Analysis of Polynomial Preconditioners

Ferdinand Vanmaele
Universität Heidelberg
Email: Vanmaele@stud.uni-heidelberg.de

***Abstract*—We explore solving large systems of linear equations using the GMRES method, and study the effect of polynomial preconditiong for this method. Besides improving convergence for difficult problems, polynomial preconditioning allows us to reduce the amount of required scalar products, at the cost of more SpMV (sparse matrix-vector multiplication) operations. Preconditioning thus improves performance when solving such systems on massively parallel systems, where communication costs of scalar products can be a bottleneck.**

***Index Terms*—Krylov subspaces, GMRES, polynomial preconditioning, communication-avoiding**

I. Introduction

Systems of linear equations $Ax = b$, where A is a sparse matrix of high dimension n, are widely solved in scientific computing through iterative methods, that is, by finding a succession of approximations from an initial value [2, Chapter 6]. This is in contrast to direct methods, which are based on the initial factorization of the coefficient matrix and give the result in a number of exactly predictable operations. The disadvantage of direct methods lies in a high number of arithmetic operations, and in the memory footprint of storing matrices that were initially sparse [3, Example 3.3].

GMRES (Generalized Minimum Residual) method is an iterative method suitable for a wide range of linear systems, including indefinite and nonsymmetric systems. In every step, it minimizes the norm of the residual vector $\|b - A\tilde{x}\|$ over a Krylov subspace. Constructing the Krylov subspace is done through a Gram-Schmidt orthogonalization procedure using inner products. On parallel and distributed systems, computing inner products requires global reduction operations involving all processes. In comparison, sparse matrix-vector multiplications only need to exchange data between neighbors [2, 14.5].

To address this issue, we consider preconditioned systems $M^{-1}Ax = M^{-1}b$, where M^{-1} is a (typically low degree) matrix polynomial. This preconditioned system has the same solution as the original system $Ax = b$, but may have faster convergence. Achieving a certain tolerance for the residual thus occurs in less GMRES steps (inner products), at the expense of more matrix-vector multiplications in every step. We discuss a polynomial which does not require knowledge of the spectrum of the matrix A. Finding the coefficients of M^{-1} requires a (possibly dense) matrix-matrix multiplication. This cost can be controlled by the degree of the polynomial, depending on the difficulty of the problem, making it more granular than classical preconditioners such as ILU decomposition.

II. Related Work

Saad [1] introduced the GMRES method for systems in which the coefficent matrix A is nonsymmetric or indefinite. Indefinite systems commonly arise, for example, in mixed finite element methods for solving problems in fluid and solid mechanics [5], [6].

Since then, the algorithms have been included in general textbooks on numerical linear algebra and scientific computing [3], [4], [2]. Implementations are available in MATLAB and scientific computing libraries such as GSL and TRILINOS. Liu [8] discusses polynomial preconditioning with GMRES and introduces a way to find the coefficients without knowledge of the matrix spectrum. Loe et. al. [9] conducted further experiments on (shared-memory) parallel hardware with TRILINOS.

Besides polynomial preconditioning, Block Krylov methods also help reduce the effect of inner products on communication. Saad [3, 6.12] put forward an overview and gives further references.

After revisiting theoretical results on Krylov subspaces and GMRES in sections III, IV and V, we introduce polynomial preconditioning and its advantages in Chapter VI. In section VII we test the impact of polynomial preconditioning on convergence for several test problems.

III. Krylov subspaces

We will look at iterative methods for solving the linear system $Ax = b$ of dimension n, where A is an invertible square matrix, with approximate solutions in Krylov subspaces. These methods can be applied to either real or complex matrices. We introduce Krylov subspaces and construct an orthonormal basis. Solving a least squares problem to find the minimum residual for vectors in this subspace leads to the GMRES method.

Definition 1 (Krylov subspace [3, 6.2])**.** The **Krylov subspace** of order m, denoted by $\mathcal{K}_p$, is the vector space spanned by $v \in \mathbb{R}^n$ and the first $m-1$ powers of A:

$$\mathcal{K}_m(A, v) = \text{span}\{v, Av, A^2v, \ldots, A^{m-1}v\}.$$

The Krylov subspaces form an increasing family of subspaces, with dimension bounded by n. It is the subspace of all vectors in $\mathbb{R}^n$ which can be written as $x = p(A)v$, where p is a polynomial of degree not exceding $m-1$.

