A deflated version of the Conjugate Gradient Algorithm

Y. Saad M. Yeung J. Erhel F. Guyomarc'h

March 8, 1998

Abstract

We present a deflated version of the conjugate gradient algorithm for solving linear systems. The new algorithm can be useful in cases when there are a small number of eigenvalues of the iteration matrix which are very close to the origin. It can also be useful when solving linear systems with multiple right-hand sides, since the eigenvalue information gathered from solving one linear system can be recycled for solving the next systems and then updated.

Key words: Conjugate Gradient; deflation; multiple right-hand sides; Lanczos algorithm.

1 Introduction

A number of recent articles have estabished the benefits of using eigenvalue deflation when solving nonsymmetric linear systems with Krylov subspace methods. It has been observed that significant improvements in convergence rates can be achieved from Krylov subspace methods by adding to these subspaces a few approximate eigenvectors associated with the eigenvalues closest to zero [3, 4, 6, 7, 11]. In practice, approximations to the eigenvectors closest to zero are obtained from the use of a certain Krylov subspace, then these approximations are dynamically updated using the new Krylov subspace. Results of experiments obtained from these variations indicate that the improvement in convergence over standard Krylov subspaces of the same dimension can sometimes be substantial, especially when the convergence of the original scheme is hampered by a small number of eigenvalues near zero, see e.g., [3, 7].

In this paper we consider extensions of this idea to the Conjugate Gradient algorithm for the symmetric case. Our starting point is an algorithm recently proposed by Erhel and Guyomarc'h [5] and which is an augmented subspace Conjugate Gradient method aimed at linear systems with several right-hand sides. Erhel and Guyomarc'h [5] propose an algorithm which adds one specific vector obtained from a subspace related to a previous right-hand side. We first extend this algorithm to one which handles an arbitrary block W of vectors.

In the second part of the paper, we apply this technique to the situation when the block W of added vectors is a set of approximate eigenvectors. This, in turn is used for solving linear linear systems with sequential multiple right-hand sides.

2 The Deflated Lanczos Algorithm

Consider a symmetric positive definite (SPD) matrix $A \in \mathbf{R}^{n\times n}$ and let k real vectors $w_1, w_2, \cdots, w_k$ be given along with a unit vector v_1 that is orthogonal to w_i for $i = 1, 2, \cdots, k$. Define $W = [w_1, w_2, \cdots, w_k]$. We assume that $[w_1, w_2, \cdots, w_k]$ is a set of linearly independent vectors. Since A is SPD, the matrix $W^T AW$ is then non singular.

The Deflated Lanczos Algorithm builds a sequence $\{v_j\}_{j=1,2,\cdots}$ of vectors such that

$$v_{j+1} \perp span\{W, v_1, v_2, \cdots, v_j\} \tag{1}$$

and

$$\|v_{j+1}\|_2 = 1. \tag{2}$$

The construction of such a sequence can be carried out with a $(k+3)$-term recurrence. Assume that the vectors $v_2, v_3, \cdots, v_j$ are available. To obtain v_{j+1}, we consider the expression

$$\begin{aligned} \sigma_{j+1} v_{j+1} &= Av_j - \gamma_{j-1} v_{j-1} - \rho_j v_j - \sum_{i=1}^{k} \nu_{i,j} A w_i \\ &= Av_j - \gamma_{j-1} v_{j-1} - \rho_j v_j - AW\hat{\nu}_j, \end{aligned} \tag{3}$$

where $\gamma_0 v_0 = 0$ and $\hat{\nu}_j := [\nu_{1,j},\, \nu_{2,j},\, \cdots,\, \nu_{k,j}]^T$ is a vector of length k. The coefficients $\rho_j, \sigma_{j+1}, \gamma_{j-1}$ and $\hat{\nu}_j$ are determined by properties (1) and (2) as follows,

$$\begin{array}{ll} W^T AW\hat{\nu}_j = W^T Av_j, & \rho_j = v_j^T Av_j - v_j^T AW\hat{\nu}_j, \\ \gamma_{j-1} = v_{j-1}^T Av_j - v_{j-1}^T AW\hat{\nu}_j, & \sigma_{j+1} = \pm\|Av_j - \gamma_{j-1}v_{j-1} - \rho_j v_j - AW\hat{\nu}_j\|_2. \end{array} \tag{4}$$

We always select σ_{j+1} to be positive.

