PSEUDOSPECTRA OF COMPANION MATRICES AND THEIR POTENTIAL IMPACT ON POLYNOMIAL ALGORITHMS

SIMONE FLORY, JAMIIRU LUTTAMAGUZI, MICHAEL TISHER, AND HONG ZHANG*

Abstract. Multivariate polynomial systems pervade engineering design, modeling and simulation. Algorithms for solving these systems form a a fundamental computational kernel in science and engineering. In the last 15 years, new theoretical ideas and computational methodologies have emerged. The current paradigm for polynomial system solving treats symbolic processing as a preparatory step that transforms multivariate polynomial systems into generalized algebraic eigenvalue problems amenable to numerical methods, which are then passed to a numeric solution process.

The resulting eigenproblems are often notoriously sensitive to changes in the matrix entries, causing problems for available numerical software. In this work, we present how this sensitivity depends on the matrix presentation using the notion of pseudospectrum of matrix. Experiments and visualization of the pseudospectra for a set of test problems indicate that appropriate reformulations of the resulting eigenproblems give rise to locally well-conditioned representations, which then could lead to satisfying numerical approximations to solutions of multivariate polynomial systems.

1. Introduction. In scientific computation we try to develop and implement algorithms in order to solve problems that arise in applications or different parts of mathematics. Since the problems we would like to solve might be ill-conditioned, we need to adjust our methods and algorithms to the particular problem in order to get the accurate numerical solution. Sometimes we might even have to reformulate the problem in order to improve its conditioning.

In this paper, we will compare different approaches to finding a few selected numerical solutions for the so-called generalized eigenvalue problem $Ax = \lambda Bx$, where A and B are complex $m \times m$ -matrices and B is very likely singular. First of all, we will provide an example of applications where the generalized eigenvalue problem arises.

The generalized eigenvalue problems considered here arise from solving quadratic polynomial systems. Let

$$\left\{ \begin{array}{l} p_1(x_1, \dots, x_n) = 0 \\ \vdots \\ p_n(x_1, \dots, x_n) = 0 \end{array} \right. \tag{1.1}$$

be a quadratic polynomial system where p_i are polynomials in n variables of degree ≤ 2. First of all, eliminating $x_2 \dots x_n$ by the multi-resultant method [2], we obtain an equation

$$(\lambda^2 M_2 + \lambda M_1 + M_0)v = 0 \tag{1.2}$$

where $M_i \in \mathbb{C}^{m \times m}, v \in \mathbb{C}^m \backslash \{0\}$, and all M_i are usually very sparse. Using the following standard technique, we can convert (1.2) into a generalized eigenvalue problem:

$$\underbrace{\begin{bmatrix} 0 & I_m \\ -M_0 & -M_1 \end{bmatrix}}_{C_2} \underbrace{\begin{bmatrix} v \\ \lambda v \end{bmatrix}}_{\hat{v}} = \lambda \underbrace{\begin{bmatrix} I_m & \\ & M_2 \end{bmatrix}}_{C_1} \underbrace{\begin{bmatrix} v \\ \lambda v \end{bmatrix}}_{\hat{v}} \Leftrightarrow C_2 x = \lambda C_1 x \tag{1.3}$$

where C_1 is likely to be singular because M_2 is.

2. Reformulating Problems and Developing Algorithms. Before starting to develop algorithms, we have to realize that quite often only a small set of the $2m$ solutions of (1.3) in a certain region is of practical interest. In particular, we are not interested in numerically infinity eigenvalues. Therefore, we will try to eliminate them in order to save computation time and obtain more accuracy in the remaining solutions. In order to find numerical solutions for the generalized eigenvalue problem we use three different approaches.