In practice, constructing the Krylov subspaces amounts to determining their basis. The power basis $(v, Av, A^2v, \ldots, A^{m-1}v)$ converges to (a multiple of) an eigenvector that corresponds to the largest eigenvalue of A, in absolute value [2, 10.2]. Therefore we construct an orthonormal basis $\{v_1, v_2, \ldots, v_m\}$ as follows.

Arnoldi's method applies the (modified) Gram-Schmidt orthonormalization procedure to the vectors obtained by successive products of the matrix A. Algorithm 1 [3, 6.3.2] gives a practical implementation.

Algorithm 1 Arnoldi-Modified Gram-Schmidt

Choose a vector v_1 of norm 1
For $j = 1, 2, \ldots, m$ Do:
  Compute $w_j := Av_j$
  For $i = 1, \cdots, j$ Do:
    $h_{ij} = (w_j, v_i)$
    $w_j := w_j - h_{ij}v_i$
  EndDo
  $h_{j+1,j} = \|w_j\|_2$. If $h_{j+1,j} = 0$ Stop
  $v_{j+1} = w_j/h_{j+1,j}$
EndDo

The modified Gram-Schmidt procedure has better numerical properties than the classical procedure [3, 6.3.2], but requires additional synchronization points when implemented on a parallel computer. Additional methods such as DCGS2 were introduced to address these issues [10]. Householder transformations, which are relatively costly but stable, were also proposed [7, 2.3.4].

The following important relations hold for the Arnoldi basis:

$$AV_m = V_{m+1}\overline{H}_m, \qquad V_m^T A V_m = H_m,$$

where V_m denotes the $n \times m$ matrix with orthonormal column vectors $v_1, \ldots, v_m$, $\overline{H}_m$ the $(m+1) \times m$ Hessenberg matrix with nonzero entries h_{ij}, and H_m the matrix obtained from $\overline{H}_m$ by deleting its last row: [3, Proposition 6.5]

$$H_m = \begin{pmatrix} h_{1,1} & h_{1,2} & \cdots & \cdots & h_{1,m} \\ h_{2,1} & h_{2,2} & & & \vdots \\ & h_{3,2} & \ddots & & \vdots \\ & & \ddots & \ddots & \vdots \\ & & & \ddots & h_{m,m-1}h_{m,m} \end{pmatrix}.$$

IV. The GMRES method

Let $r_0 = b - Ax_0$ denote the residual for an initial approximation x_0. Let $\mathcal{K}_m := \mathcal{K}_m(A, r_0)$. We seek to minimize the residual on the affine subspace $x_0 + \mathcal{K}_m$ of dimension m.

Definition 2. Choose $v_1 = r_0/\|r_0\|$ for the Arnoldi basis, and let $\beta := \|r_0\|$. Any vector in $x_0 + \mathcal{K}_m$ can be written as $x = x_0 + V_m y$, where $y \in \mathbb{R}^m$. By the above relation, the objective is given as: [3, 6.5.1]

$$\begin{aligned} J(y) &= \|b - A(x_0 + V_m y)\|_2 = \|r_0 - AV_m y\|_2 \\ &= \|\beta v_1 - V_{m+1}\overline{H}_m y\|_2. \end{aligned}$$

By orthonormality of V_{m+1} column vectors, $J(y) = \|\beta e_1 - \overline{H}_m y\|$. The **GMRES approximation** is then the unique vector of $x_0 + \mathcal{K}_m$ which minimizes J.

This results in the following algorithm: [3, Algorithm 6.9]

Computing y_m is an $(m+1) \times m$ matrix least squares problem with Hessenberg structure that can be solved via QR factorization in the usual manner, at the cost of $\mathcal{O}(m^2)$ operations. To save further work, rather than construct QR factorizations of the successive matrices $\overline{H}_1, \overline{H}_2, \ldots$ independently, we can use an updating process to get the QR factorization of $\overline{H}_m$ from that of $\overline{H}_{m-1}$. Doing so requires a single **Givens rotation** and $\mathcal{O}(m)$ work [4, Mechanics of GMRES].

Algorithm 2 GMRES

Compute $r_0 = b - Ax_0$, $\beta := \|r_0\|_2$, and $v_1 := r_0/\beta$
For $j = 1, 2, \ldots, m$ Do:
  $\langle$ step n of Arnoldi iteration, Algorithm 1 $\rangle$
EndDo
Define the $(m+1) \times m$ Hessenberg matrix $\overline{H}_m = \{h_{ij}\}_{1 \le i \le m+1, 1 \le j \le m}$
Compute y_m the minimizer of $\|\beta e_1 - \overline{H}_m y\|_2$ and $x_m = x_0 + V_m y_m$

This approach also enables obtaining the residual norm of the approximate solution without computing x_k, thus allowing to decide when to stop the algorithm inexpensively [1, 3.2]. Saad [1], [3] gives the detailed implementation.

