Hermite matrix in Lagrange basis for scaling static output feedback polynomial matrix inequalities

Akın Delibaşı[1,2,3], Didier Henrion[1,2,4]

June 1, 2010

Abstract

Using Hermite's formulation of polynomial stability conditions, static output feedback (SOF) controller design can be formulated as a polynomial matrix inequality (PMI), a (generally nonconvex) nonlinear semidefinite programming problem that can be solved (locally) with PENNON, an implementation of a penalty and augmented Lagrangian method. Typically, Hermite SOF PMI problems are badly scaled and experiments reveal that this has a negative impact on the overall performance of the solver. In this note we recall the algebraic interpretation of Hermite's quadratic form as a particular Bézoutian and we use results on polynomial interpolation to express the Hermite PMI in a Lagrange polynomial basis, as an alternative to the conventional power basis. Numerical experiments on benchmark problem instances show the substantial improvement brought by the approach, in terms of problem scaling, number of iterations and convergence behavior of PENNON.

Keywords: Static output feedback, Hermite stability criterion, Polynomial matrix inequality, Nonlinear semidefinite programming.

1 Introduction

In 1854 the French mathematician Charles Hermite studied quadratic forms for counting the number of roots of a polynomial in the upper half of the complex plane (or, by a simple rotation, in the left half-plane), more than two decades before Routh, who was apparently not aware of Hermite's work, see [8]. Hurwitz himself used some of Hermite's ideas to derive in 1895 his celebrated algebraic criterion for polynomial stability, now called the Routh-Hurwitz criterion and taught to engineering students in tabular form.

[1]CNRS; LAAS; 7 avenue du colonel Roche, F-31077 Toulouse, France
[2]Université de Toulouse; UPS, INSA, INP, ISAE; LAAS; F-31077 Toulouse, France
[3]Department of Electrical Engineering, Yıldız Technical University, Beşiktaş, Istanbul, Turkey.
[4]Faculty of Electrical Engineering, Czech Technical University in Prague, Czech Republic.

Hermite's criterion can be interpreted as a symmetric formulation of the Routh-Hurwitz criterion. This symmetry can be exploited in a semidefinite programming framework, as shown in [3] and [4] in the context of simultaneous stabilization of linear systems. Along the same vein, in [5] the problem of static output feedback (SOF) design was formulated as a polynomial matrix inequality (PMI) problem. In some cases (e.g. only one input or output available for feedback) this PMI problem simplifies to a bilinear matrix inequality (BMI) that can be solved numerically with PENBMI, a particular instance of PENNON, a general penalty and augmented Lagrangian method for nonlinear and semidefinite programming. Only convergence to a local optimum is guaranteed, but experiments reported in [5] show that quite often the approach is viable numerically. In particular, the SOF PMI formulation involves only controller parameters, and does not introduce (a typically large number of) Lyapunov variables.

Our motivation in this paper is to contribute along the lines initiated in [5] and to study the impact of SOF BMI problem formulation on the behavior of PENBMI, in particular w.r.t. data scaling and number of iterations. The Hermite matrix depends quadratically on coefficients of the characteristic polynomial, in turn depending polynomially on the controller parameters. As a result, coefficients of a given Hermite matrix typically differ by several orders of magnitude, and experiments reveal that this poor data scaling significantly impacts on the performance of PENBMI.

In this paper we use an alternative formulation of the Hermite matrix, using a Lagrange polynomial basis instead of the standard power basis. We build on previous work from the computer algebra and real algebraic geometry communities, recalling the interpretation of Hermite's quadratic form as a particular Bézoutian, the resultant of two polynomials, see [6] and references therein. This interpretation provides a natural choice for the nodes of the Lagrange basis. The construction of the Hermite matrix in this basis is carried out efficiently by interpolation, overcoming difficulties inherent to Vandermonde matrices, as suggested in [13] for general Bézout matrices.

In addition to digesting and tailoring to our needs results from computational algebraic geometry, we extend slightly the characterization of [13] to Hermitian forms with complex and repeated interpolation nodes. A key contribution of our work is to notice that in our SOF design application framework these nodes are roots of either imaginary or real part of a target characteristic polynomial featuring spectral properties desirable for the closed-loop system. This target polynomial is the main tuning parameter of our approach, and we provide numerical evidence that a suitably choice of target polynomial, compatible with achievable closed-loop dynamics, results in a significant improvement of SOF BMI problem scaling, with positive effects on the overall behavior (convergence, number of outer and inner iterations, line-search steps) of PENBMI. Furthermore, some of the problems that were not solvable in the power basis, see [5], can now be solved in the Lagrange basis. These improvements are illustrated on numerical examples extracted from the publicly available benchmark collection COMPl$_e$ib, see [9].

2 PMI formulation of SOF design problem

We briefly recall the polynomial matrix inequality (PMI) formulation of static output feedback (SOF) design problem proposed in [5].

Consider the linear system

$$\begin{array}{rcl} \dot{x} & = & Ax + Bu \\ y & = & Cx \end{array}$$

of order n with m inputs and p outputs, that we want to stabilize by static output feedback (SOF)

$$u = Ky.$$

In other words, given matrices $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{n\times m}$, $C \in \mathbb{R}^{p\times n}$, we want to find matrix $K \in \mathbb{R}^{m\times p}$ such that the eigenvalues of closed-loop matrix $A + BKC$ all belong to the left half of the complex plane.

Let $k \in \mathbb{R}^{mp}$ be the vector obtained by stacking the columns of matrix K. Define

$$q(s,k) = \det\,(sI - A - BKC) = \sum_{i=0}^{n} q_i(k)s^i \tag{1}$$

as the characteristic polynomial of matrix $A + BKC$. Coefficients of increasing powers of indeterminate s in polynomial $q(s,k)$ are multivariate polynomials in k, i.e.

$$q_i(k) = \sum_{\alpha} q_{i\alpha} k^{\alpha} \tag{2}$$

where $\alpha \in \mathbb{N}^{mp}$ describes all monomial powers.

The Routh-Hurwitz criterion for stability of polynomials has a symmetric version called the Hermite criterion. A polynomial is stable if and only if its Hermite matrix, quadratic in the polynomial coefficients, is positive definite. Algebraically, the Hermite matrix can be defined via the Bézoutian, a symmetric form of the resultant.

Let $a(u)$, $b(u)$ be two polynomials of degree n of the indeterminate u. Define the bivariate quadratic form

$$\frac{a(u)b(v) - a(v)b(u)}{u - v} = \sum_{i=1}^{n}\sum_{j=1}^{n} R_{i,j} u^{i-1} v^{j-1}. \tag{3}$$

The n-by-n matrix with entries $R_{i,j}$ is the Bézoutian matrix, whose determinant is the resultant of a and b, obtained by eliminating variable u from the system of equations $a(u) = b(u) = 0$.

