Universiteit Utrecht

Mathematics Institute

Deliberate Ill-Conditioning of Krylov Matrices

by

Jan Brandts

Preprint

No. 1181

February, 2001

Also available at URL `http://www.math.uu.nl/people/brandts`

Deliberate Ill-Conditioning of Krylov Matrices

Jan Brandts*

February, 2001

Abstract

This paper starts off with studying simple extrapolation methods for the classical iteration schemes such as Richardson, Jacobi and Gauss-Seidel iteration. The extrapolation procedures can be interpreted as approximate minimal residual methods in a Krylov subspace. It seems therefore logical to consider, conversely, classical methods as pre-processors for Krylov subspace methods, as was done by Zítko (1996) for the Conjugate Gradient method.

The observation made by Ipsen (1998) that small residuals necessarily imply an ill-conditioned Krylov matrix, explains the success of such pre-processing schemes: residuals of classical methods are (unscaled) power method iterates, and building a Krylov subspace on such a classical residual will therefore lead to expansion vectors that are at small angle to the previous Krylov vectors. This results in an ill-conditioned Krylov matrix. In this paper, we present a large number of experiments that support this claim, and give theoretical interpretations of the pre-processing.

The results are mainly of interest in Krylov subspace methods for non-Hermitian matrices based on long recurrences, and in particular for applications with heavy memory limitations. Also, in applications in which minimal residual methods stagnate due to a lack of ill-conditioning, the use of a classical pre-processor can be a cheap and easily parallelizable remedy.

1 Introduction

Among the iterative methods for solving large and sparse linear systems of equations, the Krylov subspace methods [4] are very popular. For special systems (i.e., positive definite and/or Hermitian matrices), elegant short recurrences lead to methods like Conjugate Gradients (CG), Minimal Residuals (MinRes), Conjugate Residuals (CR) and Symmetric LQ (SYMMLQ). Their generalizations, respectively the Full Orthogonalization Method (FOM), Generalized MR (GMRES) and Generalized CR (GCR) for non-Hermitian systems, use however necessarily long recursions for building orthogonal bases of the Krylov subspace, which, for large systems, becomes the more (and very) expensive as the number of iterations increases. Therefore, it is important to find ways, for example by suitable preconditioning, to keep this number as small as possible. If one is willing to sacrifice the minimization properties of the methods, one could consider using the bi-orthogonal Petrov-Galerkin approach

*Mathematical Institute, Utrecht University, P.O.Box 80.010, 3508 TA, Utrecht, The Netherlands. E-mail: brandts@math.uu.nl

leading to methods with short recurrences like BiCG, CGS and QMR. One could also restart or truncate the method. Either way it is of interest to build Krylov subspaces that contain good approximations of the initial residuals.

1.1 Aiming for ill-conditioned Krylov matrices

In this paper we investigate the effect of aiming the initial residual r_0 of a Krylov subspace method in the direction of a dominant eigenvector of the preconditioned system matrix $B := AK^{-1}$ through application of the K-preconditioned Richardson iteration. Only afterwards, we will apply the Krylov subspace method. This would force ill-conditioning of the Krylov matrix $[r_0, \ldots, B^{k-1}r_0]$, which, as observed by Ipsen [6], is a necessary condition for small minimal residuals.

One has to be cautious when working with vectors that are at close angle to one another because of the unavoidable effects of finite precision arithmetic [7], and be aware of differences in true residuals and residuals obtained through updating processes. It should be intuitively clear though, that when building a Krylov subspace on a random initial residual r_0 in a high-dimensional space, it will in general not contain good approximations of r_0. Especially in long recurrences, it seems a waist to keep such an irrelevant part of the subspace in memory and to use it in computation. In asymptotic convergence rates of the methods, this waist is usually not visible. However, aiming for as little iterations as possible, these asymptotics hardly show up in the first place. In particular in non-normal applications, it is by now well-known [15] that one should be more worried about the initial phase of an iterative process. Therefore, in this paper, we concentrate on the start of Krylov subspace methods, as opposed to their asymptotic behavior.

1.2 Outline

The outline of this paper is as follows. In Section 2 we recall iterative methods. Then, in Section 3 we introduce extrapolation methods for classical iteration schemes. A general reference for extrapolation is [2], and for some more recent work see [8, 9, 18]. Alongside some numerical experiments we will discuss some stability matters and preconditioning together in Section 4. Apart from being useful on their own account, the extrapolation theory will serve as a tool for deriving, in Section 5, asymptotical properties of the effect of preprocessing the initial residual in minimal residual methods. In [17] this idea is worked out for the Conjugate Gradient method while we also refer to [12] and the references therein. After a mathematical analysis in Section 6 we see our expectations confirmed by more numerical experiments in Section 7. In Section 8 we comment on related topics and give our final conclusions. In Appendix A we give details on the testmatrices used.

2 Iterative methods for linear systems of equations

Iterative methods for approximating the solution of a non-singular system of linear equations $Ax = b$ are based upon the following principle. Starting with an initial guess x_0 for the solution, the initial residual $r_0 := b - Ax_0$ is calculated and a sequence x_k with corresponding residuals $r_k = b - Ax_k$ is constructed, using only the action of A on specific vectors. Since x is unknown, it is not clear where x_k should converge to. However, we do know that we want the residuals r_k to converge to zero. Therefore, we are interested in identifying a so called "search direction" u_k for the approximation and a "correction direction" c_k of the residual with the property that

$$Au_k = c_k \qquad \text{and} \qquad \|r_k - c_k\| < \|r_k\|. \tag{1}$$

If we have found such a pair (u_k, c_k), then setting

$$x_{k+1} := x_k + u_k \tag{2}$$

realizes the reduction of the norm of the residual as aimed for in (1), since

$$r_{k+1} = b - Ax_{k+1} = b - A(x_k + u_k) = r_k - c_k. \tag{3}$$

It is easy to generate pairs (u_k, c_k) at random such that $Au_k = c_k$; the problem is to find them such that c_k reduces the residual well. The ideal situation would be to have $u_k = A^{-1}r_k$ since then $c_k = r_k$ and we would have found $x := x_k + u_k$. However, calculating $A^{-1}r_k$, or solving $Au_k = r_k$, is in general as least as difficult as solving our original problem $Ax = b$.

2.1 Classical iterative methods

In classical methods, the problem of generating pairs of vectors as in (1) is approached as follows. One looks for a matrix K that is, in some sense, an approximation of A, and that is such, that solving $Ku_k = r_k$ is relatively (very) easy. Then, writing $B := AK^{-1}$, we have $c_k = Au_k = Br_k \approx r_k$, since if K was supposed to approximate A then B should be close to the identity. This gives the following algorithm. Start with some x_0 and corresponding r_0, and repeat until the norm of the residual is small enough

$$u_k = K^{-1}r_k, \quad c_k = Au_k, \quad r_{k+1} = r_k - c_k, \quad x_{k+1} = x_k + u_k, \quad k = k + 1. \tag{4}$$

Combining the lines above, one can easily write this iteration into the more familiar form

$$x_{k+1} = x_k + K^{-1}(b - Ax_k). \tag{5}$$

The choice $K = I$ is called the Richardson iteration, $K = D$, with D the diagonal of A, the Jacobi iteration, and $K = (L + D)$, with L the strict lower triangular part of A, is the Gauss Seidel method.

Theorem 2.1 ([5]) *The method (4) is convergent if and only if the spectral radius of the iteration matrix $I - B$ is less than one. Sufficient for convergence of the Gauss-Seidel method is positive definiteness of A. Sufficient for the Jacobi iteration to converge is positive definiteness of both A and $2D - A$.*

It has been observed in many practical applications that the convergence of classical methods (if convergent at all) is very poor. This is due to the following. First note, that the residual r_{k+1} can be expressed in terms of r_k and hence, recursively, in terms of the initial residual as follows. Writing $M := I - B$ we have

$$r_{k+1} = r_k - c_k = Mr_k = M^{k+1}r_0. \tag{6}$$

The *preconditioning* K to A is often too poor. Either the spectral radius $\rho(I-B)$ of the residual reduction matrix M is (much) larger than one, or, if smaller, only *very little* smaller than one. This leads in practice to a correction vector c_k that is, compared to r_k, very small in magnitude, and therefore only a small change in the norm of the residual is the result in each iteration step. Second, as already observed in the introduction, as a result of non-normality of the matrix M, the condition number $\kappa(V)$ of (any) basis V of eigenvectors might ruin the convergence even in case the spectral radius is small enough. Explicitly we have

$$\|r_{k+1}\| \leq \|M^{k+1}\| \leq \kappa(V)\rho(M)^{k+1}\|r_0\|, \tag{7}$$

which does mean an upper bound that reduces by a factor $\rho(M)$ in each iteration, but the upper bound may start as an extremely large value. A simple example illustrates that not only the upper bound is large; also the size of the residuals may increase substantially before convergence.

Example 2.2 Suppose that $v_1 := (\epsilon, 1)$ and $v_2 := (\epsilon, -1)$ are eigenvectors of a two by two matrix with respective eigenvalues $\frac{1}{2}$ and $\frac{3}{2}$. Then the Richardson iteration matrix $I-B$ has the same eigenvectors but with eigenvalues $\frac{1}{2}$ and $-\frac{1}{2}$. Let r_0 be the vector v_1+v_2, then $\|r_0\| = 2\epsilon$. Applying $I-B$ gives $r_1 = \frac{1}{2}(v_1 - v_2)$. In spite of the spectral radius being one half, the norm of the first residual is $\|r_1\| = 1$.

The pseudo-spectrum [15] of M often gives a better bound on the norm of the powers of a matrix, although the relevant pseudo-spectral radius is not easy to compute.

2.2 The Local Minimal Residual method

A first effort to overcome some of the problems of classical iterative methods is the following. Having found a pair (u_k, c_k) such that $Au_k = c_k$, we know that for all $\alpha \in I\!R$ the pair

$$(\hat{u}_k, \hat{c}_k) := (\alpha u_k, \alpha c_k) \tag{8}$$

satisfies $A\hat{u}_k = \hat{c}_k$. We can compute α_k such, that $\alpha_k c_k$ is the *best possible* correction of the residual r_k within the one-dimensional linear subspace spanned by c_k (with respect to the L^2-norm). It is easy to see that $\alpha_k c_k$ should be the L^2 orthogonal projection of r_k on the space spanned by c_k. Hence, α_k be computed accordingly, which leads to the following improvement over (4).

$$u_k = K^{-1}r_k, \quad c_k = Au_k, \quad \alpha_k = \frac{r_k^* c_k}{c_k^* c_k}, \quad r_{k+1} = r_k - \alpha_k c_k, \quad x_{k+1} = x_k + \alpha_k u_k. \tag{9}$$

This method is called a Local Minimal Residual (LMR) method, and clearly, the residuals for this method are non-increasing. If one extends this idea, one arrives at the Generalized Conjugate Residual method (GCR), which is mathematically equivalent to Generalized Minimal Residuals (GMRES).