Relations (3) for $j = 1, \ldots, m$ can be rewritten in matrix form. Setting

$$V_m = [v_1,\, v_2,\, \cdots,\, v_m], \qquad \bar{T}_m = \begin{bmatrix} \nu_{11} & \nu_{12} & \nu_{13} & \cdots & \nu_{1m} \\ \vdots & \vdots & \vdots & & \vdots \\ \nu_{k1} & \nu_{k2} & \nu_{k3} & \cdots & \nu_{km} \\ \rho_1 & \gamma_1 & 0 & \cdots & 0 \\ \sigma_2 & \rho_2 & \gamma_2 & \ddots & \vdots \\ 0 & \sigma_3 & \rho_3 & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & \gamma_{m-1} \\ 0 & \cdots & 0 & \sigma_m & \rho_m \end{bmatrix} = \begin{bmatrix} \Delta_m \\ T_m \end{bmatrix},$$

where Δ_m denotes the submatrix consisting of the first k rows of $\bar{T}_m$ and the submatrix T_m is obtained by deleting the first k rows of $\bar{T}_m$, then we have

$$AV_m = [\,AW,\, V_m\,]\,\bar{T}_m + \sigma_{m+1} v_{m+1} e_m^T,$$

where e_m is the last column of the $m \times m$ indentity matrix I_m. It can be proved that the submatrix T_m is symmetric. Thus $\sigma_{j+1} = \gamma_j$ and therefore we do not need to compute both σ_j's and γ_j's. Putting relations (3) and (4) together gives the following algorithm.

ALGORITHM 1. **Deflated Lanczos Algorithm**

1. *Choose k vectors $w_1, w_2, \cdots, w_k$. Define $W = [w_1,\, w_2,\, \cdots,\, w_k]$.*
2. *Choose an initial vector v_1 such that $v_1^T W = 0$ and $\|v_1\|_2 = 1$.*
3. *Set $\sigma_1 v_0 = 0$.*
4. *For $j = 1, 2, \cdots$,*
5. *Solve $W^T AW\hat{\nu}_j = W^T Av_j$ for $\hat{\nu}_j$;*
6. $z_j = Av_j - AW\hat{\nu}_j$;
7. $\rho_j = v_j^T z_j$;
8. $\hat{v}_{j+1} = z_j - \sigma_j v_{j-1} - \rho_j v_j$;
9. $\sigma_{j+1} = \|\hat{v}_{j+1}\|_2$; *If $\sigma_{j+1} = 0$ then stop*;
10. $v_{j+1} = \hat{v}_{j+1}/\sigma_{j+1}$;
11. *End*

3 The Deflated-CG Algorithm

We now turn to the linear system

$$Ax = b, \tag{5}$$

where A is SPD. Based on the deflated Lanczos procedure described in section 2, we wish to derive an oblique projection method, following a standard technique used in the derivation of the Conjugate Gradient algorithm [10, p.179] from the standard Lanczos procedure.

Here, the w_i's and v_j's are as defined in the deflated Lanczos procedure and we use the notations W and V_j introduced in Section 2. Assume an initial guess x_0 to (5) is given such that $r_0 := b - Ax_0 \perp W$. We set $v_1 = r_0/\|r_0\|_2$. Define

$$\mathcal{K}_{k,j}(A, W, r_0) = span\{W, V_j\}.$$

At the j-th step of our projection method, we seek an approximate solution x_j with

$$x_j \in x_0 + \mathcal{K}_{k,j}(A, W, r_0) \quad and \quad r_j = b - Ax_j \perp \mathcal{K}_{k,j}(A, W, r_0). \tag{6}$$

As in the standard CG case, it can be showed that x_i and r_j with properties (6) satisfy

$$x_j = x_0 + \|r_0\|_2 \left(-W\Delta_j + V_j\right) T_j^{-1} e_1$$

and

$$r_j = c_j v_{j+1}$$

for some scalar c_j, where e_1 is the first column of the $j \times j$ identity matrix. Thus $\mathcal{K}_{k,j}(A, W, r_0) = span\{W, r_0, r_1, \cdots, r_{j-1}\}$.