The Hermite matrix in power basis of $q(s,k)$, denoted by $H^P(k)$, is defined as the Bézoutian matrix of the real and imaginary parts of $q(ju,k)$:

$$\begin{array}{rcl} a(u,k) & = & \operatorname{Im} q(ju,k) \\ b(u,k) & = & \operatorname{Re} q(ju,k). \end{array}$$

The roots of polynomial $q(s,k)$ belongs to the left half-plane if and only if

$$H^P(k) = \sum_{i=0}^{n}\sum_{j=0}^{n} q_i(k)q_j(k)H^P_{i,j} \succ 0.$$

The above relation is a matrix inequality depending polynomially on parameters k. Therefore, finding k amounts to solving a polynomial matrix inequality (PMI) problem.

Example 2.1 As an illustrative example, consider problem NN6 in [9]. The closed-loop characteristic polynomial is (to 8 significant digits):

$$\begin{aligned} q(s,k) \;=\; & s^9 + 23.300000s^8 + (4007.6500 - 14.688300k_2 + 14.685000k_4)s^7 \\ & + (91133.935 - 14.685000k_1 + 14.688300k_3 + 15.132810k_4)s^6 \\ & + (1149834.9 - 57334.489k_2 + 15.132810k_3 + 36171.693k_4)s^5 \\ & + (20216420 - 57334.489k_1 + 36171.693k_3 + 35714.763k_4)s^4 \\ & + (49276365 - 12660338k_2 + 35714.763k_3 + 3174671.8k_4)s^3 \\ & + (-1562.6281 \cdot 10^5 - 12660338k_1 - 3174671.8k_3 + 3133948.9k_4)s^2 \\ & + (-4315.5562 \cdot 10^5 + 95113415k_2 + 3133948.9k_3)s \\ & + 95113415k_1 \end{aligned}$$

with SOF gain $K = [k_1\, k_2\, k_3\, k_4]$. The 9-by-9 Hermite matrix of this polynomial cannot be displayed entirely for space reasons. Therefore we choose two representative entries :

$$\begin{aligned} H^P_{2,2}(k) \;=\; & 67436093 \cdot 10^9 - 77679687 \cdot 10^7 k_1 - 14862689 \cdot 10^9 k_2 \\ & - 18597671 \cdot 10^8 k_3 - 13524733 \cdot 10^8 k_4 - 43073807 \cdot 10^6 k_1 k_3 \\ & - 30195388 \cdot 10^7 k_1 k_4 + 30195388 \cdot 10^7 k_2 k_3 + 29808058 \cdot 10^7 k_2 k_4 \\ & + 99492593 \cdot 10^5 k_3^2 + 98216357 \cdot 10^5 k_3 k_4 \end{aligned}$$

and

$$H^P_{9,9}(k) = 23.300000.$$

Since we aim at using a local optimization solver such as PENBMI, convergence and behavior of the algorithm strongly depends on the choice of the initial point. If we choose, say $k = 0$ or any random small gain as an initial point, we notice that the above entries differ by several orders of magnitude. Experiments reveal that such a bad scaling have an overall negative impact on the behavior of the solver.

3 Scaling and conditioning

In the numerical analysis literature, there are two well-known scaling strategies to improve performance of algorithms. The first one is balancing, and it is used to improve conditioning of the eigenvalues of matrices with non-orthogonal eigenvectors, see e.g. [10] and references therein. Balancing has an overall positive effect on the matrix scaling, and it is used as a pre-processing step even when the ultimate objective is not computing the eigenvalues. For example, balancing is used systematically on the state-space representations of linear time-invariant systems in the Control System Toolbox for Matlab. It is also used when computing the roots of a polynomial with the Matlab function `roots`, which actually computes the eigenvalues of a suitably balanced companion matrix. In our application context, this scaling strategy is irrelevant however, since Hermite matrices are symmetric hence normal, so that their eigenvalues are perfectly conditioned.

The second standard scaling strategy is used to improve conditioning with respect to matrix inversion. Once a matrix norm is chosen, the condition number or conditioning is defined as the product of the norm of a matrix with the norm of the inverse of the matrix. Diagonal scaling can be used to minimize the conditioning of a matrix, see [7, Section 7.3]. Even though our objective is to not to invert Hermite matrices or to solve linear systems of equations involving Hermite matrices, we use the condition number

$$\kappa(H) = \|H\|_F \|H^{-1}\|_F$$

with respect to the Frobenius norm

$$\|H\|_F = \sqrt{\sum_{i,j} H_{i,j}^2}$$

as a representative of scaling of the Hermite matrix (any other matrix norm could be appropriate). In the PENBMI algorithm, the Hermite matrix enters explicitly in the penalty function and also when computing dual multipliers in the augmented Lagrangian, so it is expected that improving conditioning of the Hermite matrix results in better convergence and behavior of the overall algorithm.

Example 3.1 In Table 1 we report average condition numbers of the Hermite matrix $H^P(k)$ of Example 2.1 for 100 randomly chosen control gains $k \in \mathbb{R}^4$ with entries uniformly distributed in the box $[-r, r]$, for increasing values of the box radius r. We see that the condition number is very large for reasonable random control gains around $k = 0$.

Table 1: Example `NN6`: average conditioning $\kappa(H^P(k))$ of the Hermite matrix in power basis for 100 random points k uniformly distributed in a box of given radius r.

r	$\kappa(H^P(k))$
1	$5.0944 \cdot 10^{17}$
3	$7.3416 \cdot 10^{17}$
5	$1.4994 \cdot 10^{18}$
15	$3.7773 \cdot 10^{20}$

Other experiments reveal that Hermite matrices expressed in the power basis are typically poorly scaled, in the sense that their conditioning is large. When dealing with a nonlinear solver such as PENBMI, this has an overall negative effect on the performance. We are therefore interested in improving conditioning of the Hermite matrix.

4 Scaled power basis

A possible remedy to address the poor conditioning properties of the Hermite matrix is to scale the frequency variable s, that is, to substitute ρs for s in the characteristic polynomial, for a suitable positive scaling ρ. Finding the optimal value of ρ (in terms of conditioning) may be formulated as an optimization problem, but numerical experiments indicate that good results are achieved when ρ is such that the constant and highest power polynomial coefficients are both equal to one. For example, this idea was implemented by Huibert Kwakernaak in the `scale` function of the Polynomial Toolbox for Matlab, see [12].

Example 4.1 Consider the simple example `AC4` in [9]. The open-loop ($k = 0$) characteristic polynomial is

$$\begin{aligned} q(s,0) &= \det(sI - A) \\ &= s^4 + 150.92600s^3 + 130.03210s^2 - 1330.6306s - 66.837750 \end{aligned}$$

with Hermite matrix in power basis

$$H^P = \begin{bmatrix} 88936.354 & 0 & 10087.554 & 0 \\ 0 & -162937.14 & 0 & 1330.631 \\ 10087.554 & 0 & 20955.855 & 0 \\ 0 & 1330.6306 & 0 & 150.92600 \end{bmatrix}.$$

The condition number is equal to $\kappa(H^P) = 1158.2$. If we choose $\rho = |\det A|^{-\frac{1}{4}} \approx 3.4974 \cdot 10^{-1}$, the scaled characteristic polynomial has unit constant and highest coefficient, and with $S = \operatorname{diag}(\rho^3, \rho^2, \rho^1, 1)$ the resulting scaled Hermite matrix reads

$$SH^PS = \begin{bmatrix} 163.48864 & 0 & 151.37636 & 0 \\ 0 & -2445.0754 & 0 & 163.00225 \\ 151.37636 & 0 & 2567.0923 & 0 \\ 0 & 163.00225 & 0 & 150.92600 \end{bmatrix}$$

with smaller condition number $\kappa(SH^PS) = 32.906$.