2.3 Minimal residual methods

In GCR, in each iteration step, all the information about the action of the inverse A^{-1} that is obtained in previous iteration steps, is being used. Explicitly, after k

iteration steps one has $k+1$ sets of vectors u_j, c_j such that $AK^{-1}u_j = c_j$, and the initial residual can therefore be corrected optimally in the space spanned by $c_0, \cdots, c_k$ by means of L^2 orthogonal projection. In GCR, this projection is implemented by means of a recursive orthogonalization procedure for the vectors c_j. It can be seen that the span of the c_i is equal to the Krylov subspace $K^{k+1}(AK^{-1}, AK^{-1}r_0)$, where, for general B and v, $K^k(B;v)$ is defined by

$$K^k(B;v) := \text{span}\{v, Bv, \cdots, B^{k-1}v\}. \tag{10}$$

Explicitly, in GCR the following minimization problem is solved,

$$\text{find } c_k \in K^k(B, Br_0) \text{ such that } \|r_0 - c_k\| \text{ is minimal.} \tag{11}$$

Equivalently, since $K^{k+1}(B, Br_0) = BK^{k+1}(B, r_0)$, we could also solve

$$\text{find } u \in K^k(B, r_0) \text{ such that } \|r_0 - Bu\| \text{ is minimal.} \tag{12}$$

An implementation based on this formulation leads to GMRES, which, though mathematically equivalent to GCR, is less expensive (and has different stability properties). Given B and the initial residual r_0, an orthogonal basis $v_1, \cdots, v_{k+1}$ for the Krylov subspace $K^{k+1}(B, r_0)$ is computed using (for example) the Gram-Schmidt process. Writing V_j for the matrix with columns $v_1, \cdots, v_j (j \leq k+1)$, this results into the equivalent relations

$$BV_k = V_{k+1}H_{k+1,k} \quad \text{and} \quad BV_k = V_kH_k + h_{k+1,k}v_{k+1}e_k^* \tag{13}$$

where $H_{k+1,k} = (h_{ij})$ is an upper Hessenberg matrix which contains the orthogonalization and normalization coefficients, and H_k its upper $k \times k$ block. Writing $v_1 = r_0/\rho$ with $\rho := \|r_0\|$, the first relation in (13) is used to find the element V_ky_k from the column span $K^k(B, v_1)$ of V_k for which $\|BV_ky_k - r_0\|$ is minimal. Indeed, using (13) we get $\|BV_ky_k - r_0\| = \|V_{k+1}H_{k+1,k}y_k - \rho V_{k+1}e_1\| = \|H_{k+1,k}y_k - \rho e_1\|$ so y_k is defined by a small least squares problem that can be solved by standard methods.

In case B is Hermitian, GCR and GMRES reduce to CR and MR respectively. In these methods, the orthogonalization reduces to a three term recursion, as opposed to the long recurrences in GCR and GMRES. We will now define extrapolation schemes of classical methods that can be interpreted as inexact minimal residual methods in a Krylov subspace, and encounter the first case of deliberate ill-conditioning of Krylov matrices.

3 Extrapolation of classical methods

Let us first concentrate on the effect of one projection step of LMR, which, one should note, requires the evaluation of two inner products. From Figure 1 it is clear that one such a step has the most effect if r_k and c_k are almost linearly dependent, which is the case when r_k and the next classical residual $r_{k+1}^C := r_k - c_k$ make a small angle. In equation (6) we have seen that the classical residuals are (unscaled) power method iterates. So, the angle between consecutive residuals should become smaller at the convergence rate of the power method. The same is valid for the angle between the c_k and r_k. This raises the question if it would pay off to apply the projection in LMR not in each step (since in LMR one computes x_{k+1}^{LMR} such that r_{k+1}^{LMR} is *orthogonal* to c_k), but only after some more steps of the classical iteration, in order to let the Power Method do its work and have r_k and c_k at small angle to one another. This would be our first example of deliberate ill-conditioning in order to speed up convergence. In the following we will give a mathematical analysis.

3.1 First extrapolation

Assume that k steps of a classical iteration have been performed, which resulted in an approximation x_k with corresponding residual $r_k := b - Ax_k$. Application of the next step of this classical iteration leads to a correction vector c_k and a search direction u_k, and to the updated approximation x_{k+1} and corrected residual r_{k+1} as in (4). Alternatively, instead of the $k+1$-st step of the classical iteration, we can, after having determined c_k and u_k, compute $a_k = r_k^* c_k / c_k^* c_k$, which leads to a different updated approximation and a different corrected residual according to (9), which we will denote from now on by s_{k+1} (Cf. Figure 1 and Alg.(3.1)). Clearly,

$$\frac{\|s_{k+1}\|}{\|r_{k+1}\|} \leq 1. \tag{14}$$

For the time being, we interpret the computation of s_{k+1} as a *post-processing* or *extrapolation* step that applied after some number k of classical iteration steps. However, for the ease of theoretical discussion, in the algorithm below this extrapolate is constructed for each value of k. In practical situations, there is no need to do this after *each* classical iteration step.

```
ALGORITHM 3.1: Extrapolated Classical Method.
    input: $A, K, b, x_0$, tolerance
    $r_0 = b - Ax_0$
    $k = 0$
    while $\|s_k\|_2 >$ tolerance
        $u_k = K^{-1} r_k$
        $c_k = Au_k$
        $\alpha_k = c_k^* r_k / (c_k^* c_k)$
        $r_{k+1} = r_k - c_k$
        $x_{k+1} = x_k + u_k$
        $s_{k+1} = r_k - \alpha_k c_k$
        $y_{k+1} = x_k + \alpha_k u_k$
        $k = k + 1$
    end (while)
```

In the following analysis, we will prove a more interesting bound than (14) for the reduction factor resulting from the extrapolation.

3.2 The improvement rate of the extrapolation

Recall the notation $M = I - AK^{-1}$, and define θ_k as the angle between r_k and r_{k+1}, i.e. let

$$\cos\theta_k = \frac{r_k^* r_{k+1}}{||r_k|| ||r_{k+1}||} = \frac{r_k^* M r_k}{||r_k|| ||M r_k||}. \tag{15}$$

Assume M has a dominant single eigenvalue λ_1 with a corresponding eigenspace spanned by the eigenvector v_1. Let λ_2 be an eigenvalue of second-largest magnitude.

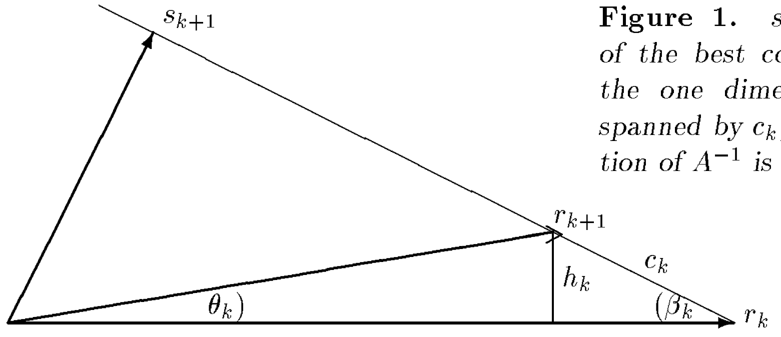


Figure 1. *s_{k+1} is the result of the best correction of r_k in the one dimensional subspace spanned by c_k, on which the action of A^{-1} is known.*

Consider Figure 1. Using basic trigonometry and some additional notations introduced in the picture, we can immediately write down

$$\sin\beta_k = \frac{||s_{k+1}||}{||r_k||} = \frac{||h_k||}{||c_k||}, \qquad \text{so,} \qquad \frac{||s_{k+1}||}{||r_{k+1}||} = \frac{||r_k||}{||c_k||}\sin\theta_k. \tag{16}$$

On the left-hand side of the expression in the right, the extra reduction of the classical residual obtained by performing the extrapolation is given.

Let us now recall some well-known results about the convergence of the power method. These results are already specialized for the situation in which the iteration matrix is M and the start vector r_0 and in which no scaling is applied to the iteration vectors. We refer to [3] for details.

Proposition 3.1 *Suppose that $r_0^* v_1 \neq 0$ and let γ_k be the angle between v_1 and r_k. Then there exists a $C > 0$ such that for all k*

$$|\sin\gamma_k| \leq C\left|\frac{\lambda_2}{\lambda_1}\right|^k. \tag{17}$$

Proposition 3.2 *Suppose that $|\lambda_1| \neq 1$. Then*

$$\frac{||r_k||}{||c_k||} \to \frac{1}{1-|\lambda_1|} \quad \text{for } k \to \infty. \tag{18}$$

Combining the two propositions above with eq.(16) leads to the following upper bound for the reduction of the residual after extrapolation.

Corollary 3.3 *Applying one extrapolation step after k steps of the classical method reduces the norm of the residual with an additional factor as follows. There exist numbers N and C such that for all $k \geq N$*

$$\frac{||s_{k+1}||}{||r_{k+1}||} \leq \min\left\{1, \frac{2C}{1-|\lambda_1|}\left|\frac{\lambda_2}{\lambda_1}\right|^k\right\}. \tag{19}$$

Proof. The angle θ_k in eq.(16) is smaller than or equal to the sum of the angles γ_k and γ_{k+1}. The same is valid for their sines. □.

An interpretation of Corollary 3.3 is the following. A classical iterative method with spectral radius almost one shows slow convergence. If this is caused by only one single eigenvalue with modulus almost one, this can be corrected successfully by extrapolation in a one-dimensional Krylov subspace. Indeed, the convergence rates of the residuals $||r_k$ and $||s_k||$ are

$$||r_k|| \leq C_1|\lambda_1|^k \quad \text{and} \quad ||s_k|| \leq C_2|\lambda_2|^k, \tag{20}$$

respectively. One can interpret these results also as iterating with the deflated matrix after having found the dominant eigenvector. The deflated matrix is never explicitly constructed. Note that the asymptotics might take many iterations to show up if the matrix M is far from normal. In case of high non-normality, it is not clear what happens at early stages, although residuals will never increase.

3.3 Second extrapolation

Consider Figure 2, which is an extension of Figure 1. It contains not only the extrapolated residual $s_{k+1} := r_k - \alpha_k c_k$ but also the next extrapolated residual $s_{k+2} := r_{k+1} - \alpha_{k+1} c_{k+1}$ that arises from approximation of r_{k+1} in the space spanned by c_{k+1}. In order to emphasize that r_{k+2} (and several other vectors) do not need to be in the same plane as r_k, r_{k+1}, c_k and s_{k+1}, they are drawn as thick lines, indicating that they might have a component orthogonal to this sheet of paper.

