogonal to $\mathbb{V}_k$.*

Proof. Exercise. □

We present the algorithm of preconditioned conjugate gradient method as follows.

```
function u = pcg(A,b,u,B,tol)
tol = tol*norm(b);
k = 1;
r = b - A*u;
rho =  1;
while sqrt(rho) ≥ tol && k<length(b)
    Br = B*r;
    rho = r'*Br;
    if k = 1
        p = Br;
    else
        beta = rho/rho_old;
        p = Br + beta*p;
    end
    Ap = A*p;
    alpha = rho/(p'*Ap);
    u = u + alpha*p;
    r = r - alpha*Ap;
    rho_old = rho;
    k = k + 1;
end
```

Similarly we collect several remarks for the implementation

- If we think $A : \mathbb{V} \to \mathbb{V}'$, the residual is in the dual space $\mathbb{V}'$. B is the Reisz representation from $\mathbb{V}'$ to $\mathbb{V}$ of some inner product. In the original CG, B is the identity matrix.
- A general guiden principle for the design of preconditioners is to find the correct Hilbert space with the correct inner product such that the operator A is continuous and stable. Then the corresponding Riesz representation of that inner product is a good preconditioner.
- For an effective PCG, the computation `B*r` is crucial. The matrix B do not have to be formed explicitly. All we need is the matrix-vector multiplication which can be replaced by a subroutine.

To perform the convergence analysis, we let $\tilde{r}_0 = Br_0$. Then one can easily verify that

$$\mathbb{V}_k = \mathrm{span}\{\tilde{r}_0, BA\tilde{r}_0, \cdots, (BA)^k\tilde{r}_0\}.$$

Since the optimality

$$\|u - u_k\|_A = \inf_{v\in\mathbb{V}_{k-1}} \|u - (u_0 + v)\|_A.$$

still holds, and BA is symmetric in the A-inner product, we obtain the following convergence rate of PCG.

Theorem 4.2. *Let A be SPD and let u_k be the kth iteration in the PCG method with the preconditioner B and an initial guess u_0. Then*

$$\|u - u_k\|_A = \inf_{\substack{p_k\in\mathcal{P}_k,\\ p_k(0)=1}} \|p_k(BA)(u - u_0)\|_A, \tag{15}$$

$$\|u - u_k\|_A \le \inf_{\substack{p_k\in\mathcal{P}_k,\\ p_k(0)=1}} \sup_{\lambda\in\sigma(BA)} |p_k(\lambda)|\, \|u - u_0\|_A, \tag{16}$$

$$\|u - u_k\|_A \le 2\left(\frac{\sqrt{\kappa(BA)} - 1}{\sqrt{\kappa(BA)} + 1}\right)^k \|u - u_0\|_A. \tag{17}$$

The SPD matrix B is called *preconditioner*. A good preconditioner should have the properties that the action of B is easy to compute and that $\kappa(BA)$ is significantly smaller than $\kappa(A)$. The design of a good preconditioner requires the knowledge of the problem and finding a good preconditioner is a central topic of scientific computing.

For a linear iterative method Φ_B with a symmetric iterator B, the iterator can be used as a preconditioner for A. The corresponding PCG method converges at a faster rate. Let ρ denote the convergence rate of Φ_B, i.e. $\rho = \rho(I - BA)$. Then

$$\delta = \frac{\sqrt{\kappa(BA)} - 1}{\sqrt{\kappa(BA)} + 1} \le \frac{1 - \sqrt{1 - \rho^2}}{\rho} < \rho.$$

A more interesting fact is that the scheme Φ_B may not be convergent at all whereas B can always be a preconditioner. That is even $\rho > 1$, the PCG rate $\delta < 1$. For example, the Jacobi method is not convergent for all SPD systems, but $B = D^{-1}$ can always be used as a preconditioner, which is often known as the diagonal preconditioner. Another popular preconditioner is an incomplete Cholesky factorization.

5. GMRES

For a general matrix A, we can reformulated the equation $Au = b$ as a least square problem

$$\min_{u\in\mathbb{R}^N} \|b - Au\|,$$

and restrict the minimization in the affine subspace $u_0 + \mathbb{V}_k$, i.e.,

$$\min_{v\in u_0+\mathbb{V}_k} \|b - Av\|. \tag{18}$$

Let u_k be the solution of (18) and $r_k = b - Au_k$. Identical to the proof of CG, we immediately get

$$\|r_k\| = \min_{p\in\mathcal{P}_k, p(0)=1} \|p(A)r_0\| \le \min_{p\in\mathcal{P}_k, p(0)=1} \|p(A)\|\|r_0\|. \tag{19}$$

Unlike the case when A is SPD, now the norm $\|p(A)\|$ is not easy to estimate since the spectrum of a general matrix A includes complex numbers and much harder to estimate. Convergence analysis for diagonalizable matrices can be found in [2].

In the algorithmic level, the GMRES is also more expensive than CG. There are no recursive formula to update the approximation. One has to

(1) Compute and store an orthogonal basis of $\mathbb{V}_k$;
(2) Solve the least square problem (18).

The k orthogonal basis can be computed by Gram-Schmidt process requiring $\mathcal{O}(kN)$ memory and complexity. The least square problem can be solved in $\mathcal{O}(k^3)$ complexity. Plus one matrix-vector product with $\mathcal{O}(sN)$ operations, where s is the sparsity of the matrix (average number of non-zeros in one row). Therefore for m GMRES iterations, the total complexity is $\mathcal{O}(smN + m^2N + m^4)$.

As the iteration progresses, the term $m^2N + m^4$ increases quickly. In addition, it requires mN memory to store m orthogonal basis. GMRES will not be efficient for large m. One simple fix is the restart. Choose a upper bound for m, say, 20. After that, discard the previous basis and record new basis. The restart will lose the optimality (19) and in turn slow down the convergence since in (18) the space is changed.

A good preconditioner is thus more important for GMRES. Suppose we can choose a preconditioner B such that $\|I - BA\| = \rho < 1$. Then solving $BA\,u = Bb$ by GMRES. The estimate (19) will lead to the convergence

$$\|Br_k\| \leq \rho^k \|Br_0\|.$$

One can also solve $AB\,v = b$ by GMRES if $\|I - AB\| = \rho < 1$ and set $u = Bv$. These two are called left or right preconditioner, respectively. Again a general guiding principle of design a good principle is to find the correct inner product such that the operator A is continuous and stable and use the corresponding Reisz representation as the preconditioner.

REFERENCES

[1] M. Hestenes and E. Stiefel. Methods of conjugate gradients for solving linear systems. *Journal cf Research of the National Bureau of Standards Vol*, 49(6), 1952. 1

[2] C. Kelley. *Iterative methods for linear and nonlinear equations*. Society for Industrial Mathematics, 1995. 12

[3] Y. Saad and M. H. Schultz. GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM J. Sci. Statist. Comput.*, 7:856–869, 1986. 1