Let $T_j = L_j D_j L_j^T$ be the LDL^T decomposition of the symmetric matrix T_j. Define

$$P_j \equiv [p_0,\, p_1,\, \cdots,\, p_{j-1}] = \left(-W\Delta_j + V_j\right) L_j^{-T} \Lambda_j$$

where $\Lambda_j = diag\{c_0, c_1, \cdots, c_{j-1}\}$.

Similarly to the standard CG case, it can be proved that the solution x_j, the residual r_j and the descent direction p_j satisfy the recurrence relations

$$x_j = x_{j-1} + \alpha_{j-1} p_{j-1}, \qquad r_j = r_{j-1} - \alpha_{j-1} A p_{j-1}, \qquad p_j = \beta_{j-1} p_{j-1} + r_j - W\hat{\mu}_j \tag{7}$$

for some $\alpha_{j-1}, \beta_{j-1}, \hat{\mu}_j$. Moreover, the vectors p_j are A-orthogonal to each other: $P_j^T A P_j$ is diagonal. In addition, they are also A-orthogonal to all w_i: $W^T A P_j = 0$.

By using the orthogonality of r_j's and the A-orthogonality of p_j's, the coefficients in (7) can be expressed via the vectors W, r_j and p_j as follows,

$$\alpha_j = \frac{r_j^T r_j}{p_j^T A p_j}, \qquad \hat{\mu}_j = \left(W^T A W\right)^{-1} W^T A r_j, \qquad \beta_j = \frac{r_{j+1}^T r_{j+1}}{r_j^T r_j}. \tag{8}$$

Putting relations (7) and (8) together gives the following algorithm.

ALGORITHM 2. **Deflated-CG**

1. *Choose k linearly independent vectors $w_1, w_2, \cdots, w_k$. Define $W = [w_1, w_2, \cdots, w_k]$.*
2. *Choose an initial guess x_0 such that $W^T r_0 = 0$, where $r_0 = b - Ax_0$.*
3. *Solve $W^T A W \hat{\mu}_0 = W^T A r_0$ for $\hat{\mu}_0$ and set $p_0 = -W\hat{\mu}_0 + r_0$.*
4. *For $j = 1, 2, \cdots,$*
5. $\quad \alpha_{j-1} = r_{j-1}^T r_{j-1} / p_{j-1}^T A p_{j-1};$

6. $x_j = x_{j-1} + \alpha_{j-1} p_{j-1}$;

7. $r_j = r_{j-1} - \alpha_{j-1} A p_{j-1}$;

8. $\beta_{j-1} = r_j^T r_j / r_{j-1}^T r_{j-1}$;

9. *Solve* $W^T A W \hat{\mu}_j = W^T A r_j$;

10. $p_j = -W\hat{\mu}_j + \beta_{j-1} p_{j-1} + r_j$;

11. *End*

To guarantee that the initial guess x_0 satisfies $W^T r_0 = 0$, we can choose

$$x_0 = x_{-1} + W \left(W^T A W\right)^{-1} W^T r_{-1}, \tag{9}$$

where x_{-1} is arbitrary and $r_{-1} := b - Ax_{-1}$. This choice was used in [5, 8, 9, 12].

When W is a null matrix, Algorithm 2 reduces to the standard CG algorithm, see, for instance, [10, p.247]. In addition to the matrices A, four vectors and three matrices of storage are required: p, Ap, r, x, W, AW and $W^T AW$.

4 Theoretical Considerations

The convergence behavior of the Deflated-CG algorithm can be analyzed by exploiting the theory developed in [5]. Let $\mathbf{W}$ be the subspace spanned by the columns of W in Algorithm 2 and let $\{ w_1^\perp, w_2^\perp, \cdots, w_{n-k}^\perp \}$ be an orthonormal basis of its orthogonal complement $\mathbf{W}^\perp$. Define

$$W^\perp = \left[w_1^\perp, w_2^\perp, \cdots, w_{n-k}^\perp \right] \quad \textit{and} \quad H = I - W \left(W^T A W\right)^{-1} W^T A.$$