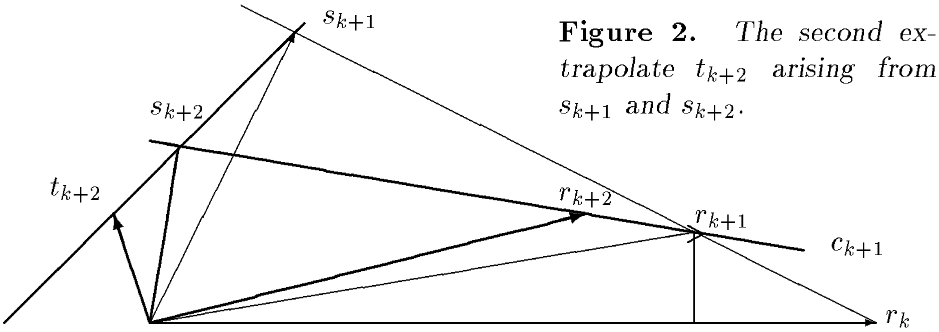


Figure 2. *The second extrapolate t_{k+2} arising from s_{k+1} and s_{k+2}.*

One might hope for the two vectors s_{k+1} and s_{k+2} to be close to linearly dependent again. Their difference is

$$d_{k+1} := s_{k+1} - s_{k+2} = (r_k - \alpha_k c_k) - (r_{k+1} - \alpha_{k+1} c_{k+1}) = (1 - \alpha_k) c_k + \alpha_{k+1} c_{k+1}. \tag{21}$$

This means, that we know the action of A^{-1} on the space spanned by d_{k+1}.

$$v_{k+1} := A^{-1} d_{k+1} = (1 - \alpha_k) u_k + \alpha_{k+1} u_{k+1}. \tag{22}$$

As before, we can now correct the extrapolated residual s_{k+2} in the space spanned by d_{k+1} to obtain an extrapolated extrapolation, which we will denote by t_{k+2}.

$$t_{k+2} := s_{k+2} - \beta_{k+1} d_{k+1}, \qquad \text{where} \quad \beta_{k+1} = \frac{s_{k+2}^* d_{k+1}}{d_{k+1}^* d_{k+1}}. \tag{23}$$

Obviously the residual t_{k+2} is always smaller in norm than s_{k+2}.

3.4 Some intuition on the improvement rate

We already noted that r_k as well as c_k converge towards the dominant eigenvector direction v_1 of M. This means, that the component in the direction of v_1 in s_{k+1} is small, since s_{k+1} is the orthogonalization of r_k to c_k. The same is valid for s_{k+2}. Since a simple computation shows that

$$Ms_{k+1} = s_{k+2} + (\alpha_{k+1} - \alpha_k)c_{k+1}, \tag{24}$$

and also that

$$\lim_{k\to\infty} (\alpha_{k+1} - \alpha_k) = 0, \tag{25}$$

we could therefore hope for the sequence s_k to behave similarly as r_k. The improvement of the second interpolate over the first will then be of order $|\lambda_3/\lambda_2|$ and hence

$$||t_k|| \leq C_3|\lambda_3|^k. \tag{26}$$

Note that he calculation of the correction direction d_{k+1} could become relatively inaccurate since both s_{k+1} and s_{k+2} are the result of the subtraction of two almost equal vectors, and are themselves subtracted from one another. We will pursue this issue further in Section 4.

3.5 Further extrapolates

The process of extrapolation can be extended in a similar fashion. Taking two vectors t_{k+1} and t_k, we can consider their difference

$$e_{k+1} := t_k - t_{k+1} = s_k - \beta_{k-1}d_{k-1} - s_{k+1} + \beta_k d_k = (1+\beta_k)d_k - \beta_{k-1}d_{k-1}. \tag{27}$$

We know from (22) that $A^{-1}d_k = v_k$ so we can correct t_{k+2} in the space spanned by e_{k+1}, leading to a new and smaller residual. We will not go into detail here, but only state the following.

Remark 3.4 In exact arithmetic and for fixed m, assuming that the largest m eigenvalues of M are single, the asymptotic behavior in k of the m-th extrapolate of the k-th classical residual R_k^m is

$$||R_k^m|| \leq C_{m+1}|\lambda_{m+1}|^k. \tag{28}$$

Moreover, the sequence $||R_k^m||$is (non-strictly) monotonic decreasing in m.

Just for clarity of notation, we will add here the computational scheme of the extrapolates. In order to be able to calculate a certain extrapolate, one only needs the one directly left and the one left above from it.

$$\begin{array}{lllll}
r_0 = R_0^0 & & & & \\
r_1 = R_1^0 & s_1 = R_1^1 & & & \\
r_2 = R_2^0 & s_2 = R_2^1 & t_2 = R_2^2 & & \\
r_3 = R_3^0 & s_3 = R_3^1 & t_3 = R_3^2 & R_3^3 & \\
r_4 = R_4^0 & s_4 = R_4^1 & t_4 = R_4^2 & R_4^3 & R_4^4
\end{array} \tag{29}$$

For completion, we give below the algorithm that computes two extrapolates in each iteration step. As mentioned before, in practice one should compute them only occasionally.

```
ALGORITHM 3.2: Twice Extrapolated Classical Method.
    input: $A$, $K$, $b$, $x_0$, tolerance
    $r_0 = b - Ax_0$
    $s_0 = r_0$
    $\alpha_{-1} = 0$
    $k = 0$
    while $||t_k||_2 >$ tolerance
        $u_k = K^{-1} r_k$
        $c_k = Au_k$
        $\alpha_k = c_k^* r_k / (c_k^* c_k)$
        $r_{k+1} = r_k - c_k$
        $x_{k+1} = x_k + u_k$
        $s_{k+1} = r_k - \alpha_k c_k$
        $y_{k+1} = x_k + \alpha_k u_k$
        $v_k = (1 - \alpha_{k-1}) u_{k-1} + \alpha_k u_k$
        $d_k = s_k - s_{k+1}$
        $\beta_k = s_{k+1}^* d_k / (d_k^* d_k)$
        $t_{k+1} = s_{k+1} - \beta_k d_k$
        $z_{k+1} = y_{k+1} + \beta_k v_k$
        $k = k + 1$
    end (while)
```

4 True residuals and preconditioning: experiments.

Our exposition so far has concentrated on taking advantage of iterates of the power method that make a small angle to each other. Our mathematical analysis gives statements that are asymptotically valid in exact arithmetic. In practice, we will have to deal with the effects of finite precision arithmetic, and this asks for special care. In what follows, we will perform some numerical experiments and try to identify points in the approach that need extra attention. We will comment on the difficulty of finding a (nearly) convergent classical method, and on true versus updated residuals. For details on the testmatrices we refer to the Appendix.

4.1 True versus recursively computed residuals

In our first experiment, we solved a system $Ax = b$, where A is the non-Hermitian SHERMAN3 matrix of dimension 5005 and b is the corresponding right-hand side taken from [11]. As preconditioning we used Incomplete LU-factorization with threshold 0.001, which we denote by ILU(0.001). We monitored the classical residuals and its first five extrapolates in logarithmic scale. The convergence history of the recursively computed residuals is shown in the left picture of Figure 3. The upper graph represents the classical residuals, going down we see the norms of the once extrapolated residuals (the s_k) and further down the second to the fifth extrapolate. In the right picture we plotted the true residuals, i.e. the residuals directly computed from the approximations.

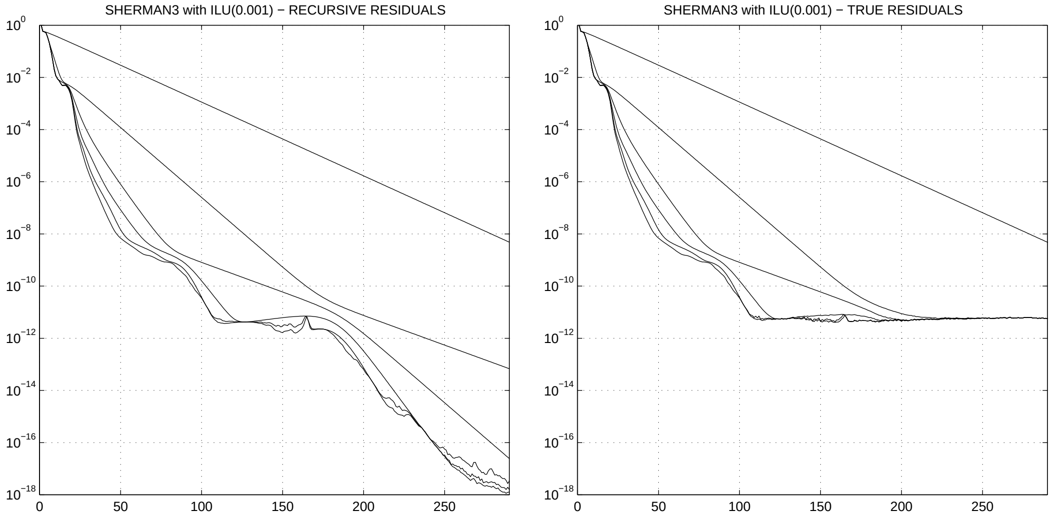


Figure 3. True versus recursively computed residuals for SHERMAN3, incomplete LU preconditioned with threshold 0.001.

As has been observed in many iterative methods, there can be significant differences between the two. And indeed, here too we see that the true residuals do not become smaller than about 1e-11 relatively, whereas the recursively computed residuals go easily all the way down to 1e-18. This should be a guideline in the use of the extrapolation: at least in the final stage, one should monitor the true residuals, as the recursive ones can be inaccurate.

4.2 An alternative computational scheme

One of the key points in our algorithm (3.2) has been the property that we know the action of A^{-1} on the difference of two consecutive iterates s_k. Explicitly, we have

$$Av_k := A\left((1-\alpha_{k-1})u_{k-1} + \alpha_k u_k\right) = d_k. \tag{30}$$

However, the relation $Av_k = d_k$ has not been established by direct calculation via the use of the action of A on v_k or A^{-1} on d_k. It might therefore very well be that in finite precision arithmetic, the *residual* $Av_k - d_k$ is non-zero, whereas when d_k was explicitly defined as $d_k := Av_k$, it would be zero. But since v_k is computed independent from d_k, it is actually possible to replace the line $d_k = s_k - s_{k+1}$ by $d_k := Av_k$. This makes sure that the correction and updating of t_{k+1} and z_{k+1} happens with vectors $\beta_k d_k$ and $\beta_k v_k$ that have the property that $d_k := Av_k$ as much as is possible in finite precision arithmetic. The same can be done for all further extrapolates, and in particular when one plans to compute many of them, this might add to the stability of the algorithm. We should however point out, that it did not make any difference to the results of the experiments in this paper.