Consider now the algorithm from a practical viewpoint. As m increases, the computational cost increases at least as $\mathcal{O}(m^2 n)$ because of the Gram-Schmidt orthogonalization. The memory cost increases as $\mathcal{O}(mn)$. For large n this limits the largest value of m that can be used. One remedy is to restart the algorithm periodically [3, 6.4.1, 6.5.5]. This is the **restarted GMRES** or GMRES(m) algorithm. Note that GMRES(m) does not necessarily converge, unless the matrix A is positive definite [1, 3.4].

V. Convergence of GMRES

Because $\|r_m\|$ is as small as possible for the subspace $\mathcal{K}_m$, by enlarging $\mathcal{K}_m$ to the space $\mathcal{K}_{m+1}$ we can only decrease the residual norm, or at worst leave it unchanged. Thus we observe that GMRES converges monotically: $\|r_{m+1}\| \le \|r_m\|$. After n steps (for an $n \times n$ problem), we have $\mathcal{K}_n = \mathbb{C}^n$ and the full GMRES algorithm is guaranteed to converge. However, for GMRES to be useful, it must converge to satisfactory precision in $n \ll m$ steps [4, p.270].

To obtain more useful information about convergence, we must turn to the following problem. The iterate $x_m \in x_0 + \mathcal{K}_m$ can be written as $x_m = x_0 + q(A)r_0$, where q is a polynomial of degree $m-1$.

Furthermore, as x_m minimizes $\|r_m\| = \|b - Ax_m\|$ in the affine subspace $x_0 + \mathcal{K}_m$, we have $x_m = x_0 + q_m(A)r_0$, where q_m is the polynomial that solves: [3, Lemma 6.31]

$$\begin{aligned} \|r_m\| = \|b - A(x_0 + q_m(A)r_0)\| &= \min_{q \in P_{m-1}} \|(I - Aq(A))r_0\| \\ &= \min_{p \in P_m, p(0)=1} \|p(A)r_0\|. \end{aligned}$$

The polynomial p is determined by GMRES. We can show:

Theorem 3 ([3, Proposition 6.32])**.** *Suppose A is diagonalizable, satisfying $A = X\Lambda X^{-1}$ with $\Lambda = diag\{\lambda_1, \lambda_2, \ldots, \lambda_n\}$*

is the diagonal matrix of eigenvalues. We have the following upper bound on the residual:

$$\|r_m\| \leq \kappa_2(X) \min_{p\in\mathbb{P}_m, p(0)=1} \max|p(\lambda_i)|.$$

This theorem can be summarized in words as follows. If A is not too far from normal in the sense that $\mathcal{K}(X)$ is not too large, and if properly normalized degree m polynomials can be found whose size on the spectrum $\Lambda(A)$ decreases quickly with m, then GMRES converges quickly [4, p.271].

Example 4. Let $n = 4000$, with entries sampled from $\mathcal{N}(2, 0.5/\sqrt{n})$. The eigenvalues are roughly uniformly distributed in the disk of radius $\frac{1}{2}$ centered at $z = 2$. $\|p(A)\|$ is then approximately minimized by $p(z) = (1 - z/2)^m$, with a convergence rate of 4^{-m} [4].

Corollary 5 ([1])**.** *Suppose A is positive real with symmetric part $M = \frac{1}{2}(A + A^T)$. We have the following upper bound on the residual:*

$$\|r_m\| \leq \left(1 - \frac{\lambda_{\min}(1/2(A^T + A))^2}{\lambda_{\max}(A^T A)}\right)^{m/2} \|r_0\|.$$

In particular, if A is symmetric positive definite, we have:

$$\|r_m\| \leq \left(\frac{\kappa(A)^2 - 1}{\kappa(A)^2}\right)^{m/2} \|r_0\|,$$

where $\kappa = \|A\|\|A^{-1}\|$ denotes the condition number of A.

