Abstract

The preconditioned conjugate gradient (PCG) method is an effective means for solving systems of linear equations where the coefficient matrix is symmetric and positive definite. The incomplete LDLᵀ factorizations are a widely used class of preconditionings, including the SSOR, Dupont-Kendall-Rachford, Generalized SSOR, ICCG(0), and MICCG(0) preconditionings. The efficient implementation of PCG with a preconditioning from this class is discussed.

Efficient Implementation of a Class of Preconditioned Conjugate Gradient Methods

Stanley C. Eisenstat

Research Report 1185

August 1980

This research was supported in part by ONR Grant NO0014-76-C-0277.
1. Introduction

Consider the system of $N$ linear equations

\[(1) \quad A x = b, \]

where the coefficient matrix $A$ is symmetric and positive definite. When $A$ is large and sparse, the preconditioned conjugate gradient (PCG) method is an effective means for solving (1) \([2, 4, 5, 9, 13]\). Given an initial guess $x_0$, we generate a sequence $\{x_k\}$ of approximations to the solution $x$ as follows:

\[(2a) \quad p_0 = r_0 = b - Ax_0\]

\[(2b) \quad \text{Solve } Mr_0 = r_0\]

\[\text{FOR } k = 0 \text{ STEP 1 UNTIL Convergence DO}\]

\[(2c) \quad a_k = (r_k, r_k^\ast) / (p_k, Ap_k)\]

\[(2d) \quad x_{k+1} = x_k + a_k p_k\]

\[(2e) \quad r_{k+1} = r_k - a_k Ap_k\]

\[(2f) \quad \text{Solve } Mr_{k+1} = r_{k+1}\]

\[(2g) \quad b_k = (r_{k+1}, r_{k+1}^\ast) / (r_k, r_k^\ast)\]

\[(2h) \quad p_{k+1} = r_{k+1} + b_k p_k\]

The effect of the preconditioning matrix $M$ is to increase the rate of convergence of the basic conjugate gradient method of Hestenes and Stiefel \([11]\). The number of multiply-adds per iteration is just $5N$, plus the number required to form $Ap_k$, plus the number required to solve $Mr_k = r_k$. 
One widely used class of preconditionings are the incomplete LDL$^t$

factorizations

$$M = (\tilde{D}+L) \tilde{D}^{-1} (\tilde{D}+L)^t,$$

where $A = L+D+L^t$, $L$ is strictly lower triangular, and $D$ and $\tilde{D}$ are positive diagonal. This class includes the SSOR [9], Dupont-Kendall-Rachford [7], Generalized SSOR [1], ICCG(0) [13], and MlCGG(0) [10] preconditionings. Letting $\text{NZ}(A)$ denote the number of nonzero entries in the matrix $A$, a straight-forward implementation of PCG with a preconditioning from this class\(^1\) would require $6N + 2\text{NZ}(A)$ multiply-adds per iteration.\(^2\)

In this brief note, we show how to reduce the work to $8N + \text{NZ}(A)$ multiply-adds, asymptotically half as many as the straight-forward implementation.\(^3\) We give details in Section 2, and consider some generalizations in Section 3.

2. Implementation

The linear system (1) can be restated in the form

\(^1\) Writing $M$ as $(\tilde{D}+L)(I+\tilde{D}^{-1}L^t)$, we solve $Mr'_k = r_k$ by solving the triangular systems $(\tilde{D}+L)t_k = r_k$, $(I+\tilde{D}^{-1}L^t)\tilde{t}_k = t_k$.

\(^2\) $2N$ (respectively, $N$) multiply-adds can be saved by symmetrically scaling the problem to make $\tilde{D} = I$ (respectively, $D = I$).

\(^3\) A similar speedup for pairs of linear iterative methods is given in [6].
\[(4) \quad [(\hat{D}+L)^{-1} A (\hat{D}+L)^{-t}] (\hat{D}+L)^t x = [(\hat{D}+L)^{-1} b]
\]
or

\[(5) \quad \hat{A} \hat{x} = \hat{b}.
\]

But applying PCG to (1) with \(M = (\tilde{D}+L)(\tilde{D}+L)^t\) is equivalent to applying PCG to (5) with \(\tilde{M} = \tilde{D}^{-1}\) and setting \(x = (\tilde{D}+L)^{-t}\hat{x}\). If we update \(x\) instead of \(\hat{x}\) at each iteration, algorithm (2) becomes:

\[\begin{align*}
\hat{p}_0 &= \hat{r}_0 = \hat{b} - \hat{A}x_0 \\
\text{(6b) Compute } \hat{r}_0 &= \hat{D}r_0 \\
\text{FOR } k = 0 \text{ STEP 1 UNTIL Convergence DO} \\
\hat{a}_k &= (\hat{r}_k, \hat{r}_k) / (\hat{p}_k, \hat{A}\hat{p}_k) \\
x_{k+1} &= x_k + \hat{a}_k (\tilde{D}+L)^{-t}\hat{p}_k \\
\hat{r}_{k+1} &= \hat{r}_k - \hat{a}_k \hat{A}\hat{p}_k \\
\text{(6g) Compute } \hat{r}_{k+1} &= \hat{D}\hat{r}_{k+1}
\end{align*}\]

Both are equivalent to applying the basic conjugate gradient method to the preconditioned system

\[\bar{x} = [\tilde{D}^{1/2}(\tilde{D}+L)^{-1} A (\tilde{D}+L)^{-t}\tilde{D}^{1/2}] [\tilde{D}^{1/2}(\tilde{D}+L)^t x] = [\tilde{D}^{1/2}(\tilde{D}+L)^{-1} b] = \tilde{b}
\]
(see [4], pp. 58-59).
\[ \hat{b}_k = \left( \hat{r}_{k+1}, \hat{r}_{k+1} \right) / (\hat{r}_k, \hat{r}_k) \]

\[ \hat{p}_{k+1} = \hat{r}_{k+1} + \hat{b}_k \hat{p}_k \]

\[ \hat{A} \hat{p}_k \]