4.3 Preconditioning

Extrapolation of a classical method makes sense if the classical method converges, or, as we will see below, does not diverge to strongly. This rather seems to restrict the range of application, since we have already noted that classical methods often fail to converge. In the following experiments, we discuss suitable preconditioning.

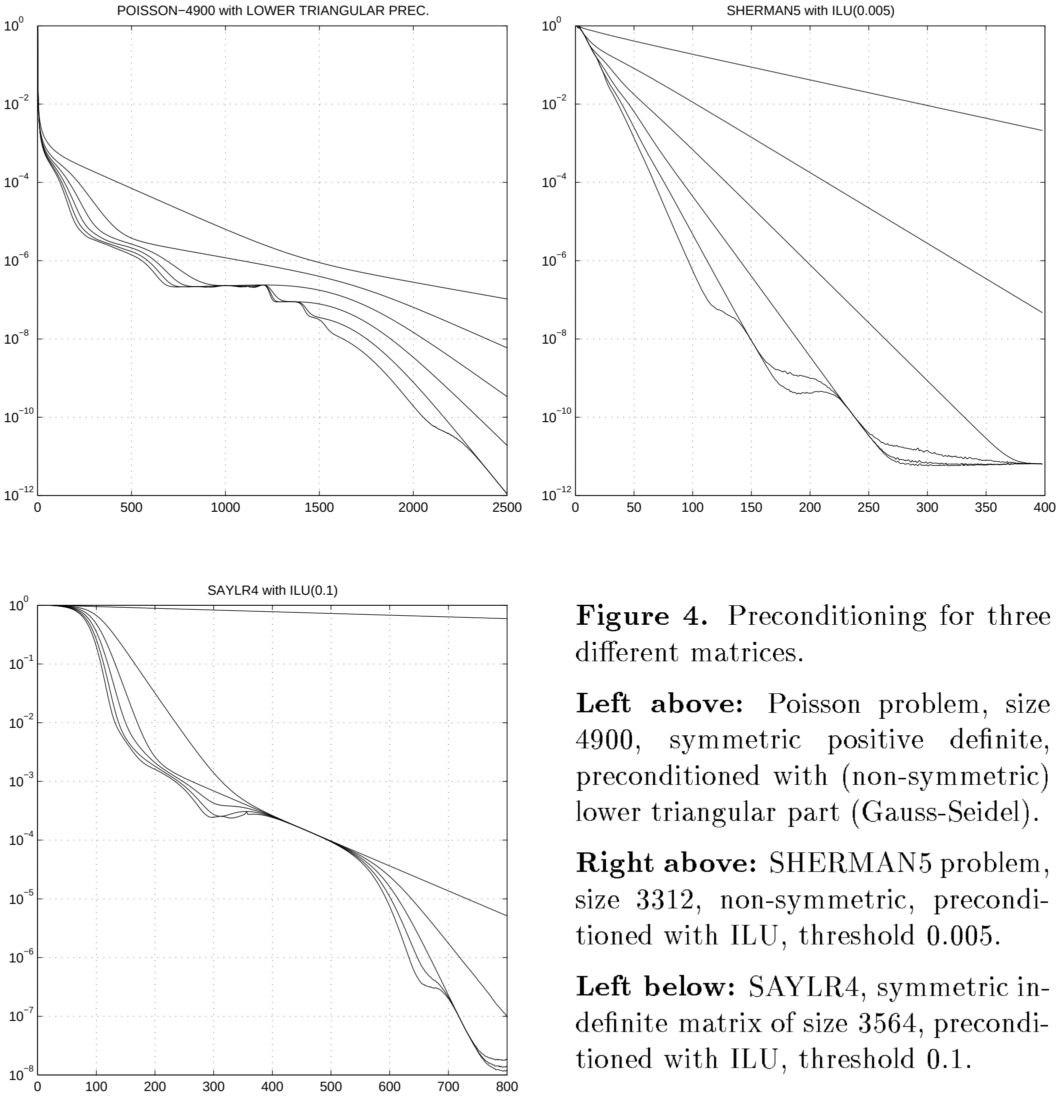


Figure 4. Preconditioning for three different matrices.

Left above: Poisson problem, size 4900, symmetric positive definite, preconditioned with (non-symmetric) lower triangular part (Gauss-Seidel).

Right above: SHERMAN5 problem, size 3312, non-symmetric, preconditioned with ILU, threshold 0.005.

Left below: SAYLR4, symmetric indefinite matrix of size 3564, preconditioned with ILU, threshold 0.1.

In all three examples, the true residuals are shown, which were (with the bare eye) indistinguishable from the recursively computed ones.

First example. Left above a Poisson problem on a square is solved using standard finite differences on, leading to a positive definite system matrix of size 4900 for which Gauss-Seidel converges (Cf.Th.2.1). As preconditioner we also tried ILU(0.1) which gave a similar picture, but then in about 700 iterations instead of 2500.

Second example. The picture right above in Figure 4 shows the results for the SHERMAN5 matrix, which is non-Hermitian and has size 3312. The problem was now which K to choose such that the classical method converges, since both Jacobi as Gauss-Seidel showed divergence. We chose to use ILU(0.005). The computation of the factorization costed only about half a percent of the computation of a complete LU factorization, but already gives very good convergence of the extrapolates.

Third example. In the picture left below we took the symmetric indefinite SAYLR4 matrix. Again we took ILU preconditioning, with threshold 0.1. Although the classical method hardly converges, the extrapolates converge quite well. Compare this with the discussion after Corollary 3.3.

Remark 4.1 Let us stress once more that it is essential that the preconditioner is constant in the iteration number. Hence, for example, the Richardson, Jacobi and Gauss-Seidel method can be used, and also (Modified) Incomplete LU decomposition. In the course of the computation of the iterates one should monitor the *true* residuals and not the *recursively* computed ones. For higher extrapolates, the adapted method (Cf.Sect.4.2) could be considered.

4.4 Slowly divergent classical method

Here, we wish to pay some extra attention to the fact that even if the classical method, after preconditioning, fails to converge, it is still possible that extrapolation is effective. Consider the following example with divergent classical method.

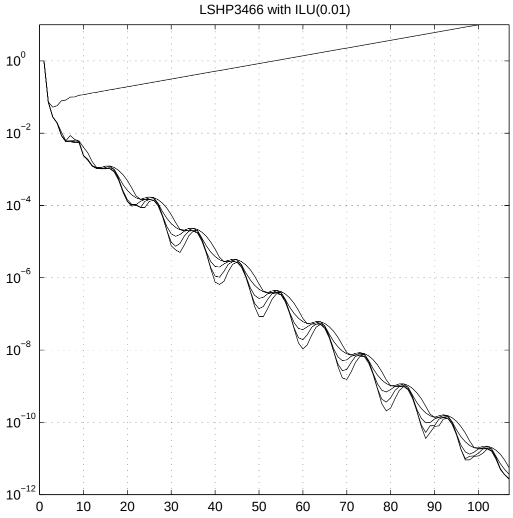


Figure 5. A divergent classical method with convergent extrapolates.

The symmetric indefinite matrix LSHP3466 of size 3466 gives a divergent classical method when ILU(0.01) is used as preconditioner. In spite of that, the extrapolates converge relatively very fast, although computing higher extrapolates does not really seem worthwhile.

4.5 A note on inexact extrapolation

Consider once more Figure 1. To get the best possible correction of r_k in the space spanned by c_k, we have to evaluate two inner products. However, when r_k and c_k are at small angle to each other then we can approximate the projection as follows.

$$\frac{r_k^* c_k}{c_k^* c_k} \approx \frac{||r_k||}{||c_k||} \approx \frac{||r_k||}{||r_k|| - ||r_{k+1}||} = \frac{1}{1 - ||r_{k+1}||/||r_k||} \rightarrow \frac{1}{1-\lambda_1}. \tag{31}$$

This approximation involves two norms that have already been computed since they are used as stopping criterion in the classical method. Since the approximation has no effect on the r_k themselves, it will gradually improve, leading to almost the same s_k as without this approximation. The resulting algorithm 4.1 is studied from a different viewpoint in [9].

```
ALGORITHM 4.1: Inexact Extrapolated Classical Method.
    input: $A, K$, $b$, $x_0$, tolerance
    $r_0 \quad = b - Ax_0$
    $\rho_0 \quad = r_0^* r_0$
    $k \quad = 0$
    while $||\hat{s}_k||_2 >$ tolerance
        $u_k \quad = K^{-1} r_k$
        $c_k \quad = A u_k$
        $r_{k+1} = r_k - c_k$
        $x_{k+1} = x_k + u_k$
        $\rho_{k+1} = r_{k+1}^* r_{k+1}$
        $\gamma_k \quad = 1/(1 - \rho_{k+1}/\rho_k)$
        $\hat{s}_{k+1} = r_k - \gamma_k c_k$
        $\hat{y}_{k+1} = x_k + \gamma_k c_k$
        $k \quad = k + 1$
    end (while)
```

The approximation of inner products in this fashion can also be done for the higher extrapolates; we expect that asymptotically, the convergence graphs will coincide with those of the exact method. This was confirmed by numerical experiments (not shown in this paper).

5 Preprocessing of minimal residual methods

In this section we will use the extrapolation method of the previous section to study some Krylov subspace methods. This is motivated by the fact that the first extrapolation step is in fact the first step of minimal residual methods like GCR and GMRES. Again we will write B for AK^{-1}, though not implying that the preconditioned matrix has been explicitly formed.

5.1 Upper bounds for minimal residual methods

Recall that, although their implementation differs, both GCR and GMRES, as well as their Hermitian versions CR and MINRES, correct the initial residual by its best approximation in the Krylov Subspace $K^{k+1}(B, Br_0)$ (Cf.Sect.2.3). Now, consider the extrapolation method of Section 3. In particular, let k be fixed and consider the sequence R_k^m for increasing m. Clearly, the first extrapolate $s_k = R_k^1$ minimizes the "initial" residual r_k by optimal correction in $K^1(B, Br_k)$. For the second extrapolate $t_k = R_k^2$ we already noticed that is was constructed by correcting R_k^1 in the space spanned by d_k. Remembering equation (21) tells us that this correction lies in $K^2(B, Br_{k-1})$, since it uses c_k and c_{k-1}. One can easily check using induction arguments that for higher iterates and higher extrapolations the same still holds.

Lemma 5.1 *For all k and all m, $R_k^m - r_k \in K^m(B, Br_{k-m})$. For application of m extrapolation steps to r_k one needs the m previous residuals.*

So, because the extrapolations form a non-optimal Krylov subspace method, the m-th iterate of a minimal residual method with start vector r_k, has a norm that is bounded by the norm of the m-th extrapolate of r_{k+m}. This results in the following theorem.