VI. Polynomial preconditioning

For any nonsingular $n\times n$ matrix M, the system $M^{-1}Ax = M^{-1}b$ has the same solution as the nonsingular system $Ax = b$. Solving this system however depends on the properties of $M^{-1}A$, and may require fewer steps to converge than the original system [3, 10.1]. We call M the **preconditioner**.

M should meet the following requirements:

- M lies between $M = A$ (for the equally hard to solve system $x = A^{-1}b$) and $M = I$ (for the original system $Ax = b$).
- M^{-1} should be inexpensive to apply to an arbitrary vector.
- M is "close enough" to A, in the sense that the preconditioned system $M^{-1}Ax = M^{-1}b$ converges more quickly than the system $Ax = b$. Specifically, we want $\|I - M^{-1}A\|$ to be small and the eigenvalues of $M^{-1}A$ close to 1, or clustered around some other value [4, Lecture 40].

With **polynomial preconditioning**, we approximate $M^{-1} = s(A)$ directly with a polynomial s, typically of low degree d. The original system is then replaced by the preconditioned system $s(A)Ax = s(A)b$.

In this system, the matrix $s(A)$ does not need to be formed explicitly: $s(A)v$ can be computed for any vector v from a sequence of matrix-by-vector products [3, 12.3]. In particular, **Horner's method** computes $s(A)v$ with d matrix-vector products:

$$\begin{aligned} s(A)v &= s_{d+1}A^d v + s_d A^{d-1}v + \cdots + s_1 v \\ &= A(A\cdots(s_{d+1}Av + s_d v) + \cdots + s_2 v) + s_1 v. \end{aligned}$$

Note that a polynomial preconditioner can be combined with other preconditioners $\tilde{M}$, such as ILU(k), by solving the system:

$$s(\tilde{M}^{-1}A)\tilde{M}^{-1}A = s(\tilde{M}^{-1}A)\tilde{M}^{-1}b.$$

We now compute a polynomial preconditioner related to the GMRES polynomial. Unlike methods such as **Chebyshev polynomials** or **least squares polynomials**, which use estimates of the spectrum of A [3, 12.3], knowledge of the spectrum is required [8].

GMRES computes a polynomial $p(A)$ corresponding to a minimal residual $\|p(A)r_0\| = \|(I - Aq(A))r_0\|$. We want to compute the coefficients of $q(A)$ for a given degree d instead. Denote this polynomial by $M^{-1} = s(A)$ as above, and let $v_0 \neq 0$. We solve:

$$\min\|(I - M^{-1}A)v_0\| = \min_{q\in\mathbb{P}_d} \|v_0 - Aq(A)v_0\|$$

by building a power basis $Y = \{v_0, Av_0, \ldots, A^d v_0\}$, and solving the normal equations

$$(AY)^T AYy = (AY)v_0.$$

The elements of y are the coefficients of M^{-1}. Note the following:

- The method becomes unstable as the columns of Y lose linear independence, but its results are generally sufficient for low-degree (up to $d = 10$) polynomials [8].
- The coefficients of the polynomial depend on the choice of v_0. Using a randomly chosen vector may be preferable in this case [9].
- The matrix AY is a (sparse) matrix of dimension $n\times(d+1)$, and takes $d+1$ matrix-vector products to compute. The product $(AY)^T AY$ is of dimension $(d+1)\times(d+1)$ and requires a sparse matrix-matrix multiplication. With $d \ll n$, the system $(AY)^T AYy = (AY)v_0$ can then be solved through a direct method such as LU decomposition.

VII. Numerical experiments

We consider the following problems. Some of these problems are very difficult for GMRES to solve, and, considering the small dimension n, may be better suited for direct solvers. They nonetheless give a good indication of the difficulties involved when using GMRES.

For every problem, the right-hand side is sampled from $\mathcal{N}(0, 1)$, the real normal distribution of mean 0 and standard deviation 1. Orthonormalization was done using the Modified Gram-Schmidt procedure. The vector for computing the power basis Y is sampled from $\mathcal{N}(-1, 1)$.

1) The **Circle Eigenvalue Matrix** of size $n = 2000$, a block diagonal matrix with 2×2 blocks:

$$\begin{pmatrix} 1+\cos(\alpha) & \sin(\alpha) \\ -\sin(\alpha) & 1+\cos(\alpha) \end{pmatrix},$$

$\alpha \in 2k\frac{\pi}{n},\ k \in \{0, 2, \ldots, n-2\}$. All eigenvalues of this matrix are on the unit circle on the complex plane with center $(1, 0)$ [8]. The condition number is estimated at 6.3662×10^3.