In our first approach (Algorithm D), we use the MATLAB eigensolver 'eig(A,B)'. The second approach (Algorithm C) is a classical one but new for this type of problem. We use a spectral

*Department of Mathematics, Louisiana State University, Baton Rouge, LA 70803.

transformation; this is, for some σ that is not an eigenvalue to the generalized eigenvalue problem (1.3), we introduce the shift σ to convert (1.3) into a standard eigenvalue problem as follows:

$$\begin{aligned} C_2 x = \lambda C_1 x \quad &\Leftrightarrow (C_2 - \sigma C_1)x = (\lambda - \sigma)C_1 x \\ &\Leftrightarrow \underbrace{\frac{1}{\lambda - \sigma}}_{\mu} x = \underbrace{(C_2 - \sigma C_1)^{-1} C_1}_{C} x \\ &\Leftrightarrow \mu x = Cx \end{aligned} \tag{2.1}$$

In the third approach (Algorithm S), we will try to eliminate some of the zero eigenvalues of C since they correspond to infinity solutions of (1.3). By eliminating these, we hope to save computation time and obtain more accuracy in the remaining solutions. The main idea for our third approach is to project the matrix pencil (C_2, C_1); that is, to project (C_2, C_1) onto a subspace. For $P, Q \in \mathbb{C}^{2m \times 2m}$ nonsingular, the matrix pencil (PC_2Q, PC_1Q) has the same eigenvalues as the matrix pencil (C_2, C_1). Therefore we can multiply (C_2, C_1) by a permutation matrix Q that rearranges B in a way that all its nonzero columns are in front of the zero columns. From here we again introduce a shift σ and obtain an equivalent problem

$$\underbrace{\frac{1}{\lambda - \sigma}}_{\mu} x = \underbrace{(C_2 Q - \sigma C_1 Q)^{-1} C_1 Q}_{S} \Leftrightarrow \mu x = Sx. \tag{2.2}$$

Then S is of the form

$$S = \begin{bmatrix} S_1 & 0 \\ S_2 & 0 \end{bmatrix} \tag{2.3}$$

where S_1 is a square matrix of order m_1, $m_1 = m+$the number of nonzero columns of M_2. Since M_2 is sparse, the size of S_1 is significantly smaller than the size of S. In fact, it is usually a little bigger than half of the size of S.

Now we can project S on S_1 by sacrificing some zero eigenvalues which we do not need anyway.

$$S_1 x_1 = \mu x_1 \tag{2.4}$$

where x_1 contains the first m_1 components of the vector x in (2.2). Finally, we apply the MATLAB matrix eigensolver 'eig' to find solutions for (2.4).

3. Experiment. In the experiment, we consider two test polynomial systems, the Morgan polynomials [3] and the Goldman polynomials [1]. The Morgan problem is known to be very sensitive; that is, to be ill-conditioned. Using multi-resultant method, both problems are converted into generalized eigenvalue problems

$$C_2 v = \lambda C_1 v, \tag{3.1}$$

where C_1 and C_2 are sparse matrices. Figures 4.1-4.2 show the sparsity patterns of (C_2, C_1) for both problems.

For the shift $\sigma = -15$, the computed eigenvalues of the eigenproblem corresponding to the Morgan polynomials are shown in Figure 4.3 where

- **Alg. D** computes solutions for the generalized eigenproblem (1.3)
- **Alg. C** computes solutions for the shifted standard eigenproblem (2.1)
- **Alg. S** computes solutions for the improved shifted standard eigenproblem (2.2)

The actual solutions of the Morgan polynomial are known to be real numbers but they are not known exactly. The results indicate that **Alg. S** is the most accurate in this example, followed by **Alg. C** and **Alg. D**.