Whereas this simple scaling strategy with one degree of freedom may prove useful for small-degree polynomials and small-size Hermite matrices, a more sophisticated approach is required for larger instances.

5 Lagrange basis

In this section we show how the Hermite matrix can be scaled by an appropriate choice of polynomial basis. Moreover, this basis allows for a straightforward entrywise construction of the Hermite matrix bypassing the Bézoutian construction (3).

5.1 Distinct interpolation points

Consider n distinct interpolation points $u_i \in \mathbb{C}$, $i = 1, \ldots, n$, and define the j-th Lagrange polynomial

$$l_j(u) = \prod_{i=1, i \neq j}^{n} \frac{u - u_i}{u_j - u_i}$$

which is such that $l_j(u_j) = 1$ and $l_j(u_i) = 0$ if $i \neq j$. In matrix form we can write

$$\begin{bmatrix} 1 \\ u \\ u^2 \\ \vdots \\ u^{n-1} \end{bmatrix} = \begin{bmatrix} 1 & 1 & \cdots & 1 \\ u_1 & u_2 & \cdots & u_n \\ u_1^2 & u_2^2 & \cdots & u_n^2 \\ \vdots & & & \vdots \\ u_1^{n-1} & u_2^{n-1} & \cdots & u_n^{n-1} \end{bmatrix} \begin{bmatrix} l_1(u) \\ l_2(u) \\ l_3(u) \\ \vdots \\ l_n(u) \end{bmatrix} = V_u l(u) \tag{4}$$

where V_u is a Vandermonde matrix. Given a univariate polynomial $q(s)$ with real coefficients, define

$$\begin{aligned} a(u) &= \operatorname{Im} q(jw) \\ b(u) &= \operatorname{Re} q(jw) \end{aligned} \tag{5}$$

as its imaginary and real parts on the imaginary axis, respectively. In the following, the star denotes transpose conjugation and the prime denotes differentiation, i.e.

$$a'(u) = \frac{da(u)}{du}.$$

Theorem 5.1 *When the interpolation points u_i are distinct, the Hermite matrix of $q(s)$ in Lagrange basis, denoted by H^L, is given entrywise by*

$$H^L_{i,j} = \begin{cases} \dfrac{a(u_i^*)b(u_j) - a(u_j)b(u_i^*)}{u_i^* - u_j} & \text{if } u_i^* \neq u_j, \\ a'(u_i^*)b(u_j) - a(u_j)b'(u_i^*) & \text{otherwise,} \end{cases}$$

for all $i, j = 1, \ldots, n$.

Proof Let us express the Bézoutian of a and b as a bivariate quadratic form

$$\frac{a(u)b(v) - a(v)b(u)}{u - v} = \begin{bmatrix} 1 \\ v \\ \vdots \\ v^{n-1} \end{bmatrix}^* H^P \begin{bmatrix} 1 \\ u \\ \vdots \\ u^{n-1} \end{bmatrix}$$

where H^P is the Hermite matrix of q in the power basis. Recalling relation (4), the Bézoutian becomes

$$\frac{a(u)b(v) - a(v)b(u)}{u - v} = l(v)^* V_v^* H^P V_u l(u) = l(v)^* H^L l(u)$$

so that the Hermite matrix of q in the Lagrange basis can be expressed as

$$H^L = V_v^* H^P V_u.$$

By evaluation at n distinct interpolation points u_i and v_j, H^L is given entrywise by

$$H^L_{i,j} = \frac{a(u_i^*)b(v_j) - a(v_j)b(u_i^*)}{u_i^* - v_j}. \tag{6}$$

Now let $u_i^* \to v_j$ for all $i, j = 1, \dots, n$. After adding and subtracting $a(v_j)b(v_j)$ to the numerator of (6), and using a limiting argument we find

$$H^L_{i,j} = a'(u_i^*)b(u_j) - a(u_j)b'(u_i^*).$$

□

In Appendix B we extend this result to the case when the interpolation points are not distinct.

5.2 Diagonal scaling

Once the Hermite matrix is given in the Lagrange basis, a diagonal scaling can be readily used to improve its conditioning significantly.

Corollary 5.2 *Let the interpolation points be (distinct) roots of either $a(u)$ or $b(u)$, as defined in (5). Then the Hermite matrix of $q(s)$ in Lagrange basis is block diagonal, with 2×2 blocks corresponding to pairs of complex conjugate points and 1×1 blocks corresponding to real points.*

Proof From Theorem 5.1, all the off-diagonal entries of H^L are given by

$$\frac{a(u_i^*)b(u_j) - a(u_j)b(u_i^*)}{u_i^* - u_j} \tag{7}$$

when interpolation points u_i and u_j are not complex conjugate. Both terms $a(u_i^*)b(u_j)$ and $a(u_j)b(u_i^*)$ are equal to zero in (7) since the interpolation points are the roots of either $a(u)$ or $b(u)$. The diagonal entries are $a'(u_i^*)b(u_j) - a(u_j)b'(u_i^*)$ since it is assumed that interpolation points are distinct. Therefore this part of H^L is 1×1 block-diagonal.

When interpolation points u_i and u_j are complex conjugate, there is only one non-zero entry (i, j) which is equal to $a'(u_i^*)b(u_j) - a(u_j)b'(u_i^*)$ and located in the off-diagonal entry, according to pairness. The diagonal entries of this case are equal to zero by virtue of equation (7). Therefore this part of H^L is 2×2 block-diagonal.□

From Corollary 5.2 it follows that we can easily find a block-diagonal scaling matrix S such that the scaled Lagrange Hermite matrix

$$H^S = SH^LS$$

has smaller condition number. Nonzero entries of S are given by

$$S_{i,j} = |H^L_{i,j}|^{-\frac{1}{2}} \tag{8}$$

whenever $H^L_{i,j}$ is a nonzero entry (i, j) of H^L. In the case H^L is positive semidefinite, it is shown in [7, Corollary 7.6] that this scaling policy is almost optimal with respect to minimization of the Euclidean condition number. Note however that here H^L is not necessarily positive semidefinite and we use the Frobenius condition number.