2) The **Bidiagonal Matrix** of size $n = 5000$ with diagonal elements 0.1, 0.2, 0.3, ..., 0.9, 1, 2, 3, ..., 4990, 4991 (*BiDiag1*). The superdiagonal elements are all set to 0.2 [8]. The symmetric part is positive definite, and the condition number estimated at 8.4589×10^4.
3) As problem 2, but the diagonal elements are set to 10, 11, 12, ..., 5009 (*BiDiag2*). The condition number is estimated at 5.0137×10^3.
4) **S1RMQ4M1** (Matrix Market), a real symmetric positive definite matrix of size $n = 5489$. The matrix has high condition number, estimated at 3.21×10^6 by Matrix Market.
5) **E20R0100** (Matrix Market), a real non-symmetric indefinite matrix of size $n = 4241$. The matrix has high condition number, estimated at 2.15×10^{10} by Matrix Market. This matrix serves as a counter-example for classical preconditioners such as ILU, where their application may result in a worse condition number.
6) **SHERMAN5** (Matrix Market), a real non-symmetric matrix of size $n = 3312$. The condition number is estimated at 3.9×10^5 by Matrix Market. This matrix serves as an example where polynomial preconditioning can worsen convergence.

We use GMRES(20) for the Bidiagonal Matrix, GMRES(50) for SIRMQ4M1, and GMRES(100) for E20R0100 and the Circle Eigenvalue Matrix. The maximum of iterations is 5000 for all problems except for SHERMAN5, which uses a maximum of 20,000 iterations. Tolerance is set to $\varepsilon = 10^{-8}$ for the relative residual $\frac{\|b-Ax_m\|}{\|b\|}$. Polynomials of degree 3, 5, 7, and 10 are used for most problems. Implementation was done in MATLAB, based on `gmres.m` from the Netlib Repository.

We observe the following:

- For all problems considered, preconditioning reduced the condition number, typically by one order of magnitude for polynomials of degree 10 (Figure 3).
- Apart from the Circle Eigenvalue Matrix, the polynomial preconditioner had very small coefficients for its higher-order terms (compare table II, III). This effect may be due to the ill-conditioned problem of computing coefficients via the normal equations with a power basis [9, III]. This corresponded with LAPACK reporting a very small RCOND for these problems ($< 10^{-17}$ for E20R011, $< 10^{-25}$ for S1RMQ4M1).
- The spectrum of S1RMQ4M1 preconditioned with degree 7 or 9 polynomial had negative eigenvalues. These eigenvalues were outliers, and the behavior was not observed for other degree polynomials. One may however expect that $M^{-1}A$, while symmetric, is in general not positive definite.
- For each problem, the preconditioner mapped the spectrum to cluster eigenvalues around 1. The effect was especially noticeable for E20R0100 and degree 10 polynomials.
- For the Bidiagonal Matrix and S1RMQ4M1, it sufficed to use degree 3 polynomials to ensure convergence to the given tolerance (Figure 5). More difficult problems required higher degree polynomials. In particular, convergence for E20R0100 stalled for degree < 10 polynomials (Figure 6). With improved convergence, the amount of inner products decreased accordingly (Figure 8).
- Apart from the Bidiagonal Matrix, the amount of SpMVs increased by ~3x (for S1RMQ4M1) up to ~6x (for the Circle Eigenvalue Matrix). For E20R0100, there was an up to ~8x increase, but with fast convergence of a degree 10 preconditioner, latter only saw an increase of ~1.25x.
- Results for SHERMAN5 varied widely between samples of v_0 of b. Degree 7 polynomials as suggested in Loe [9] improved convergence for some samples, but worsened it for others. Degree 12 and 15 polynomials had more consistent results, but were still not guaranteed to improve convergence.

CONCLUSION

Polynomial preconditioners can be used along with standard preconditioning, and are effective to implement on parallel systems as a series of sparse matrix-vector products. We have introduced a polynomial preconditioner related to the minimum residual (GMRES) polynomial.

To compute the coefficient of the polynomial, no knowledge of the spectrum of A is required. The method, which bases on computing a power basis $(A^i v_0)$ is however ill-conditioned, which limits the degree of the polynomial.

Applying this preconditioner to several difficult problems, including E20R0100, shows it can be used to significantly improve convergence of GMRES. In particular, on parallel systems, synchronization points from inner products in each GMRES step are reduced. Care should be taken for problems such as SHERMAN5, where effectiveness varies strongly between the random vector samples v_0 used for computing the preconditioner.