Notation. Denote by GCR$(m)r$ and CL$(k)r$ the residuals obtained after applying to the initial residual r, m steps of GCR and the classical method respectively.

Theorem 5.2 *The following asymptotics hold for GCR applied to a residual obtained by the classical iteration method.*

$$||\mathrm{GCR}(m)\mathrm{CL}(k)r_0|| \leq ||R_{k+m}^m|| \leq C_{m+1}|\lambda_{m+1}|^{k+m}. \tag{32}$$

This means, that a graph in which the convergence history of the classical method, all its extrapolates and GCR are displayed, no graphs intersect. The un-extrapolated classical method and GCR form their respective upper and lower bounds as illustrated in Figure 6 for a small convection-diffusion problem.

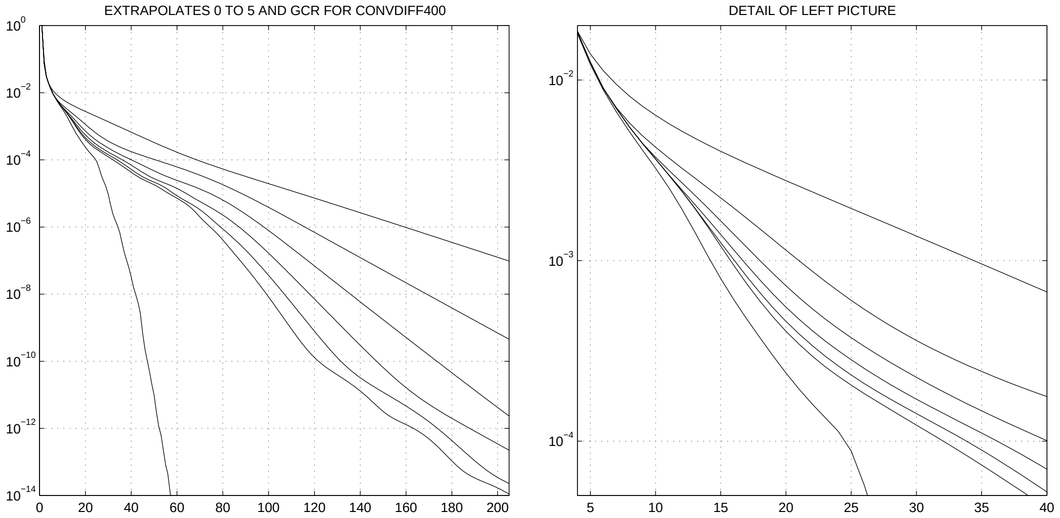


Figure 6. Classical method (upper graph), GCR (lower graph) and the five extrapolates in between for CONVDIFF400, right picture is a detail of the left.

Remark 5.3 Since the asymptotics for small m will need to be realized in practice before the asymptotics for larger m can take place, we have informally re-derived the superlinear upper bound for the convergence pattern of minimal residual methods (Cf.[16]). In a logarithmic convergence history of a minimal residual method, the norms of the residuals are bounded by a continuous piecewise linear graph of which the slopes decrease for k tending to infinity.

These observations naturally raise the question if it would pay off to use a (few steps of a) Krylov subspace minimal residual method as extrapolation. Alternatively, one could consider the classical method as a *preprocessing* for this Krylov subspace method. We will from now on continue our analysis from this point of view.

5.2 Preprocessing

The right picture in Figure 6 (depicted again, and extended, in Figure 7) will serve as a model for our further discussion. First note that in that picture, the GCR graph stays quite close to the graph of the fifth extrapolate. It only starts to turn away from it after iteration 25, roughly. A consequence of this proximity is the following. Suppose that we do $k \leq 20$ classical iterations with the same starting vector as before. Then, we use r_k as initial residual for GCR. From Theorem 5.2 it follows that the first five iterations of GCR must give residuals that are smaller than or equal to the extrapolates $||R^m_{k+m}||, m = 0, \cdots, 5$. And indeed , this is what we clearly see in Figure 7. The stars '*' on the dotted lines indicate GCR convergence histories that had initial residual $r_5, r_{10}, \cdots, r_{25}$ (which, due to MatLab plotting, have x-coordinates $6, 11, \cdots, 26$).

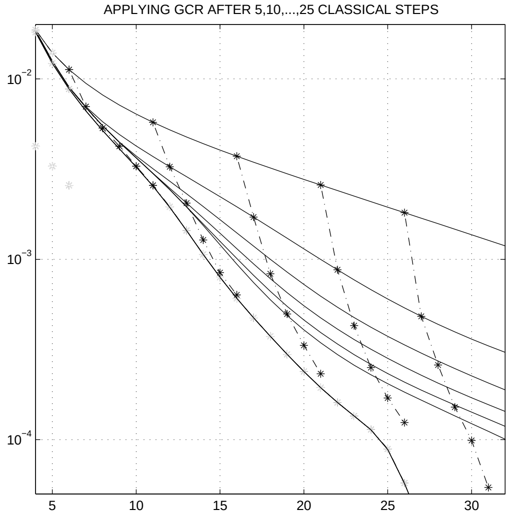
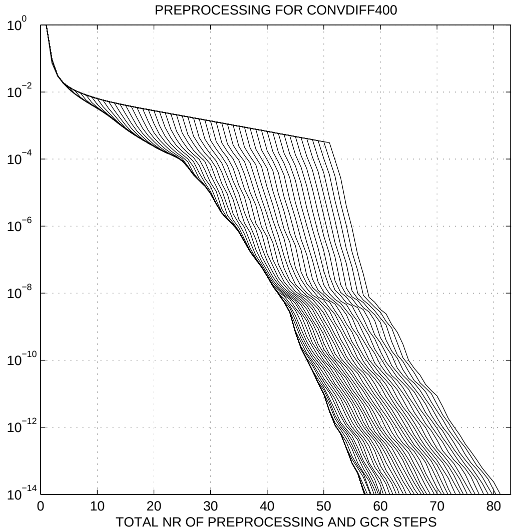


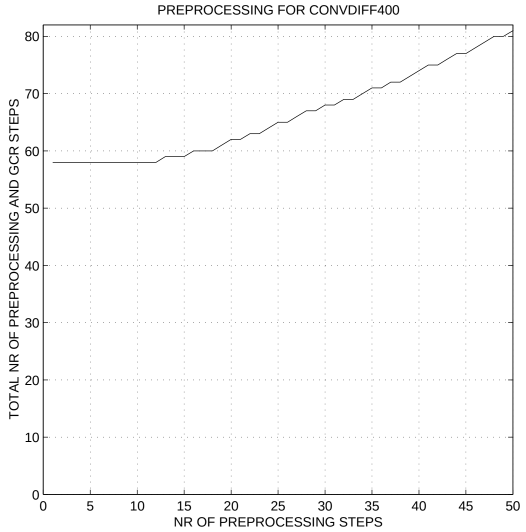
Figure 7. As an example, let the initial residual for GCR be r_{15} (situated at $x = 16$ as a result of r_0 being at $x = 1$). The first GCR iterate is *equal* to the first extrapolate $s_{16} = R^1_{16}$ of r_{16}. The second GCR iterate cannot be *larger* than second iterate of r_{17}, and so on. Since the graph of the fifth extrapolate is still close to the original GCR graph, the graph of pre-processed GCR is necessarily almost forced back onto the original GCR graph.

In each iteration of GCR starting with a pre-processed initial residual, the graph of the next extrapolate has to be passed. Since the GCR$(\cdot)r_0$ graph stays close to the fifth extrapolate, the graphs of GCR(5)CL(ℓ), for ℓ roughly smaller then 20, almost catch up with the GCR$(\cdot)r_0$ graph. Of course it could be that for larger ℓ this still happens, because it could be that the graphs of still higher extrapolates stick to the GCR$(\cdot)r_0$ graph even longer than the fifth extrapolate's graph.

Remark 5.4 Starting GCR after a larger number of preprocessing iterations does, of course, not need to produce a graph that falls back onto the original GCR graph. Nor does it (for any number of preprocessing steps) have to stick onto this graph for the rest of the convergence history. Nevertheless, according to our theory, the more preprocessing steps are taken, the faster the upper bounds for GCR decay.

Before proceeding with a mathematical analysis of preprocessing, we will show, in Figure 8, what happens if we preprocess the small CONVDIFF400 problem with Gauss Seidel steps, in number varying from zero to fifty. We set the relative residual tolerance to 1e-14. In the picture left above, we see fifty convergence histories. The lowest one, of course, is plain GCR, and the upper one is GCR pre-processed with fifty Gauss-Seidel steps. The other graphs split off the upper graph after a number of preprocessing steps and go GCR-like down. In the upper right picture, the total number of iterations (Gauss-Seidel plus GCR) is given against the number k of Gauss-Seidel steps. It is remarkable that this graphs stays constant until $k = 12$.





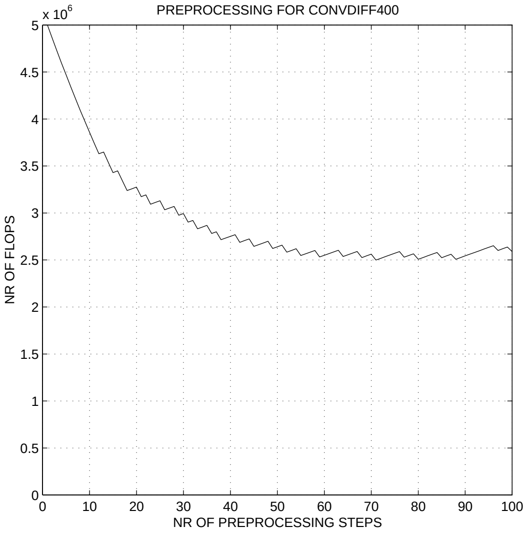


Figure 8. Three aspects of preprocessing.

Left above: Problem CONVDIFF400. Convergence graphs of GCR with $k = 0, 1, \cdots, 50$ Gauss-Seidel preprocessing steps.

Right above: Total number of iterations (Gauss-Seidel steps plus GCR steps) against number of preprocessing steps.

Left below: Amount of flops against number of preprocessing steps. Good preprocessing gains a factor two.

This means that preprocessing GCR with 12 Gauss-Seidel steps needs only 35 GCR iterations, while plain GCR needs 57. All the graphs with $k \leq 12$ preprocessing steps "converged" to the original GCR graph, which emphasizes the success of preprocessing. The dimension of the Krylov subspace reduced from 57 to 35, which is interesting since the amount of work to build a k dimensional subspace is quadratic in k.

Left below in Figure 8 we pictured the amount of floating point operations (flops) against the number of preprocessing steps. Although this number is not totally reliable because we calculated the true residuals in each step, the general shape of the graph shows that it saves quite a lot of work when preprocessing steps are done. Also, one does not really have to be afraid of doing to many steps since there seems to be a relatively long flat part of the graph. The up-down structure in the right part of the graph is explained by the fact that there, where the graph goes down a bit, another GCR step has been saved. It takes however to many preprocessing steps to reduce the number of GCR steps by one, so it does not pay of in the total amount of work done.