\( \hat{A} \hat{p}_k \) can be computed efficiently by taking advantage of the following identity:

\[ \hat{A} \hat{p}_k = (\hat{D} + \hat{L})^{-1} \left[ (\hat{D} + \hat{L}) + (\hat{D} + \hat{L})^t - (2\hat{D} - \hat{D}) \right] (\hat{D} + \hat{L})^{-t} \hat{p}_k \]

\[ = (\hat{D} + \hat{L})^{-t} \hat{p}_k + (\hat{D} + \hat{L})^{-1} [\hat{p}_k - K(\hat{D} + \hat{L})^{-t} \hat{p}_k] \]

where \( K = 2\hat{D} - \hat{D} \). Thus

\[ \hat{c}_k = (\hat{D} + \hat{L})^{-t} \hat{p}_k \]

\[ \hat{A} \hat{p}_k = \hat{c}_k + (\hat{D} + \hat{L})^{-1} (\hat{p}_k - K \hat{c}_k) \]

which requires \( 2N + NZ(A) \) multiply-adds. \( \hat{c}_k \) can also be used to update \( x_k \) in (6d), so that the total cost for each PCG iteration is just \( 8N + NZ(A) \) multiply-adds,\(^5\) versus \( 6N + 2NZ(A) \) for the straight-forward implementation.

3. Generalizations

The approach presented in Section 2 extends immediately to preconditionings of the form

\[ 5 \] Again, \( 3N \) multiply-adds can be saved by symmetrically scaling the problem so that \( \hat{D} = I \).
where $\tilde{S}$ is positive diagonal. Moreover, if we take $K = \tilde{D} + \tilde{D}^T - D$ in (7) and (8), then $\tilde{D}$ need not be diagonal or even symmetric. In this case, $\tilde{D}$ would reflect changes to both the diagonal and off-diagonal entries of $A$ in generating an incomplete factorization. If we assume that only the nonzero entries of $A$ are changed, i.e., that $(K)_{ij}$ is nonzero only if $(A)_{ij}$ is nonzero, then the operation count is $7N + NZ(A) + NZ(K)$.

Another application is to preconditioning nonsymmetric systems. Let

\[(10) \quad M = (\tilde{D} + L) \tilde{S}^{-1} (\tilde{D} + U),\]

be an incomplete LDU factorization of a nonsymmetric matrix $A$, where $A = L + D + U$, $L$ (respectively, $U$) is strictly lower (respectively, upper) triangular, and $D$ and $\tilde{S}$ are diagonal. Then a number of authors have proposed solving the linear system $Ax = b$ by solving the normal equations for one of the preconditioned systems

\[(11a) \quad \hat{A}_1 \hat{x} = [(\tilde{S} (\tilde{D} + L)^{-1} A (\tilde{D} + U)^{-1}) [(\tilde{D} + U) x] = [\tilde{S} (\tilde{D} + L)^{-1} b] = \hat{b}\]

(see [12]) and

\[(11b) \quad \hat{A}_2 \hat{x} = [(\tilde{D} + U)^{-1} \tilde{S} (\tilde{D} + L)^{-1} A] x = [(\tilde{D} + U)^{-1} \tilde{S} (\tilde{D} + L)^{-1} b] = \hat{b}\]

(see [14, 3]). $\hat{A}_2 \hat{p}$ can be computed as

\[(12) \quad \hat{A}_2 \hat{p} = (\tilde{D} + U)^{-1} \tilde{S} [\hat{p} + (\tilde{D} + L)^{-1} (D + U - \tilde{D}) \hat{p}]\]

in $4N + NZ(L) + 2NZ(U)$ multiply-adds, whereas $\hat{A}_1 \hat{p}$ can be computed as
(13a) \[ \hat{c} = (\hat{D} + U)^{-1} p \]

(13b) \[ \hat{A}_1 \hat{\delta} = \delta \left[ \hat{c} + (\hat{D} + L)^{-1} (\hat{p} - (2\hat{D} - D)\hat{c}) \right] \]

in \(4N+NZ(L)+NZ(U)\) multiply-adds. Thus the first approach would be more efficient per iteration, although more iterations might be required to achieve comparable accuracy [14].

REFERENCES

A generalized SSOR method.

On preconditioning and convergence acceleration in sparse matrix problems.

Conjugate gradient type methods for unsymmetric and inconsistent systems of linear equations.

Conjugate Gradient Methods for Partial Differential Equations.

A generalized conjugate gradient method for the numerical solution of elliptic partial differential equations.

---

6 The same would be true if a Generalized Conjugate Residual method such as Orthomin [15, 8] were used to solve (11a) or (11b).
Alternating methods for sets of linear equations.

An approximate factorization procedure for solving self-adjoint elliptic difference equations.

Solving approximations to the convection diffusion equation.

The analysis and application of sparse matrix algorithms in the finite element method.

[10] Ivar Gustafsson.
A class of first order factorization methods.

Methods of conjugate gradients for solving linear systems.

The incomplete Cholesky–conjugate gradient method for the solution of systems of linear equations.

An iterative solution method for linear systems of which the coefficient matrix is a symmetric M-matrix.

An ILUCG Algorithm Which Minimizes in the Euclidean Norm.

Orthomin, an iterative method for solving sparse sets of simultaneous linear equations.