REFERENCES

[1] Y. SAAD, M. H. Schultz, *GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems,* SIAM J. Sci. Stat. Comput. Vol. 7, No. 3, July 1986.
[2] F. MAGOULES, F. ROUX, G. HOUZEAUX, *Parallel Scientific Computing,* Wiley, 2015.
[3] Y. SAAD, *Iterative Methods for Sparse Linear Systems*, SIAM, 2003.
[4] L. N. TREFETHEN, D. BAU, *Numerical Linear Algebra*, SIAM, 1997.
[5] M. BENZI, G. H. GOLUB, J. LIESEN, *Numerical solution of saddle point problems,* Acta Numerica (2005), pp. 1-137.
[6] M. ROZLOŽNÍK, *Saddle-Point Problems and Their Iterative Solution*, Birkhäuser, 2018.
[7] R. BARRET et. al., *Templates for the Solution of Linear Sytems: Building Blocks for Iterative Methods*, SIAM, 1994.
[8] Q. LIU, R. B. MORGAN, W. WILCOX, *Polynomial Preconditioned GMRES and GMRES-DR*, SIAM J. Sci. Comput. Vol. 37, No. 5, pp. S407-S428, 2015.
[9] J. A. LOE, H. K. THORNQUIST, E. G. BOMAN, *Polynomial Preconditioned GMRES to Reduce Communication in Parallel Computing*, arXiv, Oct 2019.
[10] V. HERNANDÉZ, J. E. ROMÁN, A. TOMAS, *A Parallel Variant of the Gram-Schmidt Process with Reorthogonalization*, Proceedings of the International Conference Parco 2005, 13-16 September 2005.

APPENDIX

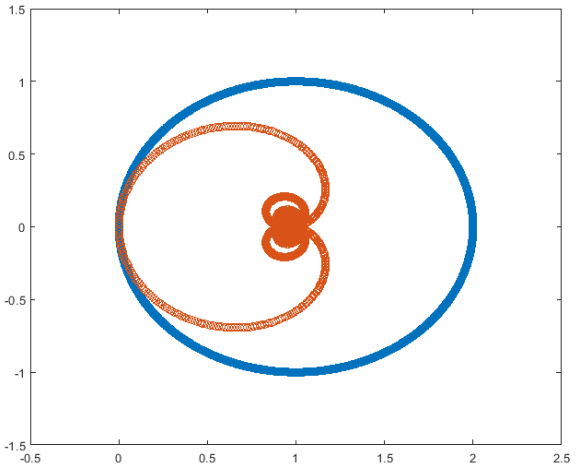

Fig. 1. Spectrum of A (blue) and $M^{-1}A$ (orange, $d = 10$) for the Circle Eigenvalue Matrix

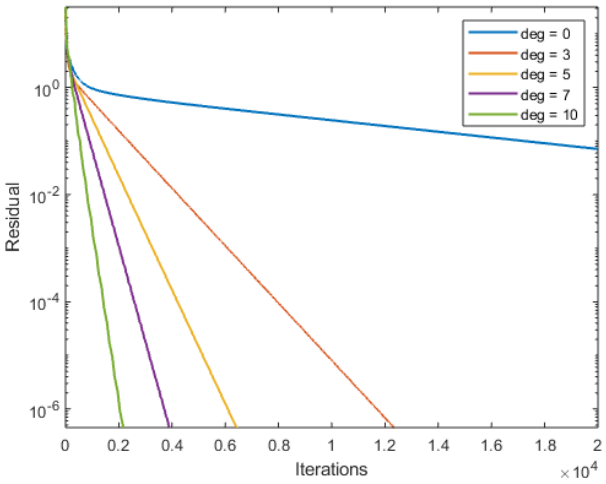

Fig. 4. Convergence of the Circle Eigenvalue Matrix, GMRES(100)