Example 5.3 As an illustrative example, consider problem `NN5` in [9]. The open-loop ($k = 0$) characteristic polynomial is

$$\begin{aligned} q(s) &= s^7 + 10.171000s^6 + 96.515330s^5 + 458.42510s^4 \\ &\quad +2249.4849s^3 + 1.2196400s^2 - 448.72180s + 6.3000000. \end{aligned}$$

The Hermite matrix in power basis has the following entries:

$$\begin{array}{lcllcl} H^P_{1,1} &=& -2826.9473 & H^P_{1,3} &=& -14171.755 \\ H^P_{1,5} &=& 608.04658 & H^P_{1,7} &=& -6.3000000 \\ H^P_{2,2} &=& -14719.034 & H^P_{2,4} &=& 206313.38 \\ H^P_{2,6} &=& -4570.2494 & H^P_{3,3} &=& 209056.94 \\ H^P_{3,5} &=& -4687.9634 & H^P_{3,7} &=& 1.2196400 \\ H^P_{4,4} &=& 1026532.4 & H^P_{4,6} &=& -22878.291 \\ H^P_{5,5} &=& 21366.759 & H^P_{5,7} &=& -458.42510 \\ H^P_{6,6} &=& 523.23232 & H^P_{7,7} &=& 10.171000, \end{array}$$

remaining nonzero entries being deduced by symmetry. Apparently, this matrix is not well-scaled, with conditioning equal to $\kappa(H^P) = 3.7046 \cdot 10^6$. Choosing interpolation points u_i as roots of $a(u)$, the imaginary part of $q(s)$ along the imaginary axis, we use Theorem 5.1 to build the Hermite matrix in Lagrange basis:

$$\begin{aligned} H^L &= \operatorname{diag}\,(-2826.9473,\ 41032866 \cdot 10^3,\ 44286011 \cdot 10^2,\ 41032866 \cdot 10^3, \\ &\qquad 44286011 \cdot 10^2, \begin{bmatrix} 0 & 22222.878 \\ 22222.878 & 0 \end{bmatrix}). \end{aligned}$$

This matrix is still badly scaled with $\kappa(H^L) = 2.0983 \cdot 10^7$, but it is almost diagonal. Using the elementary diagonal scaling matrix S given by (8), we obtain

$$H^S = SH^LS = \operatorname{diag}\,(-1,\ 1,\ 1,\ 1,\ 1, \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix})$$

which is a well-scaled representation of the Hermite matrix, with perfect conditioning $\kappa(H^S) = 1$.

In the above example, we study scaling of the Hermite matrix of the open-loop characteristic polynomial, that is, when $k = 0$. The conditioning is improved for this value of k but also in a neighborhood. We can improve conditioning of the Hermite matrix in the neighborhood of any other given value of k which is input to the PENBMI solver. In particular, we would like the conditioning to be small around points which are valid stabilizing feedback gains. Obviously, we do not know these values a priori. In the following section, we explain our strategy to scale the Hermite matrix without explicit knowledge of k.

5.3 Target polynomial

In our control application, our main tuning tool is the target polynomial, denoted by $q(s)$. The target polynomial provides the interpolation points required to build well-scaled Lagrange basis Hermite matrix in the SOF problem. These points are defined as in Corollary 5.2 as the roots of either the real or imaginary part of $q(s)$ when evaluated along the imaginary axis. In the previous section we explained that the conditioning of the Hermite matrix is significantly improved by expressing the Hermite matrix in the Lagrange basis corresponding to these points.

In the context of SOF design, the target polynomial may be either choosen as

- a valid closed-loop characteristic polynomial (1) for a specific value of k, or as
- a polynomial with desired pole distribution for the closed-loop system.

Furthermore, we invoke a continuity argument to claim that the conditioning of the Hermite matrix does not change abruptly in a neighborhood of a given target polynomial.

Example 5.4 Consider again Example 2.1 and let the target polynomial be an achievable closed-loop characteristic polynomial $q(s,k) = \det(sI - A - BKC)$, where

$$k = [-4.3264 \cdot 10^{-1},\ \ -1.6656,\ \ 1.2537 \cdot 10^{-1},\ \ 2.8772 \cdot 10^{-1}]$$

is a random feedback gain. The roots of the imaginary part of $q(s,k)$ are chosen as interpolation points

$$u = (0,\ \pm 60.847,\ \pm 16.007,\ \pm 9.2218, \pm 2.7034i)\,.$$

Here are two representative entries of the resulting Lagrange basis Hermite matrix:

$$\begin{aligned} H^S_{3,3}(k) \;=\;& 9.4439251 \cdot 10^{-1} + 1.9763715 \cdot 10^{-4} k_1 - 8.9049916 \cdot 10^{-4} k_2 \\ & - 8.6909277 \cdot 10^{-3} k_3 + 1.9212126 \cdot 10^{-1} k_4 \\ & + 3.8300306 \cdot 10^{-9} k_1 k_2 - 1.0276186 \cdot 10^{-8} k_1 k_3 \\ & + 3.3905595 \cdot 10^{-5} k_1 k_4 - 3.4222179 \cdot 10^{-5} k_2 k_3 \\ & - 3.8046300 \cdot 10^{-5} k_2 k_4 + 2.7420115 \cdot 10^{-9} k_3^2 \\ & + 6.5442491 \cdot 10^{-6} k_3 k_4 + 1.015195648 \cdot 10^{-5} k_4^2 \end{aligned}$$

and

$$\begin{aligned} H^S_{2,4}(k) \;=\;& -3.1581768 \cdot 10^{-5} - 2.6157943 \cdot 10^{-6} k_1 + 1.8882813 \cdot 10^{-4} k_2 \\ & + 1.8800760 \cdot 10^{-5} k_3 + 1.2109347 \cdot 10^{-3} k_4 \\ & - 1.5348324 \cdot 10^{-7} k_1 k_2 - 1.0342207 \cdot 10^{-9} k_1 k_3 \\ & + 8.2034001 \cdot 10^{-6} k_1 k_4 - 7.2727791 \cdot 10^{-7} k_2 k_3 \\ & - 1.0073150 \cdot 10^{-6} k_2 k_4 - 4.4843248 \cdot 10^{-8} k_3^2 \\ & - 4.6869455 \cdot 10^{-5} k_3 k_4 - 4.3676630 \cdot 10^{-5} k_4^2. \end{aligned}$$

In Table 2 we report the conditioning of $H^S(k)$ around $k = 0$. Comparing with the results of the power basis Hermite matrix $H^P(k)$ given in Example 2.1, we observe a significant improvement.

Table 2: Example NN6 revisited: average conditioning $\kappa(H^S(k))$ of the Hermite matrix in Lagrange basis for 100 random points k uniformly distributed in a box of given radius r.

r	$\kappa(H^S(k))$
1	$4.7564 \cdot 10^4$
3	$3.5934 \cdot 10^5$
5	$1.9632 \cdot 10^5$
15	$8.0353 \cdot 10^6$

6 Numerical examples

In this section, we present the benefits of Lagrange basis against power basis and scaled power basis when solving SOF PMI problems found in the database COMPl$_e$ib, see [9]. Even though Michal Kočvara and Michael Stingl informed us that an AMPL interface to PENNON is now available to solve PMI problems, in this paper we consider only BMIs (i.e. quadratic PMIs) and the PENBMI solver (a particular instance of PENNON focusing on BMIs) under the YALMIP modeling interface, see [11]. The numerical examples are processed with YALMIP R20070523 and PENBMI 2.1 under Matlab R2007a running on a Pentium D 3.4GHz system with 1GB RAM. We set the PENBMI penalty parameter `P0` by default to 0.001 (note that this is not the default YALMIP setting).