6 Mathematical aspects of the preprocessing

In this section we will try to analyze preprocessing from a number of different viewpoints. First we will see that it can be interpreted as changing the inner product of the projection method, and provide some intuition on the remarkable phenomenon visible in Figure 8, where we noticed that preprocessing can replace early minimal residual iterations. Throughout this section we assume that $\rho(M) < 1$.

6.1 Preprocessing is changing the inner product

To start with, we will study the difference between the following two means to reduce the initial residual. Write $M = I - B$.

(A) Apply m steps Classical Method followed by k steps Krylov Subspace method. Denote the resulting residual by r^a_{k+m} =GMRES(k)CL(m)r_0.

(B) Apply k steps Krylov Subspace method followed by m steps Classical Method. Denote the resulting residual by r^b_{k+m} =CL(m)GMRES(k)r_0.

In the first approach (A), the initial residual r_0 is first multiplied by M^m and then optimally corrected in the Krylov Subspace $K^k(B, BM^m r_0)$.

Proposition 6.1 *B and $I - B$ are commuting matrices, and hence*

$$K^k(B, BM^m r_0) = M^m K^k(B, Br_0). \tag{33}$$

Corollary 6.2 *The correction of $M^m r_0$ is of the form $M^m c^a$ with $c^a \in K^k(B; Br_0)$ where c^a satisfies*

$$\forall v \in K^k(B, Br_0) \quad : \quad (M^m(r_0 - c^a), M^m v) = 0. \tag{34}$$

This makes c^a into the best approximation of r_0 in the space $K^k(B; Br_0)$ with respect to the inner product generated by the Hermitian positive definite matrix $(M^m)^* M^m$, and hence the minimizer of the norm

$$\|r^a_{k+m}\| = \|M^m(r_0 - c^a)\| = \min\{\|M^m(r_0 - c)\| \mid c \in K^k(B; Br_0)\}. \tag{35}$$

This means that the m pre-processing steps can be interpreted as *changing the inner product of the Krylov Subspace Method.*

In the second approach (B), r_0 is first approximated in $K^k(B; Br_0)$ and the resulting corrected residual is multiplied by M^m, leading to

$$r^b_{k+m} = M^m(r_0 - c^b), \quad \text{where } \|r_0 - c^b\| = \min\{\|(r_0 - c)\| \mid c \in K^k(B; Br_0)\}. \tag{36}$$

Theorem 6.3 *The first approach (A) reduces the residual best.*

$$||r^a_{k+m}|| = ||M^m(r_0 - c^a)|| \leq ||M^m(r_0 - c^b)|| = ||r^b_{k+m}||. \tag{37}$$

Proof. By definition, c^a minimizes the norm $||M^m(r_0 - \cdot)||$ in $K^k(B; Br_0)$. □

This result gives an interesting view on the preprocessing. It is well-know that initially, a minimal residual method reduces the eigenvector components belonging to the extremal eigenvalues best. Applying m steps of a classical method *after* those components have been reduced, would therefore give much better residual reduction (in these m steps) than applying m classical steps *before* starting the minimal residual method.

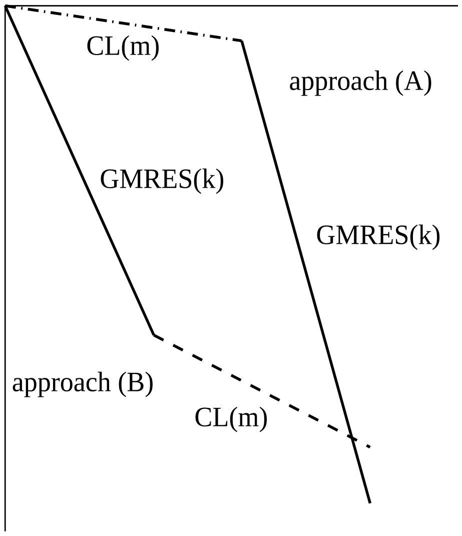


Figure 9. Illustrating Theorem 6.3. Residual reduction of approaches (A) and (B) on the vertical axis, and iteration number on the horizontal. The method that is applied secondly, profits from pre-processing by the first. Indeed, the slope of CL(m) is larger in (B) then in (A), and the slope of GMRES(k) is larger in (A) then in (B). But GMRES as second method is always best.

Nevertheless, Theorem 6.3 shows that approach (A) gives a better total reduction (for the combination of classical method and GMRES), which implies that pre-processed GMRES behaves better than plain GMRES. Apart from that, the gain of (A) over (B) is bigger than the gain a GMRES-pre-processed classical method has over a plain classical method.

6.2 Explanation of GMRES(k)CL(m) $\approx$ GMRES($k + m$)

As observed in numerical experiments of previous and upcoming sections of this paper, sometimes the pre-processing seems to replace a part of the optimal Krylov subspace method, or, put differently, GMRES(k)CL(m)$\approx$ GMRES($k+m$). It would be interesting to identify when this happens, because obviously, replacing the first m steps of GMRES by a classical iteration would save memory and computational costs. For ease of explanation, we analyze the case $m = 1$. First note that

$$K^k(B; (I - B)r_0) \subset K^{k+1}(B; r_0). \tag{38}$$

The space on the left-hand side is the k-dimensional Krylov subspace in which the pre-processed 'initial' residual $(I - B)r_0$ is approximated. On the right-hand side we have the $(k + 1)$-dimensional subspace in which r_0 is approximated. The question of equality of both approaches can therefore be reformulated as the question when the minimizer $y \in K^{k+1}(B; r_0)$ of the norm $\|r_0 - By\|$ happens to be in $K^k(B; (I-B)r_0)$ as well. Now, recall that

$$\forall v \in K^{k+1}(B; r_0), \quad v = P(B)r_0 \tag{39}$$

for some polynomial P of degree $k+1$, and correspondingly,

$$\forall w \in K^k(B; (I-B)r_0), \quad w = Q(B)(I-B)r_0 \tag{40}$$

for some polynomial Q of degree k. Since polynomials in B commute, the answer to our problem is therefore given: if the polynomial P^* defining the minimizer y has a factor $I - B$, the second approach gives the same residual.

Since the zeros of P^* are the eigenvalues of the projected matrix $H_{k+1} := V_{k+1}^* A V_{k+1}$ (the so-called Ritz values, see (13)), the two convergence graphs of GMRES(k)CL(1) and GMRES($k+1$) stick together from the point N onwards that H_j has a Ritz value equal to one for all $j \geq N$. This would probably be a converged Ritz value, and hence an eigenvalue of B, the preconditioned matrix AK^{-1}. Since the goal of preconditioning is to make B resemble the identity, it can be expected that B has indeed eigenvalues close to one.

Observation 6.4 The success of the pre-processing of GMRES by K-preconditioned Richardson iteration depends on the quality of the preconditioner. If $B := AK^{-1}$ has m eigenvalues close to one, and if the Ritz values of plain GMRES would converge to those eigenvalues *before* the approximation to the solution of the linear system $Ax = b$ has been found with sufficient accuracy, then the use of m pre-processing steps is suggested to save memory and computational costs.

Naturally, one does not know such detailed information on beforehand. Nevertheless, as our computational examples suggest, and in particular for very large matrices, it does not seem seldom that there are indeed converged Ritz values close to one. Moreover, in cases where pre-processing does not significantly improve the procedure, it does not seem to harm it either.

Remark 6.5 The analysis is not restricted to the classical iterations considered so far. As a matter of fact, replacing $I - B$ by $zI - B$ shows that any complex value z can be used to obtained the same effect. Good *a priori* guesses z for other eigenvalues of B or for Ritz values would therefore lead to successful pre-processings as well.

7 More experiments with pre-processing

We will now proceed with presenting a few more experiments that will illustrate the success of preprocessing. We employ the same presentation of results as was done in Figure 8, so we show convergence histories of GMRES pre-processed with k classical iteration steps for relevant k. Then we show the amount of flops against k, and the total number of iterations against k. In some of the pictures it may seem as if convergence graphs intersect others. This is not the case. Almost horizontal lines are caused by a sequence of short horizontal lines each belonging to a different convergence graph. For details on the testmatrices, see the appendix and [11].

7.1 SHERMAN3, non-Hermitian, size 5005, with ILU(0.001)

Our first experiment is with the SHERMAN3 matrix from Figure 3. From the flat start of the graph in the right picture we conclude that until eleven preprocessing steps, the end of the twelve convergence graphs (nearly) coincide. From the middle picture this is also clear by the reduction of the amount of flops until about $k = 14$.

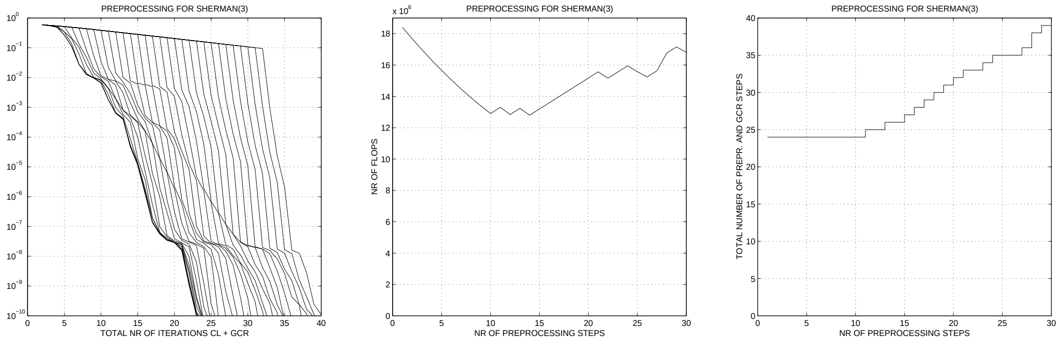


Figure 9. Preprocessing SHERMAN3. Convergence histories, number of flops, and total number of iterations. The y-range of center and right picture starts at $y = 0$.

Also noteworthy and visible from the right picture is that plain GMRES requires 24 iterations, while GMRES pre-processed with 30 classical steps requires only 9.

7.2 SHERMAN5, non-Hermitian, size 3312, with ILU(0.05)

Our second experiment is with the SHERMAN5 matrix from Figure 4. The flat start of the graph in the right tells us that until fifteen preprocessing steps, the end of the sixteen convergence graphs (nearly) coincide. The middle picture shows a reduction of the amount of flops at $k = 30$ of almost a factor two.