We observe that the residuals r_j produced by the Deflated-CG algorithm lie in the subspace $\mathbf{W}^\perp$ since $W^T r_j = 0$ by construction. If we define $\tilde{r}_j = \left(W^\perp\right)^T r_j$, then $\tilde{r}_j$'s are the residual vectors produced by the standard CG applied to the system

$$\tilde{A}\tilde{x} = \tilde{b}$$

with the initial guess $\tilde{x}_0 = \left(W^\perp\right)^T x_0$, where $\tilde{A} = \left(W^\perp\right)^T H^T A H W^\perp$ and $\tilde{b} = \left(W^\perp\right)^T H^T b$. Based on this observation, it can be proved that the approximate solution obtained at the jth step of the Deflated-CG algorithm satisfys

$$\|x_* - x_j\|_A \le 2 \left(\frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1} \right)^j \|x_* - x_0\|_A, \tag{10}$$

where x_* is the exact solution of (5) and κ is the condition number of $\tilde{A}$. This result can also be got by directly applying the polynomial formalism built in [5].

Deflated-CG can also be viewed as a Preconditioned Conjugate Gradient (PCG) but with a singular preconditioner. Define

$$C = HH^T.$$

We use the taxonomy defined in [1], where two versions of PCG are denoted by $Omin(A, C, A)$ and $Odir(A, C, A)$. Then Deflated-CG is equivalent to the version $Odir(A, C, A)$ of PCG applied to A with the preconditioner C and started with $r_0 \perp W$.

We now get the same convergence rate as previously. Indeed, the version $Odir(A, C, A)$ does not break down, converges with an asymptotic rate governed by κ and given by (10). See for example [10, p.194].

5 Systems with multiple dependent right-hand sides

In this section, we present a method which applies the Deflated-CG algorithm to the solution of several symmetric linear systems of the form

$$Ax^{(s)} = b^{(s)}, \qquad s = 1, 2, \cdots, \nu, \tag{11}$$

where $A \in \mathbf{R}^{n\times n}$ is SPD and where the different right-hand sides $b^{(s)}$ depend on the solutions of their previous systems. This problem has been recently considered by Erhel and Guyomarc'h [5]. The main idea in [5] is to solve the first system by CG and to recycle the Krylov subspace $K_m(A, x_0)$ created in the first system to accelerate the speeds of convergence in the subsequent systems solving. The disadvantage of this approach is that the memory requirements could be huge when m is large [2] since it requires keeping a basis of an earlier Krylov subspace $K_m(A, x_0)$. An alternative whose goal is to maintain a similar convergence rate, is to adopt the idea of eigenvalue deflation, as used in the Deflated-GMRES algorithm, for example, see [3, 4, 7]. Deflated GMRES improves the matrix condition number by injecting the desired approximate eigenvectors into its Krylov solution subspace. These approximate eigenvectors are usually selected to be those hampering the convergence of the original scheme and are refined in every restart stage. Imitating the approach of Deflated-GMRES, we add some approximate eigenvectors, usually corresponding to eigenvalues nearest zero, to the Krylov solution subspace when we solve each, but the first, system of (11). The eigenvectors are refined every time we solve a system of (11) so that they get closer and closer to their corresponding real ones. In this way, we may expect that the convergence will be accelerated. The memory requirements for this approach are fairly low, since only a small number of eigenvectors, typically 4 to 10, and some data obtained in the earlier system solving are required. The main algorithm of our method is the following.

ALGORITHM 3. **Deflated-CG for Dependent Multiple Right-hand Sides**

1. *Given integer k. Let $\lambda_1, \lambda_2, \cdots, \lambda_k$ be the k smallest eigenvalues of A.*
2. *Choose an integer $l \geq k$.*
3. *Solve the first system of (11) with the CG algorithm [10, p.179], and keep the data in the first l steps.*
4. *Compute k approximate eigenvectors $w_1, w_2, \cdots, w_k$ associates with $\lambda_1, \lambda_2, \cdots, \lambda_k$ respectively through the data we have kept and set $W^{(2)} = [w_1, w_2, \cdots, w_k]$.*
5. *For $s = 2, 3, \cdots, \nu$*
6. *Choose $l \geq 0$.*
7. *Solve the s^{th} system of (11) by the Deflated-CG algorithm with $W = W^{(s)}$. Keep the data in the first l steps.*
8. *Refine $W^{(s)}$ via the data we kept at Step 7 and set $W^{(s+1)}$ to be the matrix consisting of these refined approximate eigenvectors.*
9. *EndFor*