As in [5], the optimization problem to be solved is

$$\begin{array}{ll} \min_{k,\lambda} & \mu\|k\| - \lambda \\ \text{s.t.} & H(k) \succeq \lambda I \end{array}$$

where $H(k)$ is the Hermite matrix in power, scaled power or Lagrange basis, $\mu > 0$ is a parameter and $\|.\|$ is the Euclidean norm. Parameter μ allows to trade off between feasibility of the BMI and a moderate norm of the feedback gain, which is generally desirable in practice, to avoid large feedback signals. This adjustment is necessary in many examples. Indeed, the smallest values of $\|k\|$ are typically located at the boundary of the feasibility set, so the resulting closed-loop system is fragile and a small perturbation on system parameters may be destabilizing.

PENBMI is a local optimization solver. Therefore, the choice of initial guess k_0, λ_0 is critical. In most of the examples we choose the origin as the initial point. However this is not always an appropriate choice, as illustrated below. In addition to this, PENBMI does not directly handle complex numbers (unless the real and imaginary parts are split off, resulting in a real coefficient problem of double size), so we restrict the interpolation points to be real numbers.

As a result of the root interlacing property, the roots of real and imaginary parts of a stable polynomial are real (and interlacing). Hence if we choose a stable target polynomial $q(s)$ the resulting interpolation points are necessarily real.

Example 6.1 Consider again problem AC4, with characteristic polynomial

$$\begin{aligned} q(s,k) &= s^4 + 150.92600s^3 + (130.03210 - 18.135000k_1 - 19612.500k_2)s^2 \\ &\quad - (1330.6306 + 19613.407k_1 + 18322.789k_2)s - (66.837750 + 980.62500k_1 + 867.10818k_2) \end{aligned}$$

and power basis Hermite matrix with entries

$$\begin{aligned}
H^P_{1,1} &= 88936.354 + 2615765.6k_1 + 2378454.6k_2 \\
&\quad + 19233397k_1^2 + 34974730k_1k_2 + 15887840k_2^2 \\
H^P_{1,3} &= 10087.554 + 148001.81k_1 + 130869.17k_2 \\
H^P_{2,2} &= -162937.14 - 2378239.7k_1 + 23845311k_2 \\
&\quad + 355689.13k_1^2 + 38500022 \cdot 10^1 k_1k_2 + 35935569 \cdot 10^1 k_2^2 \\
H^P_{2,4} &= 1330.6306 + 19613.407k_1 + 18322.789k_2 \\
H^P_{3,3} &= 20955.855 + 16876.364k_1 - 2941713.4k_2 \\
H^P_{4,4} &= 150.92600.
\end{aligned}$$

Open-loop poles of the system are $(2.5792\,, -5.0000 \cdot 10^{-2}\,, -3.4552,\ -150.00)$. If we define our target polynomial roots as $(-5.0000 \cdot 10^{-2}\,, -5.0000 \cdot 10^{-2}\,, -3.4552\,, -150.00)$, keeping the stable open-loop poles and shifting the unstable open-loop pole to the left of the imaginary axis, our 4 interpolation points (roots of the real part of the target polynomial) are $u = (\pm 23.100\,, \pm 4.9276 \cdot 10^{-2})$ and the resulting Lagrange basis Hermite matrix has entries

$$\begin{aligned}
H^S_{1,1} &= 6.3432594 \cdot 10^{-1} + 3.1878941 \cdot 10^{-1}k_1 - 17.462079k_2 \\
&\quad + 4.4822907k_1^2 + 4.4060140k_1k_2 + 4.1121739k_2^2 \\
H^S_{1,2} &= -3.7795293 \cdot 10^{-1} - 1.0581354k_1 - 18.455273k_2 \\
&\quad - 3.6574685 \cdot 10^{-3}k_1^2 - 4.4045142k_1k_2 - 4.1114926k_2^2 \\
H^S_{1,3} &= 2.4459288 + 36.285639k_1 + 42.619220k_2 \\
&\quad + 7.8459729k_1^2 + 189.06139k_1k_2 + 169.77324k_2^2 \\
H^S_{1,4} &= 1.9481147 + 28.929191k_1 + 12.037605k_2 \\
&\quad + 7.5224565k_1^2 - 161.11487k_1k_2 - 157.07808k_2^2 \\
H^S_{2,2} &= 6.3432594 \cdot 10^{-1} + 3.1878941 \cdot 10^{-1}k_1 - 17.462079k_2 \\
&\quad + 4.4822907 \cdot 10^{-3}k_1^2 + 4.4060140k_1k_2 + 4.1121739k_2^2
\end{aligned}$$

and

$$\begin{aligned}
H^S_{2,3} &= 1.9481074 + 28.929083k_1 + 12.037568k_2 \\
&\quad + 7.5224134k_1^2 - 161.11415k_1k_2 - 157.07737k_2^2 \\
H^S_{2,4} &= 2.4459288 + 36.285639k_1 + 42.619220k_2 \\
&\quad + 7.8459729k_1^2 + 189.06139k_1k_2 + 169.77324k_2^2 \\
H^S_{3,3} &= 659.47243 + 19434.408k_1 + 18140.083k_2 \\
&\quad + 143181.92k_1^2 + 267314.59k_1k_2 + 124766.18k_2^2 \\
H^S_{3,4} &= 665.36241 + 19520.378k_1 + 17279.067k_2 \\
&\quad + 143169.06k_1^2 + 253396.76k_1k_2 + 111775.41k_2^2 \\
H^S_{4,4} &= 659.47243 + 19434.408k_1 + 18140.083k_2 \\
&\quad + 143181.92k_1^2 + 267314.59k_1k_2 + 124766.18k_2^2.
\end{aligned}$$

Choosing the power basis representation with the origin $k_0 = 0$ as initial point and trade-off parameter $\mu = 10^{-5}$, PENBMI stops by a line-search failure and YALMIP displays a warning. However, we obtain a feasible solution $\lambda = 150.88$ and $k = [1.4181,\ -1.6809]$. This computation requires 43 outer iterations, 433 inner iterations and 825 linesearch steps. On the other hand, in the Lagrange basis representation, the problem is solved with no error or warning, yielding $\lambda = 9.8287 \cdot 10^{-1}$, $k = [-5.0902 \cdot 10^{-2},\ -2.0985 \cdot 10^{-2}]$ with 17 outer iterations, 100 inner iterations and 159 line-search steps.