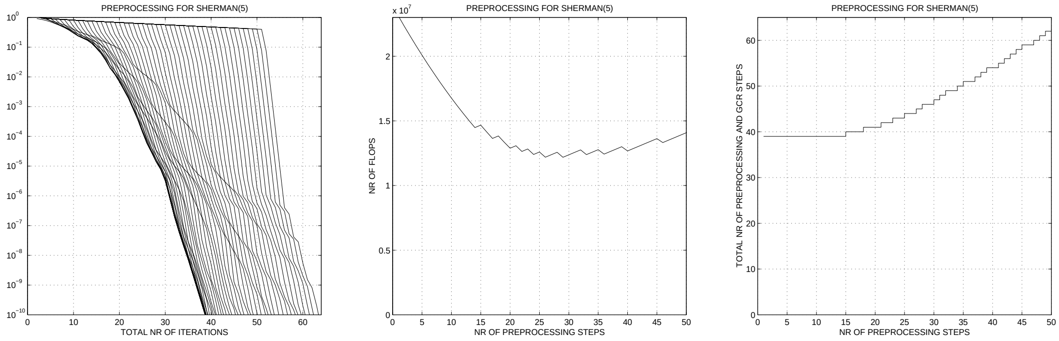


Figure 10. Preprocessing SHERMAN5. Convergence histories, number of flops, and total number of iterations.

The amount of GMRES steps is reduced from 39 in the un-pre-processed case to 13 in the case with 50 preprocessing steps.

7.3 SAYLR4, symmetric indefinite, size 3564, with ILU(0.1)

Third experiment is with the SAYLR4 matrix from Figure 4. The preprocessing appears still effective, although the flat part of the right picture is only seven preprocessing steps long. Again, there is a reduction of the amount of flops of almost a factor two.

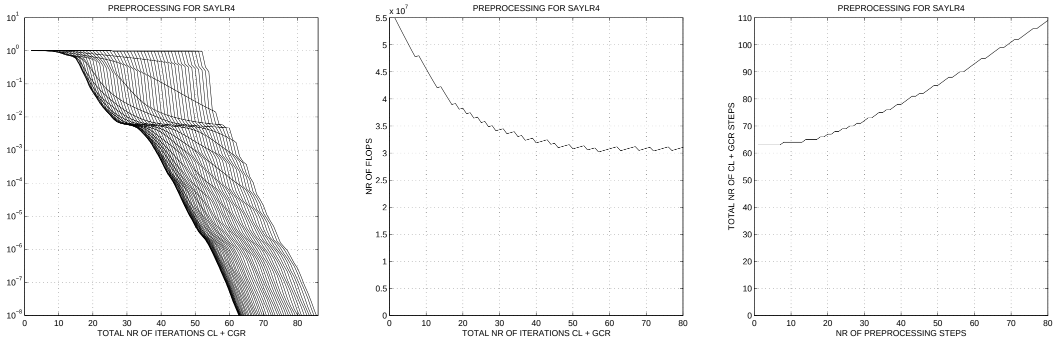


Figure 11. Preprocessing SAYLR4. Convergence histories, number of flops, and total number of iterations. The y-range of center and right picture starts at $y = 0$.

In this experiment we were did not use a Krylov method for symmetric matrices like MinRes because we could not find a suitable preconditioner for the symmetric case. It should be noted that MatLab's Incomplete LU decomposition for symmetric matrices gives factors L and U such that $A \approx LU$ even though the product LU is not always symmetric. The non-symmetric preconditioning caused MinRes to stagnate, and that is why we used GMRES. The amount of GMRES steps is reduced from 64 in the un-pre-processed case to 30 in the case with 50 preprocessing steps.

7.4 LSHP3466, symmetric indefinite, size 3466, with ILU(0.01)

Next experiment is with the matrix LSHP3466 from Figure 5, for which the classical method ultimately diverges. Preprocessing is also in this case still effective, although there is only a small reduction of the amount of flops.

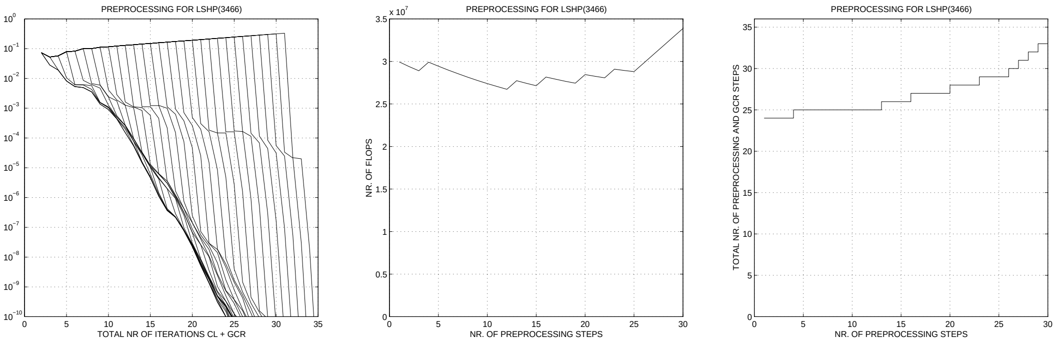


Figure 12. Preprocessing LSHP3466. Convergence histories, number of flops, and total number of iterations. The y-range of center and right picture starts at $y = 0$.

The amount of GMRES steps is however drastically reduced from 24 in the un-preprocessed case to 3 in the case with 30 preprocessing steps. The reason that this does not show in the flop count is due to the fact that matrix-vector multiplications and preconditioning solves are considerably more expensive than inner products for this matrix. Therefore it would also not have made a big difference if we could have used a Krylov subspace method for Hermitian matrices (see previous experiment).

7.5 POISSON and CONVDIFF, size 4900, with Gauss-Seidel

Last experiment is with the matrix POISSON 4900 from Figure 4. First we used Gauss-Seidel preconditioning, which asks for a method for non-Hermitian matrices. In spite of that, preprocessing is also in very effective, there is a considerable reduction of the amount of flops.

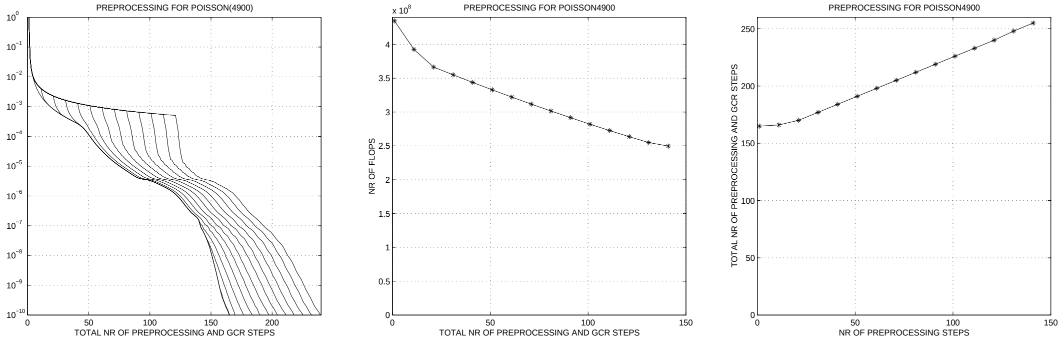


Figure 13. Preprocessing POISSON4900. Gauss-Seidel preconditioning, and GMRES.

For completion, we will also show the effect of preprocessing in the symmetric positive definite case. For that, we applied diagonal preconditioning to POISSON4900 (Jacobi iteration) and changed GMRES to Conjugate Residuals. As already remarked before, we do not expect a big gain in flops since both Jacobi and CG need one matrix vector multiplication, and CG three inner products per iteration. Our expectations are confirmed.

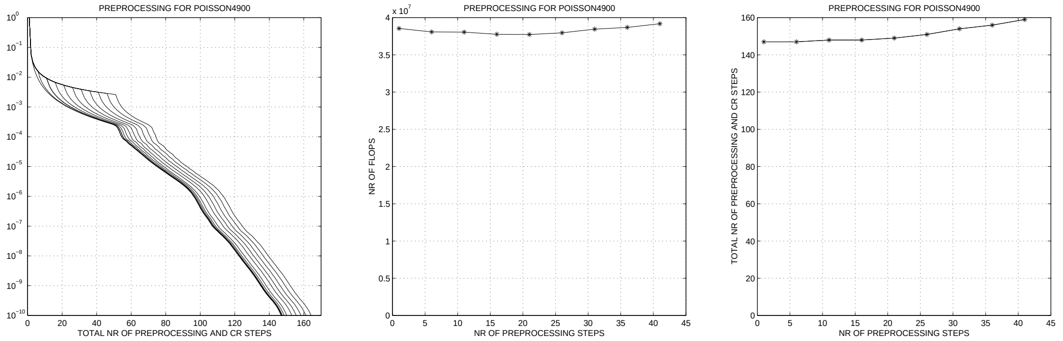


Figure 14. POISSON4900 with Jacobi preprocessing for Conjugate Residuals.

To conclude, we added to the matrix POISSON4900 an non-Hermitian perturbation due to convection resulting in a matrix CONVDIFF4900. Gauss-Seidel preconditioning was used, which resulted in a divergent preprocessing. Nevertheless, preprocessing is in this case still effective, although more than 20 steps should not be taken since then the relative reduction of the residual (from more than 1e5 to less than 1e-10) comes in the area of machine precision. So a disadvantage of a too fast diverging classical method is that the final reachable precision is lower by about the amount of increase of the norm during the preprocessing.

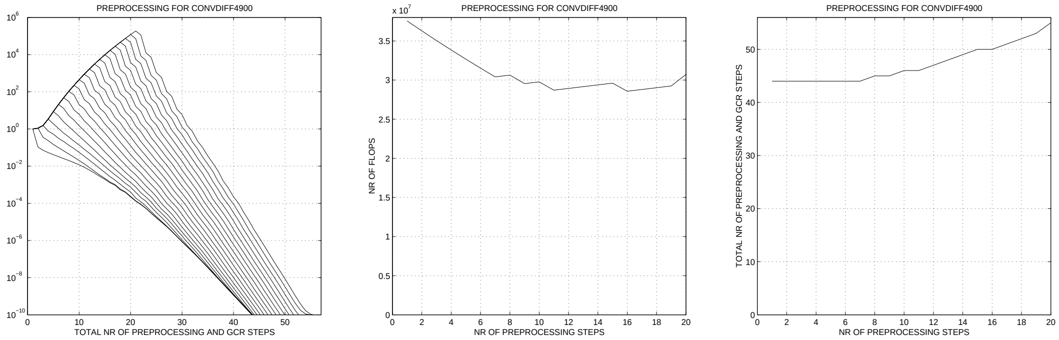


Figure 15. Preprocessing CONVDIFF4900 with a divergent classical method.