Three projection techniques are described in [3] to obtain $W^{(s)}$. The one called *harmonic* is often the best in finding eigenvalues nearest zero. Let l linearly independent vectors $\{z_1, z_2, \cdots, z_l\}$ be given. The *harmonic* projection method extracts the k desired approximate eigenvectors by solving the generalized eigenproblem

$$Gy - \theta F y = 0, \qquad W = Z\,y,$$

where $Z := [z_1, z_2, \cdots, z_l]\,, G = (AZ)^T AZ$ and $F = Z^T AZ$. In our case, we set at each system

$$W^{(1)} = \emptyset, \qquad Z^{(s)} = [W^{(s)}, P_l^{(s)}], \qquad W^{(s+1)} = Z^{(s)} Y^{(s)}, \tag{12}$$

with $P_l^{(s)} = [p_0^{(s)}, p_1^{(s)}, \cdots, p_{l-1}^{(s)}]$ and $Y^{(s)} = [y_1^{(s)}, \ldots, y_k^{(s)}]$. Each vector $y_i^{(s)}$ is the solution of the generalized eigenvalue problem

$$G^{(s)} y - \theta_i F^{(s)} y = 0, \tag{13}$$

where $\theta_1, \ldots, \theta_k$ are the k smallest Ritz values.

The convergence rate of Algorithm 3 can be estimated from section 4. We label all the eigenvalues of A in increasing order $\lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$. In the special case where the column vectors, $w_1, w_2, \cdots, w_k$, of W are exact eigenvectors of A associated with the smallest eigenvalus $\lambda_1, \lambda_2, \cdots, \lambda_k$, it can be shown that $\kappa = \lambda_n/\lambda_{k+1}$. When the column vectors of W are not exact but near eigenvectors associated with $\lambda_1, \lambda_2, \cdots, \lambda_k$, one may expect that $\kappa \approx \lambda_n/\lambda_{k+1}$. Thus, Algorithm 3 is approximately solving a system with improved conditionning when s is large.

6 Practical considerations

Here we take a look at the memory and computational costs of the algorithm. It is assumed that $k \ll n$ so that the costs are dominated by terms that are functions of n.

In Deflated-CG, if we use the harmonic eigenvalue approach as was described earlier, we need to store W and AW or $2k$ vectors of length n each. Deflated-CG requires to compute Hr_j n Line 10 at each iteration. The time complexity is due to the BLAS2 operations $z_j = (AW)^T r_j$ and $r_j - W(W^T AW)^{-1} z_j$ which are $O(kn)$ so that CPU overhead for this is not too high.

In Algorithm 3, we compute W approximate eigenvectors using the generalized eigenproblem (13). Algorithm 3 requires to store not only W and AW but also l vectors $P_l^{(s)}$, amounting to $2k + l$ vectors of length n. There exist special schemes to compute W, as defined by (12) and (13), and AW which mainly consist of BLAS3 operations of complexity $O(n(l + 1)k)$. The computation of $W^T(AW)$ is a BLAS3 operation of complexity $O(nk^2)$. If k and l are kept small, the overhead is not too high.

One advantage of Algorithm 3 is that we can choose k small to save memory with no loss in efficiency. If l is too large, we must rely on secondary storage or choose l small except in the first system. Another advantage of the algorithm is that we can easily refine W when handling several right-hand sides and improve the convergence of each new system with a constant memory overhead.

7 Numerical Experiments

All the experiments were performed in MATLAB with machine precision 10^{-16}. The stopping criterion is that the true relative residual $\|b - Ax_j\|_2/\|b\|_2$ is less than 10^{-7}.

The following examples demonstrate the efficiency of Algorithm 3 when applied to (11). We always chose the initial guess $x_0 = 0$ in solving the first system and $x_{-1} = 0$ in (9) for the remaining systems solving. The number of the right-hand sides of (11) was 10 and $b^{(s)}$'s were independent random vectors with iid entries from $N(0, 1)$. Also, we kept the data in the first $l = 20$ steps and $k = 5$ approximate eigenvectors associated with smallest eigenvalues were calculated. All the test matrices were from the Harwell-Boeing Collection.