We notice however that using the same trade-off parameter μ for both representations is not fair since H^P and H^S have significantly different scalings. If we choose $\mu = 0.1$ for the power basis representation, no problem is detected during the process and we obtain $\lambda = 150.87$, $k = [8.0929 \cdot 10^{-2},\ -1.6953 \cdot 10^{-1}]$ after 26 outer iterations, 188 inner iterations and 238 line-search steps. In addition to this, if we use $\rho = 3.4974 \cdot 10^{-1}$ for the power basis scaling, the results are $\lambda = 150.79$, $k = [3.9160 \cdot 10^{-1},\ -5.7927 \cdot 10^{-1}]$ after 21 outer iterations, 153 inner iterations and 197 line-search steps. So it seems that the Lagrange basis representation becomes relevant mainly for high degree systems. This is confirmed by the experiments below.

Consider the `AC7`, `AC17`, `PAS`, `REA3`, `UWV`, `NN5`, `NN1` and `HE1` SOF BMI problems of COMPl$_e$ib. In Table 3 we report comparative results for the power basis, scaled power basis and Lagrange basis representations. As in Example 4.1, the main strategy to choose the target polynomials (and hence the interpolation points) is to mirror the open-loop stable roots, and to shift the open-loop roots to, say $-5.0000 \cdot 10^{-2}$ (any other small negative value may be suitable). We see that the behavior indicators of PENBMI are significantly better in the Lagrange basis, and the improvement is more dramatic for larger degree examples. More specifically:

- for small degree systems like `AC17` or `PAS` there is only a minor improvement. Furthermore, strict feasibility is not achieved for the scaled power basis and our first attempt of Lagrange basis. At the second attempt, we use a different strategy, shifting the unstable poles to $-1.0000 \cdot 10^{-3}$, to define the target polynomial;
- at the first attempt to solve the `REA3` example, strict feasibility is not achieved in the power basis, since λ is almost zero. Therefore it is necessary to tune the μ parameter.

Results of the second attempt show that the BMI problem is solved and the Lagrange and scaled power basis computations are slightly less expensive than the power basis computation;

- example `UWV` has two inputs and two outputs. However, because of cancellation of higher degree terms in the characteristic polynomial, the degree of the Hermite matrix is equal to 2 and we can use PENBMI on this problem;

- on open-loop stable systems such as `UWV` or `AC17`, the improvement brought by the Lagrange basis is less significant. Since the main purpose of our optimization problem is to minimize the norm of control gain, we observe that the Lagrangian basis is still slightly better than the other representations;

- PENBMI is unable to reach a feasible point for examples `NN5`, `NN1` and `HE1`, when we choose the origin as the initial point. Indeed, local optimization techniques seek an optimal point inside the feasible set in a neighborhood of the initial point. Therefore, achievement of the solver may be sensitive to the initial point. When the initial point is defined heuristically or randomly, the improvement is significant for system `NN5` in Lagrange basis. However, there is no improvement over the low-degree examples `NN1` and `HE1`, when we use this simple strategy to define the target polynomial.

Consider the `NN6` SOF BMI problem that was not previously solvable in the power basis, see [5]. Open-loop poles of the system are

$$\sigma_0 = \left(2.7303,\ 0,\ -7.2028\cdot 10^{-2} \pm 60.804i,\ -1.0785\cdot 10^{-1} \pm 15.677i,\ -2.6764,\ -3.3000,\ -19.694\right).$$

When we use the scaled power basis ($\rho = 1.0979\cdot 10^{-1}$), we find a solution for this problem. The results are $\lambda = 2.8592$, $k = [5.8934\cdot 10^{-1},\ 7.1564,\ -5.1821\cdot 10^{-1},\ 109.21]$ with 22 outer iterations, 108 inner iterations and 131 line-search steps, using the origin $k_0 = 0$ as initial point and trade-off parameter $\mu = 10^{-3}$.

We also find a solution with the Hermite matrix in Lagrange basis. The strategy to define the target polynomial is to change the unstable open-loop poles into slightly stable poles (shifting the real part to a small negative value). According to this strategy, our target polynomial has the following roots

$$\begin{aligned}\sigma_1 \ = \ & \left(-1.0000\cdot 10^{-3} \pm i,\ -7.2028\cdot 10^{-2} \pm 60.804i,\ -1.0785\cdot 10^{-1} \pm 15.677i,\right.\\ & \left.-2.6764,\ -3.3000,\ -19.694\right).\end{aligned}$$

The BMI SOF problem is solved with no error or warning in the Lagrange basis, yielding $\lambda = 8.8487\cdot 10^{-1}$, $k = [1.3682,\ 4.8816,\ 44.959,\ 59.016]$ with 17 outer iterations, 80 inner iterations and 138 linesearch steps, using the orgin as initial point and trade-off parameter $\mu = 10^{-5}$.

7 Conclusion

The Hermite matrix arising in the symmetric formulation of the polynomial stability criterion is typically ill-scaled when expressed in the standard power basis. As a consequence, a nonlinear semidefinite programming solver such as PENNON may experience convergence problems when applied on polynomial matrix inequalities (PMIs) coming from benchmark static output feedback (SOF) problems. In this paper we reformulated Hermite's SOF PMI in a Lagrange polynomial basis. We slightly extended the results of [13] to use polynomial interpolation on possibly complex and repeated nodes to construct the Hermite matrix, bypassing potential numerical issues connected with Vandermonde matrices. In our control application, a natural choice of Lagrange nodes are the roots of a target polynomial, the desired closed-loop characteristic polynomial.

The idea of using the Lagrange polynomial basis to address numerical problems which are typically ill-scaled when formulated in the power basis has already proven successful in other contexts. For example, in [2] it was shown that roots of extremely ill-scaled polynomials (such as a degree 200 Wilkinson polynomial) can be found at machine precision using eigenvalue computation of generalized companion matrices obtained by an iterative choice of Lagrange interpolation nodes. In [14] the fast Fourier transform (a particular interpolation technique) was used to perform spectral factorization of polynomials of degree up to one million. Another example of successful use of alternative bases and high-degree polynomial interpolation to address various problems of scientific computing is the `chebfun` Matlab package, see [1]. Even though our computational results on SOF PMI problems are less dramatic, we believe that the use of alternative bases and interpolation can be instrumental to addressing various other control problems formulated in a polynomial setting.

This paper focused on quadratic (BMI) problems corresponding to systems with one input or one output for feedback. The immediate next step is to validate our scaling strategy on multi-input multi-output systems, and hence on PMI problems of degree higher than two. To go further than mere closed-loop stabilization, we are also planning to incorporate a polynomial formulation of H_2 and H_∞ norm optimization problems.

Appendix A: Matlab implementation

A Matlab implementation of the method described in this paper is available at

`homepages.laas.fr/henrion/software/hermitesof.m`

Our implementation uses the Symbolic Math Toolbox and the YALMIP interface. It is not optimized for efficiency, and therefore it can be time-consuming already for medium-size examples.