In order to analyze this result and investigate what role the shift plays using algorithms D and S, we use the pseudospectra of the matrices C and S_1 in equations (2.1) and (2.4). The ε-pseudospectrum of a matrix A is defined as (see [4])

$$\Lambda_\epsilon(A) \quad = \{z \in C : \exists \quad \delta A \in C^{m \times m} \quad with \quad \|\delta A\| \le \varepsilon \quad s.t. \quad z \in \Lambda(A + \delta A)\},$$

which is equivalent to

$$\Lambda_\epsilon(A) \quad = \{z \in C : \sigma_m(zI - A) \leq \varepsilon\}. \tag{3.2}$$

In Figures 4.4 - 4.6, we use the MATLAB function *psconth* (which was modified from the function *pscont* from the MATLAB Test Matrix Toolbox) to obtain a 3-dimensional plot of $\log_{10}\sigma_m$ (z-axis) against the complex plane containing the spectra of the matrix. The x,y-plane contains the z-level sets of the 3-dimensional plot as contours. The level set corresponding to the z-value k encloses the ε-pseudospectrum $\Lambda_\epsilon(A)$ with $\varepsilon = 10^{-k}$. The bigger the level set is for a given value of ε, the more sensitive is the eigenvalue to changes in the entries in matrices C or S.

Figure 4.4 (left) shows the results for Goldman polynomial using **Alg. C** for $\sigma = -15$. The exact eigenvalues for the corresponding eigenproblem are -2, 0, 1. Comparing Figure 4.4 (left) to Figures 4.4 (right) - 4.6, we can see that Morgan polynomial is much more sensitive than Goldman polynomial since Morgan polynomial needs more accurate digits in σ_m (z-axis) in order to produce the same perturbations in the eigenvalues than Goldman polynomial does. Therefore, we concentrate on Morgan polynomial. From Figures 4.4 (right) - 4.6 (left) we can see that the closer the chosen shift σ is to a particular eigenvalue, the better-conditioned the problem is concerning that eigenvalue. For instance, the closer we get to the exact eigenvalue $-7.18...$ in Figures 4.5 - 4.6 (left), the smaller are the level sets we get for a fixed number of accurate digits in σ_m. Finally, when using **Alg. S** (Figure 4.6 (right)), we can even improve the results obtained using **Alg. C**.

4. Conclusion. According to the results we obtained, especially the results from our third approach (Alg. S), we managed to improve the conditioning of the problem. This may not seem too much in this comparatively small case, but small achievements like this can be crucial when dealing with systems of multivariate polynomial equations, which are in general highly sensitive to small deviations in the input data. Our attempt to improve some standard methods and algorithms can also be extended to achieve more significant success; for example, the matrix S_1 could be reduced further in order to get rid of all zero eigenvalues. Therefore, this paper is just meant to motivate more research in finding stable computational methods to solve multivariate polynomial systems.

REFERENCES

[1] E.-W. Chiconh, R. N. Goldman, and J. R. Miller, *Using multivariate resultants to find the intersection of three quadric surfaces*, ACM Trans. on Graphics, 10 (1991), pp. 378–400.

[2] D. Manocha, *Solving systems of polynomial equations*, IEEE Computer Graphics and Applications, (1994), pp. 46–55.

[3] A. P. Morgan, *Polynomial continuation and its relationship to the symbolic reduction of polynomial systems*, Symbolic and Numerical Computation for Artificial Intelligence, B. Donald et al. eds., (1992), pp. 23–45.

[4] L. N. Trefethen and D. Bau, *Numerical Linear Algebra*, SIAM, 1997.