We observe that the number of iterations decreases significantly after the first few systems solving and quickly tends to its lower bound. It is because the approximate eigenvectors we chose quickly approach to their corresponding limits in the first few rounds of refinements. When these approximate eigenvectors have arrived at their limits, the number of iterations no longer decreases. In practice, we may omit step 8 or vary k and l when we find that the approximate eigenvectors are no longer improved during running Algorithm 3. But it is still an open problem for us how to define a criterion to characterise the so-called unimprovements of the approximate eigenvectors.

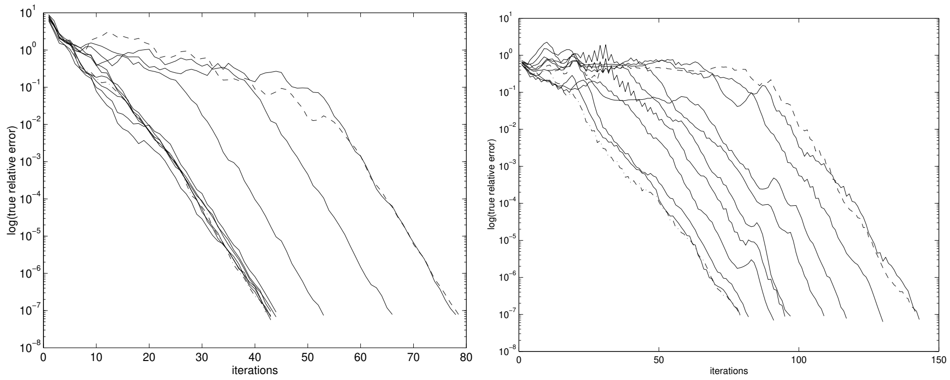

Figure 1: Solid: convergence histories of systems one to nine; dashdot: convergence history of system ten; dashed: convergence history of system ten solved by the standard preconditioned CG. (a) The matrix A is the second matrix named BCSSTK15 from the BCSSTRUC2 group of the Harwell-Boeing Collection. $ic(1)$ preconditioner was used. (b) The matrix A is the forth matrix named 1138BUS from the PSADMIT group. $ic(0)$ preconditioner was used.

References

[1] S. F. Ashby, T. A. Manteuffel and P. E. Saylor, *A taxonomy for conjugate gradient methods,* SIAM J. Numer. Anal., 27 (1990), pp. 1542-1568.

[2] M. O. Bristeau and J. Erhel, *Augmented Conjugate Gradient. Application in an iterative process for the solution of scattering problems,* http://www.irisa.fr/aladin/perso/erhel.html, 1998.

[3] A. Chapman and Y. Saad, *Deflated and augmented Krylov subspace techniques,* Numer. Linear Algebra Appl., to appear.

[4] J. Erhel and K. Burrage, *On the performance of various adaptive preconditioned GMRES strategies.* Research report 1081, INRIA, Feb. 1997.

[5] J. Erhel and F. Guyomarc'h, *An augmented subspace Conjugate Gradient,* http://www.irisa.fr /aladin/perso/erhel.html, 1997.

[6] S. A. Kharchenko and A. Yu. Yeremin, *Eigenvalue translation based preconditioners for the GMRES(k) method,* Numer. Linear Algebra Appl., 2 (1995), pp. 51-70.

[7] R. B. Morgan, *A restarted GMRES method augmented with eigenvectors,* SIAM J. Matrix Analysis and Applications, 16(4), 1154-1171, 1995.

[8] B. Parlett, *A new look at the Lanczos algorithm for solving symmetric systems of linear equations,* Linear algebra and its applications, 29 (1980), pp. 323-346.

[9] Y. Saad, *On the Lanczos method for solving symmetric linear systems with several right-hand sides,* Mathematics of computation, 178 (1987), pp. 651-662.

[10] ——, *Iterative methods for sparse linear systems,* PWS Publishing Company, 1996.

[11] ——, *Analysis of augmented Krylov subspace methods,* SIAM J. Matrix Anal. Appl., Vol. 18, No. 2, pp. 435-449, April 1997.

[12] H. van der Vorst, *An iterative method for solving $f(A)x = b$ using Krylov subspace information obtained for the symmetric positive definite matrix A,* Journal of Computational and Applied Mathematics, 18 (1987), pp. 249-263.