Let us use function `hermitesof` with its default tunings:

```
>> [A,B1,B,C1,C] = COMPleib('NN1');
>> A,B,C
A =
     0     1     0
     0     0     1
     0    13     0
B =
     0
     0
     1
C =

     0     5    -1
    -1    -1     0
>> [H,K] = hermitesof(A,B,C)
Quadratic matrix variable 3x3 (symmetric, real, 2 variables)
Linear matrix variable 1x2 (full, real, 2 variables)
```

Here are some sample entries of the resulting Hermite matrix

```
>> sdisplay(H(1,1))
-0.6168744435*K(2)-0.2372594014*K(1)*K(2)+0.04745188027*K(2)^2
>> sdisplay(H(3,2))
0.3019687672*K(1)+0.01984184931*K(2)-0.009656748637*K(1)*K(2)
+0.0003260141644*K(2)^2+0.04013338907*K(1)^2
```

For this example, the Hermite matrix is quadratic in feedback matrix `K`. This Hermite matrix is expressed in Lagrange basis, with Lagrange nodes chosen as the roots of the imaginary part of a random target polynomial, see the online help of function `hermitesof` for more information. In particular, it means that each call to `hermitesof` produces different coefficients. However these coefficients have comparable magnitudes:

```
>> [H,K]=hermitesof(A,B,C);
>> sdisplay(H(1,1))
-0.9592151361*K(2)-0.3689288985*K(1)*K(2)+0.0737857797*K(2)^2
>> sdisplay(H(3,2))
6.702455704*K(1)+0.5150092145*K(2)-0.3440108908*K(1)*K(2)
+0.0184651235*K(2)^2+1.258426367*K(1)^2
```

The output of function `hermitesof` is reproducible if the user provides the roots of the target polynomial:

```
>> opt = []; opt.roots = [-1 -2 -3];
>> [H,K]=hermitesof(A,B,C,opt);
>> sdisplay(H(1,1))
-0.196969697*K(2)-0.07575757576*K(1)*K(2)+0.01515151515*K(2)^2
>> sdisplay(H(3,2))
0.2*K(1)-0.01818181818*K(2)-0.01212121212*K(1)*K(2)
+0.0007575757576*K(2)^2+0.04166666667*K(1)^2
```

The Hermite matrix can also be provided in the power basis:

```
>> opt = []; opt.basis = 'p';
>> [H,K]=hermitesof(A,B,C,opt);
>> sdisplay(H(1,1))
-13*K(2)-5*K(1)*K(2)+K(2)^2
>> sdisplay(H(3,2))
0
```

For more complicated examples, the Hermite matrix `H` is not necessarily quadratic in `K`:

```
>> [A,B1,B,C1,C] = COMPleib('NN1');
>> size(B), size(C)
ans =
     5     3
ans =
     3     5
>> [H,K]=hermitesof(A,B,C)
Polynomial matrix variable 5x5 (symmetric, real, 9 variables)
Linear matrix variable 3x3 (full, real, 9 variables)
>> degree(H)
ans =
     5
```

Appendix B: Hermite matrix in Lagrange basis for repeated interpolation points

Let us define the bivariate polynomials

$$c_{i,j}(u,v) = \frac{\partial^{i+j-2}}{\partial u^{i-1}\partial v^{j-1}}\left(\frac{a(u)b(v)-a(v)b(u)}{u-v}\right)$$

for all $i,j = 1,\ldots,n$ and denote by

$$a^{(k)}(u) = \frac{d^k a(u)}{du^k}$$

the k-th derivative of univariate polynomial $a(u)$.

Theorem 7.1 *When the interpolation points u_i are all equal, the Hermite matrix of $q(s)$ in Lagrange basis is given entrywise by*

$$H^L_{i,j} = \begin{cases} \dfrac{c_{i,j}(u_i^*, u_j)}{(i-1)!(j-1)!} & \text{if } u_i^* \neq u_j, \\[2ex] \displaystyle\sum_{k=0}^{i-1} \frac{a^{(j+k)}(u_i^*)b^{(i-k-1)}(u_j) - a^{(i-k-1)}(u_j)b^{(j+k)}(u_i^*)}{(j+k)!(i-k-1)!} & \text{otherwise,} \end{cases}$$

for all $i,j = 1,\ldots,n$.

Proof The proof of this result follows along the same lines as the proof of Theorem 5.1, with additional notational difficulties due to higher-order differentiations. □

Example 7.2 Let us choose $n = 3$ equal interpolation points ($u_1 = u_2 = u_3 = x$). According to Theorem 7.1, H^L has the following entries:

$$\begin{array}{rcl}
H^L_{11} & = & \frac{a'(x)b(x)-a(x)b'(x)}{1!} \\
H^L_{12} & = & \frac{a^{(2)}(x)b(x)-a(x)b^{(2)}(x)}{2!} \\
H^L_{13} & = & \frac{a^{(3)}(x)b(x)-a(x)b^{(3)}(x)}{3!} \\
H^L_{22} & = & \frac{a^{(2)}(x)b'(x)-a'(x)b^{(2)}(x)}{2!} + \frac{a^{(3)}(x)b(x)-a(x)b^{(3)}(x)}{3!} \\
H^L_{23} & = & \frac{a^{(3)}(x)b'(x)-a'(x)b^{(3)}(x)}{3!} + \frac{a^{(4)}(x)b(x)-a(x)b^{(4)}(x)}{4!} \\
H^L_{33} & = & \frac{a^{(3)}(x)b^{(2)}(x)-a^{(2)}(x)b^{(3)}(x)}{3!2!} + \frac{a^{(4)}(x)b'(x)-a'(x)b^{(4)}(x)}{4!} + \frac{a^{(5)}(x)b(x)-a(x)b^{(5)}(x)}{5!}.
\end{array}$$

Based on Theorem 5.1 and Theorem 7.1, we leave it to the reader to derive entrywise expressions for the Lagrange basis Hermite matrix in the general case when only some interpolation points are repeated.

Acknowledgments

This work was partly funded by TÜBITAK, the Scientific and Technological Research Council of Turkey, and by project No. 103/10/0628 of the Grant Agency of the Czech Republic.

References

[1] Z. Battles, L. N. Trefethen. An extension of Matlab to continuous functions and operators. SIAM Journal on Scientific Computing, 25(5):1743-1770, 2004.

[2] S. Fortune. An iterated eigenvalue algorithm for approximating roots of univariate polynomials. Journal of Symbolic Computation, 33(5):627-646, 2002.

[3] D. Henrion, S. Tarbouriech, M. Šebek. Rank-one LMI approach to simultaneous stabilization of linear systems. Systems and Control Letters, 38(2):79-89, 1999.

[4] D. Henrion, M. Kočvara, M. Stingl. Solving simultaneous stabilization BMI problems with PENNON. Proceedings of IFIP Conference on System Modeling and Optimization, Sophia Antipolis, France, July 2003.