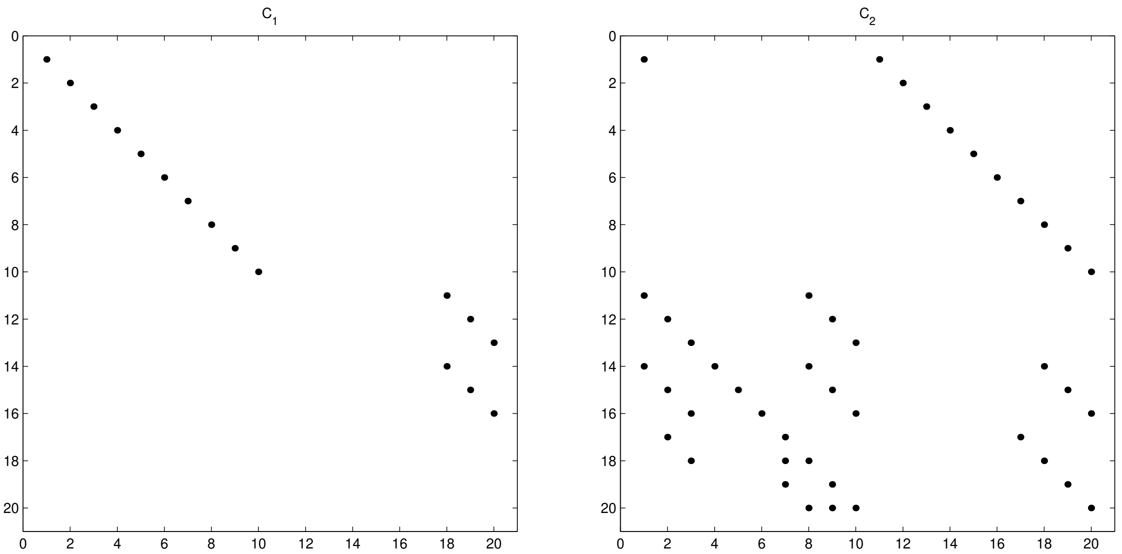

Fig. 4.1. *The matrix pencil* (C_2, C_2) *for Morgan polynomial*

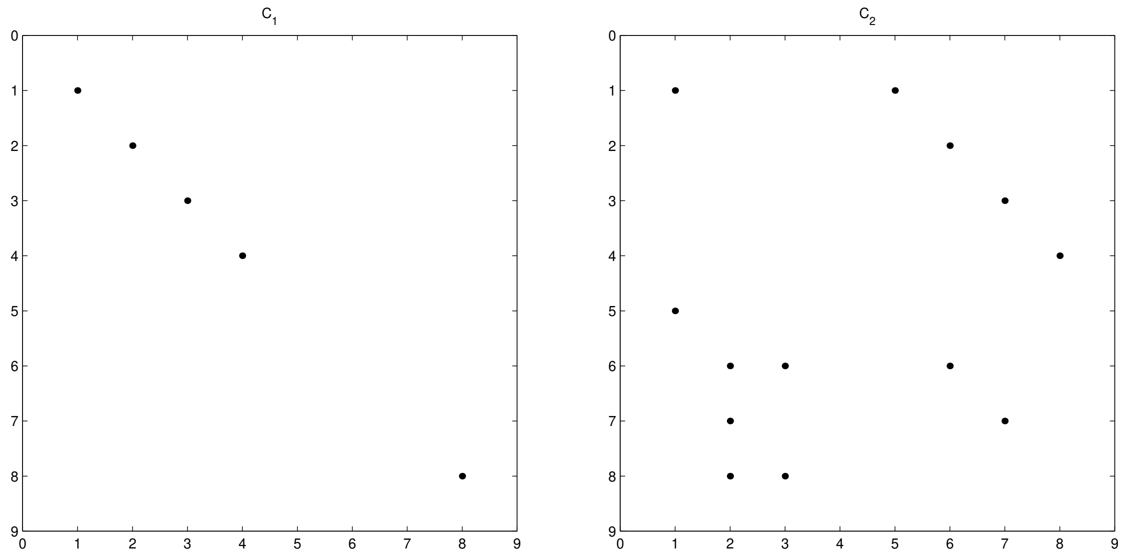

Fig. 4.2. *The matrix pencil* (C_2, C_2) *for Goldman polynomial*