8 Conclusions

In this paper we have developed extrapolation methods for classical iteration schemes, after which we used them as inspiration for the related topic of preprocessing in Krylov subspace methods. Indeed, it all depends on your point of view whether to call a Krylov subspace method that is applied *after* a classical method an *extrapolation method* for this classical method, or to call a classical method that is applied *before* a Krylov subspace method a *preprocessing method* for the Krylov subspace method.

We have performed many numerical experiments that show the behavior of both, and for a range of different matrices arising from different applications. They also served as examples to illustrate the different algorithms presented.

Finally we have made an effort to give mathematical foundation for the phenomena that have been observed, in particular the counter-intuitive fact that sometimes, a pre-processed minimal residual method's convergence graph becomes indistinguishable close to the un-pre-processed minimal residual method's graph. This would mean that too much energy is put into building a too large Krylov subspace, in which case the preprocessing really means a big decrease of the amount of work needed to solve the system.

8.1 Additional remarks

On the whole, preprocessing and extrapolation both make use of the simple classical iterative method. It should be noted that in particular with respect to parallelism this is very interesting and rewarding. Classical methods are much better parallelizable that Krylov subspace methods, so that the gain due to preprocessing in terms of CPU-time can be much larger than in terms of floating point operations in specific cases.

Apart from using preprocessing in the context of solving linear systems, the whole idea of aiming a residual in the direction of the Krylov subspace to build can naturally be applied to Arnoldi's method for eigenvalue approximation. This might lead to improved approximations of the extremal eigenvalues. Also note, that the analysis of Section 6.2. can be done differently using ideas of the Implicitly Restarted Arnoldi (IRA) method [13, 10]. In this method, clever use is made of a relation between applying the μ-shifted QR-eigenvalue algorithm to H_k and pre-multiplication of the start-vector by $\mu I - B$. An analysis via IRA is given in [1]. The one in Section 6.2. seems easier though.

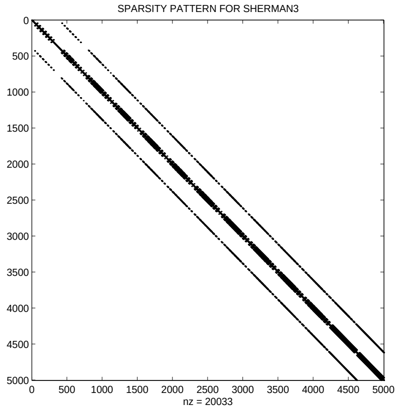
Acknowledgments

I would like to thank Gerard Sleijpen, Paul Smit [14], Bernd Fisher, Jan Modersitzki, Jan Zitko [17, 18], and Michal Křížek [9] for discussing the topic on several occasions and the hospitality of their home institutions. Moreover I am indebted to Michal Křížek for reading earlier versions of this manuscript. Finally, the support through a Fellowship of the Royal Netherlands Academy of Arts and Sciences is gratefully acknowledged.

Appendix. More about the testmatrices

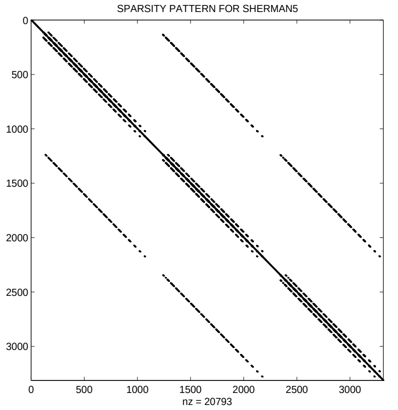
In this appendix we will present some details on the testmatrices that were used in the numerical experiments. With a single exception, they are taken from the Harwell-Boeing collection, which features as a part of Matrix Market [11] on the Internet. For all matrices, we will give size, type, number of non-zeros, a MatLab spy-plot, a two-norm condition number estimate, and an estimate for the following measure for the deviation from normality,

$$\eta(A) := \frac{\|A^*A - AA^*\|}{\|A^*A\|}. \tag{41}$$



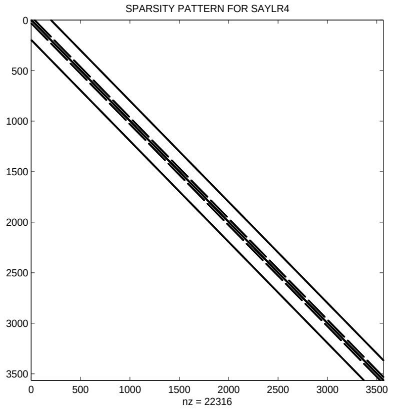
SHERMAN3: Oil reservoir simulation challenge matrix from the Harwell-Boeing collection. IMPES simulation of a black oil model.

Size: 5005 by 5005, 20033 non-zeros
Type: real, non-symmetric
Cond.nr. estimate: 6.9e+16
Estimate $\eta(A)$: 1.4e-8



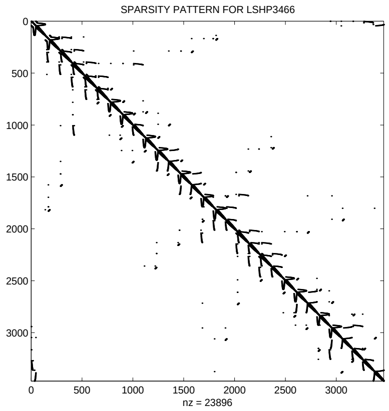
SHERMAN5: Oil reservoir simulation challenge matrix from the Harwell-Boeing collection. Fully implicit black oil model.

Size: 3312 by 3312, 20793 non-zeros
Type: real, non-symmetric
Cond.nr. estimate: 3.9e5
Estimate $\eta(A)$: 9.8e-1



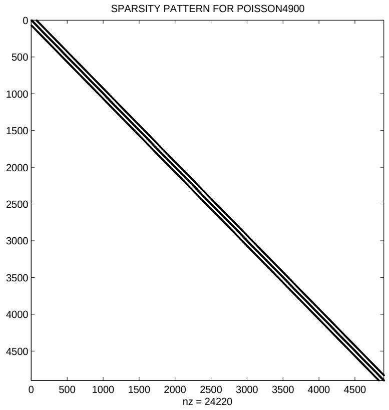
SAYLR4: Saylor's petroleum engineering/reservoir simulation matrix from the Harwell-Boeing collection.

Size: 3564 by 3564, 22316 non-zeros
Type: real, symmetric indefinite
Cond.nr. estimate: 6.9e6
Exact $\eta(A)$= 0



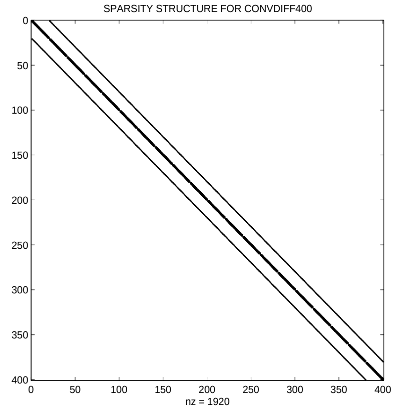
LSHP3466: Graded L-shapes patterns, matrix from the Harwell-Boeing collection.

Size: 3466 by 3466, 13681 non-zeros
Type: real, symmetric indefinite
Cond.nr. estimate: 1.2e5
Exact $\eta(A)$= 0



CONVDIFF 4900: Standard discretisation of a convection dominated convection-diffusion problem on a square using finite differences. Taken from [14].

Size: 4900 by 4900, 24220 non-zeros
Type: real, non-symmetric
Cond.nr. estimate: 3e3
Estimate $\eta(A)$= 1.3e-1



CONVDIFF 400: Standard discretisation of a convection-diffusion problem with mild convection term using finite differences. Taken from [14].

Size: 400 by 400, 1920 non-zeros
Type: real, non-symmetric
Cond.nr. estimate: 1.3e2
Estimate $\eta(A)$= 6.1e-2

References

[1] J.H. Brandts (2000) *Explanation of a phenomenon witnessed in pre-processed GMRES.* ENUMATH 99 - Proceedings of the 3rd European Conference on Numerical Mathematics and Advanced Applications, Jyväskylä, Finland, July 26-30, 1999, ed. by P. Neittaanmäki, T. Tiihonen and P. Tarvainen, World Scientific, Singapore, pp. 440-448.

[2] C. Brezinski and M.R. Zaglia, Extrapolation methods. Theory and practice, *North Holland, Amsterdam*, 1991.

[3] G.H. Golub and C.F. van Loan, Matrix computations, third edition, *The John Hopkins University Press*, 1996.

[4] G.H. Golub and H.A. van der Vorst, *Closer to the solution: iterative linear solvers*, in I.S. Duff and G.A.Watson (eds), The State of the art in numerical analysis, pp 63–92, Clarendon Press, Oxford, 1997.

[5] W. Hackbusch, Iterative solution of large sparse systems of equations *Applied Mathematical Sciences*, Vol. 95, Springer-Verlag, New York, U.S.A., 1993.

[6] I.C.F. Ipsen, *Expressions and bounds for the GMRES residual*, BIT, 38(2), pp. 101-115, 1998.

[7] N.J. Higham, Accuracy and stability of numerical algorithms SIAM, 1996.

[8] K. Jbilou and H. Sadok, *Analysis of some vector extrapolation methods for solving systems of linear equations*, Numer. Math. 70, pp. 73–89, 1995.

[9] M. Křížek, L. Liu and P. Neittaanmäki, *Post-processing of Gauss-Seidel iterations*, *to appear in* Numerical Linear Algebra and Applications.

[10] R.B. Lehouqc, Analysis and Implementation of an Implicitly Restarted Arnoldi Iteration, *Ph.D. thesis, Rice University, Houston, Texas.*

[11] Matrix Market, *http://math.nist.gov/MatrixMarket/*

[12] A. Sidi, *Extrapolation vs. projection methods for linear systems of equations*, J. Comp. Appl. Math. 22, pp. 71-88, 1988.

[13] D.C. Sorensen, *Implicit Application of Polynomial Filters in a k-step Arnoldi Method*, SIAM J. Matrix Anal. Appl. Vol 13, pp. 357–385, 1992.

[14] P. Smit, Numerical analysis of eigenvalue algorithms based on subspace iterations. *PhD thesis, Catholic University Brabant, Netherlands*, 1997.

[15] L.N. Trefethen, *Pseudo-spectra of matrices*, in D.F. Griffiths and G.A. Watson, Numerical Analysis 1991, Longman, pp. 234–266, 1992.

[16] H.A. van der Vorst and C. Vuik, *The superlinear convergence behaviour of GMRES*, J. Comp. Appl. Math. 48, pp 327–341, 1993.

[17] J. Zitko, *Combining the preconditioned conjugate gradient method with a matrix iterative method*, Appl. Math. Vol 41, No. 1, pp. 19–39, 1996.

[18] J. Zitko, *Using successive approximations for improving the convergence of the GMRES method*, Appl. Math. Vol 43, No. 5, pp. 1–19, 1998.