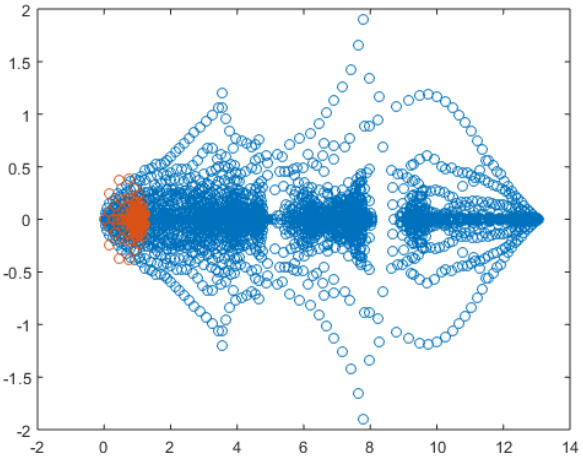

Fig. 2. Spectrum of A (blue) and $M^{-1}A$ (orange, $d = 10$) for E20R0100

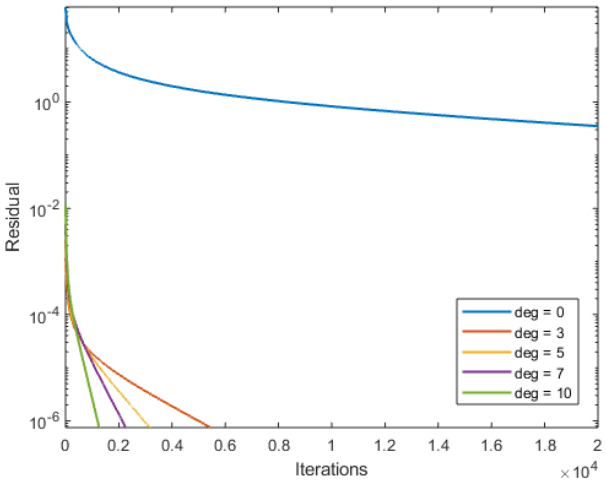

Fig. 5. Convergence of S1RMQ4M1, GMRES(50)

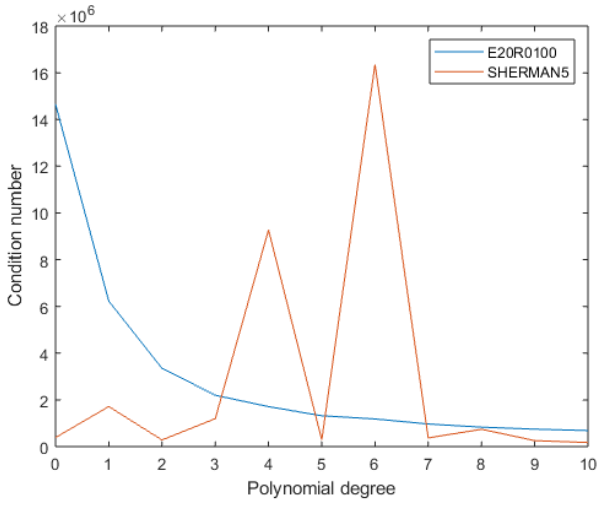

Fig. 3. Condition numbers for E20R0100 and SHERMAN5

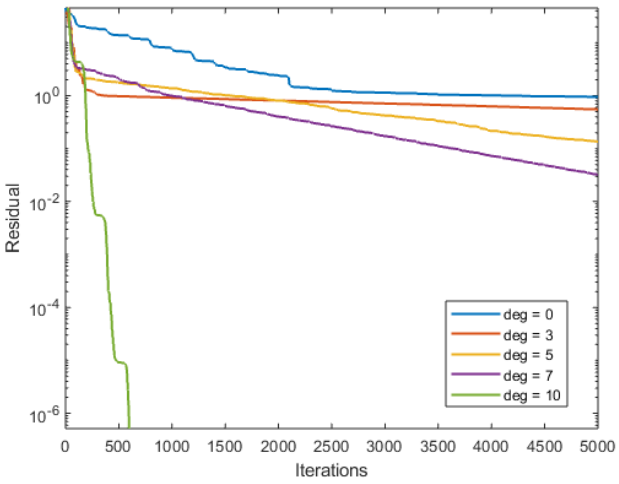

Fig. 6. Convergence of E20R0100, GMRES(100)