[5] D. Henrion, J. Löfberg, M. Kočvara, M. Stingl. Solving polynomial static output feedback problems with PENBMI. Proceedings of joint IEEE Conference on Decision and Control (CDC) and European Control Conference (ECC), Seville, Spain, December 2005.

[6] D. Henrion, M. Šebek. Plane geometry and convexity of polynomial stability regions. Proceedings of International Symposium on Symbolic and Algebraic Computations (ISSAC), Hagenberg, Austria, July 2008.

[7] N. J. Higham. Accuracy and Stability of Numerical Algorithms. SIAM, Philadelphia, PA, 2002.

[8] E. I. Jury. Remembering four stability theory pioneers of the nineteenth century. IEEE Trans. Autom. Control, 41(9):1242-1244, 1996.

[9] F. Leibfritz. COMPl$_e$ib: constrained matrix optimization problem library: a collection of test examples for nonlinear semidefinite programs, control system design and related problems. Research report, Department of Mathematics, University of Trier, Germany, 2003. See `www.compleib.de`.

[10] D. Lemonnier, P. Van Dooren. Balancing regular matrix pencils. SIAM J. Matrix Anal. Appl., 28(1):253–263, 2006.

[11] J. Löfberg. YALMIP: a toolbox for modeling and optimization in Matlab. Proceedings of the IEEE Symposium on Computer-Aided Control System Design (CACSD), Taipei, Taiwan, September 2004. See `users.isy.liu.se/johanl/yalmip`.

[12] M. Šebek, H. Kwakernaak, D. Henrion, S. Pejchová. Recent progress in polynomial methods and Polynomial Toolbox for Matlab version 2.0. Proceedings of IEEE Conference on Decision and Control, Tampa, FL, December 1998. See `www.polyx.com`.

[13] A. Shakoori. The Bézout matrix in the Lagrange basis. Proceedings of Encuentro de Algebra Computacional y Aplicaciones (EACA), University of Cantabria, Santander, Spain, July 2004.

[14] G. A. Sitton, C. S. Burrus, J. W. Fox, S. Treitel. Factoring very high degree polynomials. IEEE Signal Processing Magazine, 20(6):27-42, 2003.

Table 3: PENBMI performance on SOF BMI problems

system	basis	μ	k_0	out. iter.	inn. iter.	lin. steps	k	λ
AC7 $n=9$	power	1	[0 0]	27	148	167	$[1.1205 \quad -3.0946\cdot 10^{-1}]$	51.640
	scaled power	1	[0 0]	34	119	191	$[1.8178 \quad 2.8823]$	38.471
	Lagrange	10^{-5}	[0 0]	15	51	67	$[5.7336 \quad 3.9995]$	$3.6356\cdot 10^{-1}$
AC17 $n=4$	power	1	[0 0]	14	65	173	$[1.6619\cdot 10^{-1} \quad 8.5782\cdot 10^{-1}]$	5.8306
	scaled power	1	[0 0]	14	85	172	$[-2.4973\cdot 10^{-2} \quad 8.1446\cdot 10^{-1}]$	5.8755
	Lagrange	1	[0 0]	16	36	57	$[-1.0855\cdot 10^{-2} \quad 1.5128\cdot 10^{-1}]$	1.0459
PAS $n=5$	power	10^{-3}	[0 0 0]	11	74	194	$[-6.5390\cdot 10^{-4} \quad -58.350 \quad -37.751]$	73.2917
	scaled power	10^{-3}	[0 0 0]	17	28	28	$[-8.8919\cdot 10^{-12} \quad -3.2912\cdot 10^{-3} \quad 1.4790\cdot 10^{-3}]$	$1.9390\cdot 10^{-13}$
	Lagrange	10^{-8}	[0 0 0]	19	27	28	$[-8.4106\cdot 10^{-6} \quad -3.9048 \quad -9.9675\cdot 10^{-1}]$	$1.4901\cdot 10^{-12}$
		10^{-5}	[0 0 0]	17	33	44	$[-3.3369\cdot 10^{-4} \quad -20.480 \quad -1.2157]$	$8.1649\cdot 10^{-3}$
REA3 $n=12$	power	1	[0 0 0]	21	28	28	$\begin{bmatrix} 0 & -1.0435\cdot 10^{-5} & -2.2281\cdot 10^{-4} \end{bmatrix}$	$8.4187\cdot 10^{-13}$
		10^{-5}	[0 0 0]	46	458	2460	$\begin{bmatrix} 0 & -43711 & -23491 \end{bmatrix}$	43787
	scaled power	10^{-1}	[0 0 0]	12	65	161	$\begin{bmatrix} 0 & -5.0576 & -7.2107\cdot 10^{-1} \end{bmatrix}$	80.448
	Lagrange	10^{-2}	[0 0 0]	16	48	68	$\begin{bmatrix} 0 & -4.2556\cdot 10^{-1} & -8.9973\cdot 10^{-2} \end{bmatrix}$	$9.9105\cdot 10^{-1}$
UWV $n=8$	power	1	[0 0; 0 0]	13	98	188	$\begin{bmatrix} -1.4319\cdot 10^{-5} & -2.6474\cdot 10^{-6} \\ -3.0817\cdot 10^{-1} & -5.6976\cdot 10^{-2} \end{bmatrix}$	27.918
	scaled power	1	[0 0; 0 0]	13	98	188	$\begin{bmatrix} -1.4319\cdot 10^{-5} & -2.6473\cdot 10^{-6} \\ -3.0817\cdot 10^{-1} & -5.6974\cdot 10^{-2} \end{bmatrix}$	27.916
	Lagrange	1	[0 0; 0 0]	15	65	82	$\begin{bmatrix} -1.6755\cdot 10^{-12} & -6.9006\cdot 10^{-13} \\ -3.6060\cdot 10^{-8} & -1.4851\cdot 10^{-8} \end{bmatrix}$	1.0000
NN5 $n=7$	power	1	[10 5]	31	139	193	$[12.382 \quad 9.0331]$	$3.9610\cdot 10^{-1}$
	scaled power	1	[10 5]	31	145	361	$[12.387 \quad 9.0368]$	$3.9564\cdot 10^{-1}$
	Lagrange	10^{-5}	[10 5]	18	45	56	$[30.931 \quad 22.295]$	$1.7652\cdot 10^{-1}$
NN1 $n=3$	power	10^{-3}	[0 30]	15	53	59	$[7.9924 \quad 72.171]$	4.2238
	scaled power	10^{-3}	[0 30]	14	44	52	$[6.9265 \quad 67.644]$	3.2762
	Lagrange	10^{-4}	[0 30]	14	49	52	$[26.936 \quad 177.20]$	4.6019
HE1 $n=4$	power	1	[1 1]	18	73	80	$[-1.5482 \quad -3.9063]$	34.359
	scaled power	1	[1 1]	20	94	107	$[-1.5423 \quad 3.8996]$	34.353
	Lagrange	10^{-1}	[1 1]	18	80	87	$[-5.1376 \quad 11.589]$	32.168