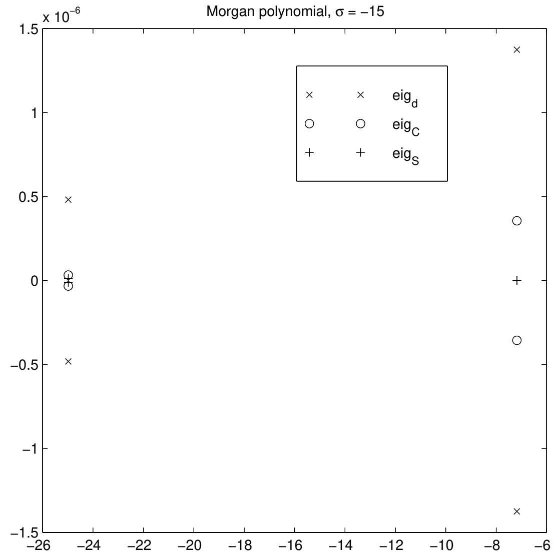

Fig. 4.3. *Computed eigenvalues for Mogan polynomial*

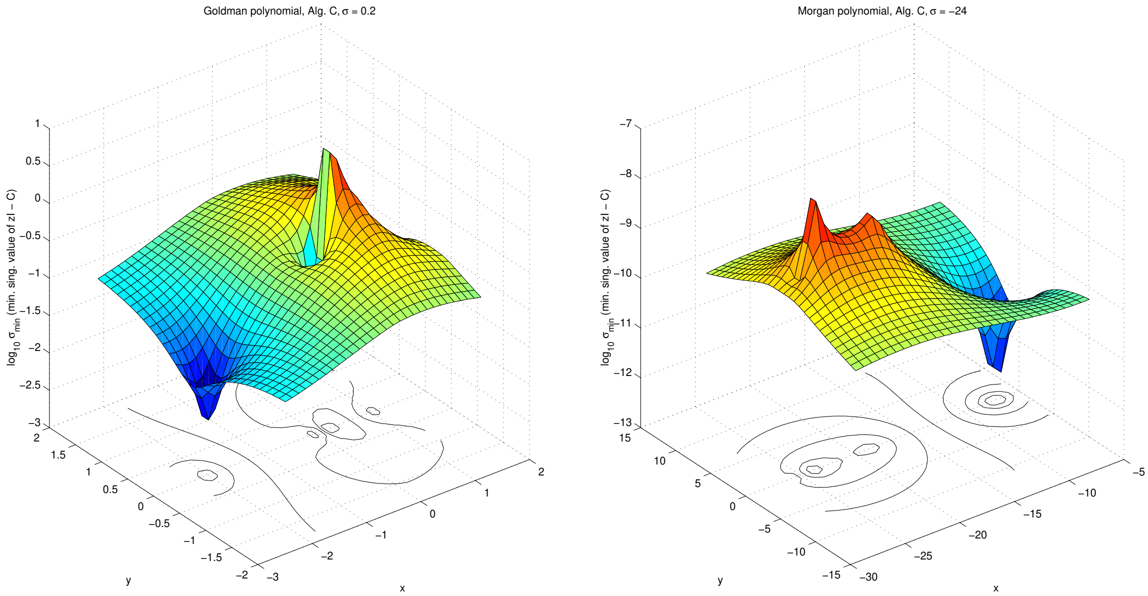

Fig. 4.4. $\Lambda_\epsilon(C)$ *for Goldman polynomial and Morgan polynomial*

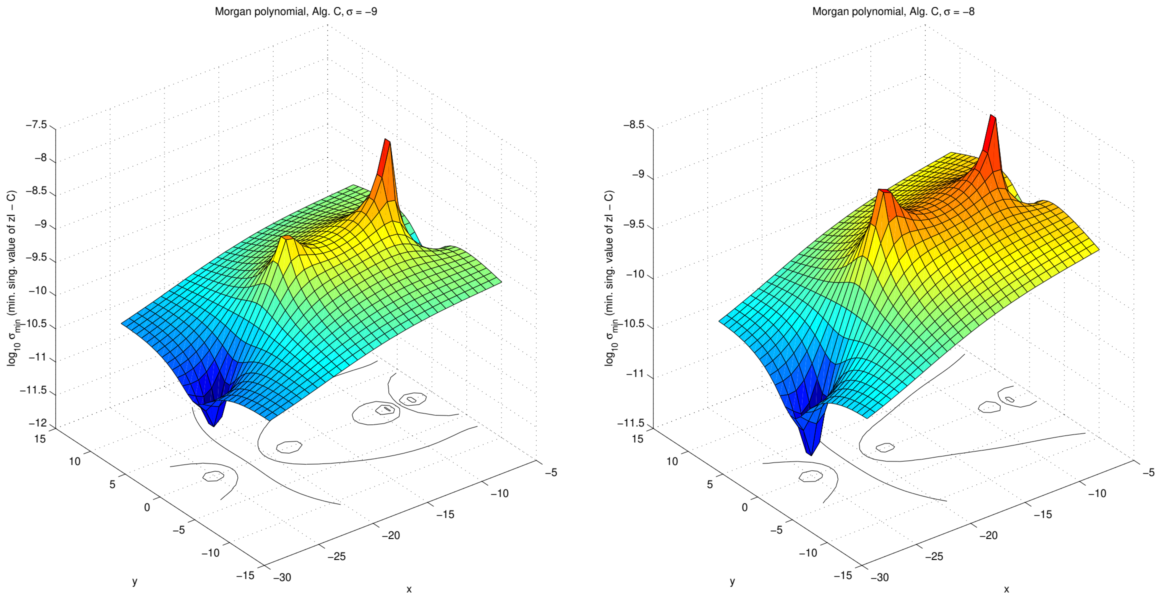