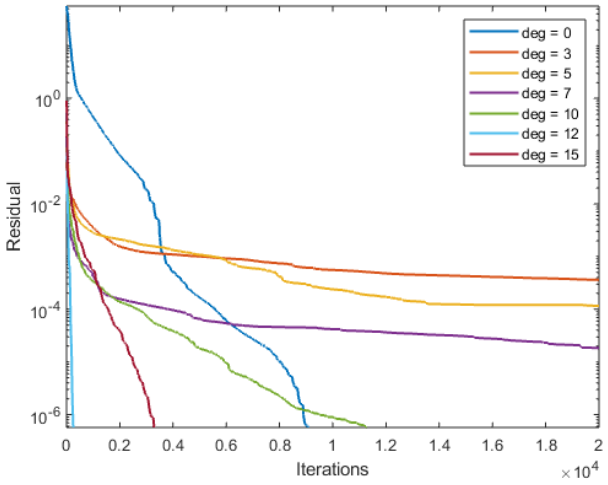

Fig. 7. Convergence of SHERMAN5, GMRES(100)

deg 3	deg 5	deg 7	deg 10
-0.1945	-0.1376	-0.1084	0.0783
0.9805	0.9684	0.9778	-0.9440
-1.9805	-2.9234	-3.9207	5.2195
2.0000	4.9100	9.1758	-17.4999
	-4.9550	-13.8125	39.6248
	3.0000	13.8695	-63.8276
		-9.2898	74.9846
		4.0000	-64.7198
			40.7217
			-18.2159
			5.5000

TABLE I
COEFFICIENTS FOR CIRCLE EIGENVALUE MATRIX PRECONDITIONER

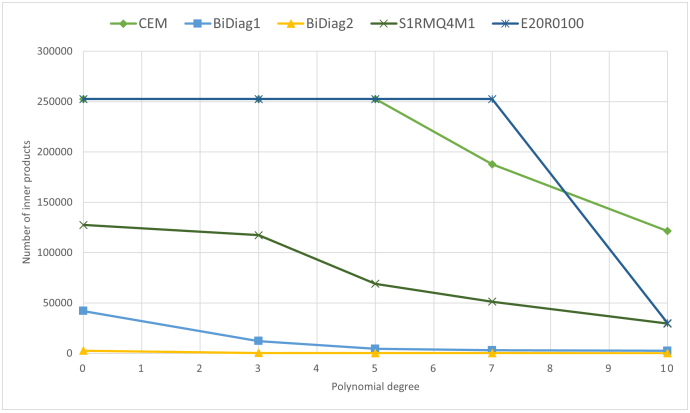

Fig. 8. Number of inner products for $\varepsilon = 10^{-8}$ and a maximum of 5000 iterations

deg 3	deg 5	deg 7	deg 10
−0.0012	−0.0001	−3.4021e-06	3.2267e-08
0.0337	0.0029	1.8944e-04	−2.4589e-06
−0.3126	−0.0466	−0.0044	8.1994e-05
1.0714	0.3634	0.0540	−0.0016
	−1.3908	−0.3836	0.0191
	2.2885	1.5686	−0.1526
		−3.4603	0.8125
		3.4870	−2.8310
			6.1825
			−7.7796
			4.7860

TABLE II
COEFFICIENTS FOR E20R0100 PRECONDITIONER

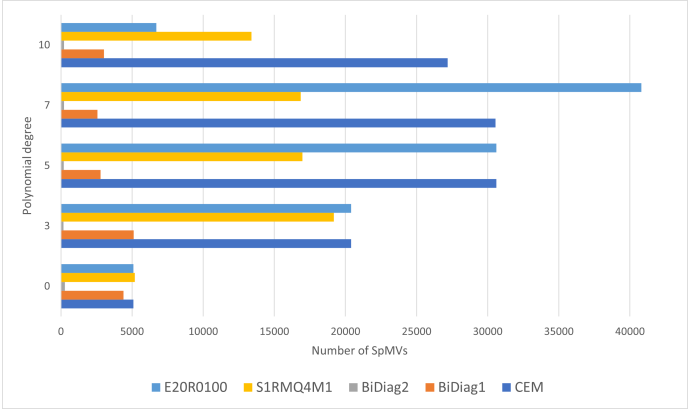

Fig. 9. Number of SpMVs for $\varepsilon = 10^{-8}$ and a maximum of 5000 iterations

deg 3	deg 5	deg 7	deg 10
3.0913e-16	1.3545e-26	6.3270e-37	−1.8078e-52
−1.3733e-10	−1.0459e-20	−6.9969e-31	2.9462e-46
2.2131e-05	3.8068e-15	4.0678e-25	−2.7248e-40
	−6.6028e-10	−1.3326e-19	1.5744e-34
	4.8290e-05	2.4458e-14	−5.8966e-29
		−2.3425e-09	1.4375e-23
		9.7378e-05	−2.2304e-18
			2.0865e-13
			−1.0575e-08
			2.3067e-04

TABLE III
COEFFICIENTS FOR S1RMQ4M1 PRECONDITIONER