Fig. 4.5. $\Lambda_\epsilon(C)$ *with the shift* σ *moving closer to the eigenvalue -7.18... for Mogan polynomial*

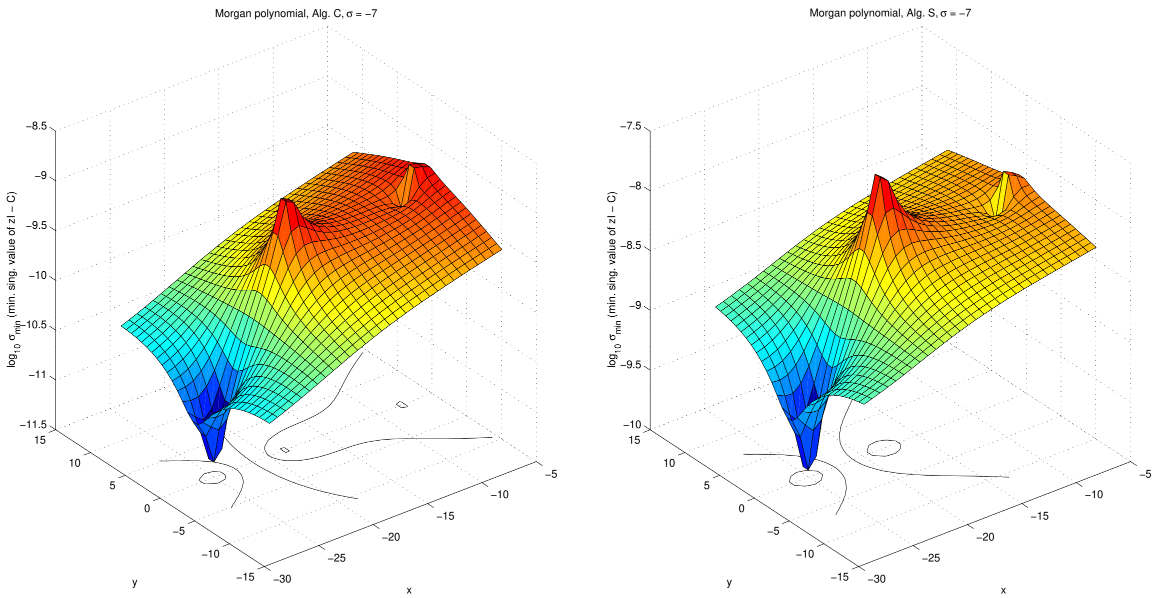

Fig. 4.6. *Left:* $\Lambda_\epsilon(C)$; *Right:* $\Lambda_\epsilon(S_1)$.