Bias correction for Vandermonde total least squares

Antonio Fazzi[a,*], Alexander Kukush[b], Ivan Markovsky[a]

[a] *Department ELEC, Vrije Universiteit Brussel (VUB), Pleinlaan 2, 1050 Brussels, Belgium*
[b] *Faculty of Mechanics and Mathematics, Taras Shevchenko National University of Kyiv, Kyiv, Ukraine*

Abstract

The low-rank approximation problem, i.e., the problem of approximating a given matrix with a matrix of lower rank, appears in many applications. In some applications the given matrix is structured and the approximation is required to have the same structure. Examples of linear structures are Hankel, Toeplitz, and Sylvester. Currently, there are only a few results for nonlinearly structured low-rank approximation problems. We consider the problem of Vandermonde structured low-rank approximation. The high condition number of the Vandermonde matrix presence in combination with the noise in the data make the the problem challenging. We propose a numerical method based on a bias correction procedure and prove consistency of the method under suitable assumptions. The performance of the method is illustrated on numerical results.

Keywords: Vandermonde matrix, bias removal, adjusted least squares, structured matrix perturbation
2000 MSC: 41A29, 68W25, 62H12, 62J10

1. Introduction

The problem considered is approximation of a matrix obtained by a perturbation of a Vandermonde matrix with another Vandermonde matrix of lower rank. Given a vector $c = (c_1, \ldots, c_n) \in \mathbb{C}^n$, the matrices we consider are rectangular Vandermonde matrices without the first row (we removed it since it is not informative), i.e.,

$$V = V(c) = \begin{pmatrix} c_1 & \cdots & c_n \\ c_1^2 & \cdots & c_n^2 \\ \vdots & \cdots & \vdots \\ c_1^m & \cdots & c_n^m \end{pmatrix}, \tag{1}$$

where the (i, j)-th entry of the matrix is the i-th power of the j-th entry of the vector c (parameter vector of V). Under the assumption that $c_i \neq c_j$ for all $i \neq j$, the matrix V is

*Corresponding author
Email addresses: Antonio.Fazzi@vub.be (Antonio Fazzi), alexander_kukush@univ.kiev.ua (Alexander Kukush), ivan.markovsky@vub.be (Ivan Markovsky)

full rank, so that the problem of low-rank approximation is aiming at coalescing two of the parameters with as small as possible perturbation. The formulation of the problem is as follows:

Problem 1. *Given a matrix of the form $V(c_0)+V(\tilde{c})$, where $V(c_0)$ is a rank-deficient Vandermonde matrix and $V(\tilde{c})$ is a Vandermonde Gaussian error matrix, we want to estimate the vector c_0.*

The Vandermonde structure is nonlinear, so the considered problem is a nonlinearly structured low-rank approximation problem.

While structured low-rank approximation problems are popular and widely studied in the literature in the case of linear (or affine) structures (see e.g. [18, 3]), there are no general efficient methods for solving the nonlinearly structured low-rank approximation problems.

The algorithm proposed in the paper computes a suboptimal solution for Problem 1 into two steps: first we compute an *adjusted least squares* estimator [12, 2] $\hat{X}$, which *corrects* the classical (biased) least squares solution of the system of equations $AX = B$, where the matrices $A \in \mathbb{C}^{m\times(n-1)}$ and $B \in \mathbb{C}^{m\times 1}$ come from the partition of the matrix V as $V = (A,B)$ and $X \in \mathbb{C}^{(n-1)\times 1}$. However, this approach does not preserve the Vandermonde structure of the data matrix, hence we need then to estimate the parameters $\hat{c} \in \mathbb{C}^n$ starting from the estimated values $\hat{A},\hat{B},\hat{X}$ computed via the adjusted least squares approach, i.e., we need to compute the sought Vandermonde matrix.

Applications. Problems of *fitting conic sections to data* [8, 16, 9] have been posed and solved as a nonlinearly structured low-rank approximation problems with polynomial structure. The problem considered in this paper — Vandermonde structured low-rank approximation — is motivated from an application in computer algebra — finding an approximate common factor for a set of given polynomials.

Let $r_1 \in \mathbb{C}^n$ be the set of roots of a polynomial $p_1(z)$ and $r_2 \in \mathbb{C}^m$ be the set of roots of $p_2(z)$. Assuming that both p_1, p_2 have simple roots, these polynomials are coprime if and only if the entries of r_1 and r_2 are pairwise different. Conversely, the presence of one (or more) common entry in r_1 and r_2 means that the polynomials p_1, p_2 have a non-trivial common factor.

Therefore, Problem 1 can be seen as a computation of distance to common divisibility for the polynomials p_1, p_2 (the problem is known in the literature also as approximate greatest common divisor or approximate common factor computation). The importance of the problem is due to its applications in the fields of system and control theory and identification [7, 4].

However, the usual definition of the distance to common divisibility problem considers a distance measure based on the coefficients of polynomials. This allows to restate the problem as a Sylvester matrix low-rank approximation problem, see [6, 10, 14, 15, 17, 18, 19, 5].

Problem 1, on the other hand, uses as a distance measure the distance between the roots of the polynomials. This formulation leads to a Vandermonde structured low-rank approximation problem. Note that the polynomials roots are uniquely defined while the coefficients are not (indeed, if we multiply the polynomials by a nonzero constant, we change the coefficients of the polynomials but not their roots).

Main contribution. In the paper we propose a suboptimal approach for the solution of Problem 1. In the first part we construct a *corrected* least squares estimator for the solution of the system of equations $AX = B$, where the matrix A collects the first $n-1$ columns of V while B is the last column of V. The proposed estimator builds an exact solution $\hat{X}$ of the system of equations $\hat{A}X = \hat{B}$, where $\hat{A}, \hat{B}$ are corrections of the matrices A, B necessary to remove the common least squares bias. However, the corrected matrix $(\hat{A}, \hat{B})$ does not have the Vandermonde structure, hence a further step is needed in order to estimate the perturbed parameter vector $\hat{c}$ (i.e., the Vandermonde matrix $V(\hat{c})$) starting from the corrected solution of the system of equations.

Notation

We denote by D^* the Hermitian adjoint of the matrix D and with D^+ the pseudo-inverse of the D. $\mathbf{E}(\cdot)$ denotes the expected value of the corresponding random variable, random vector or random matrix.

In what follows, the "exact data" $V(c_0)$ is a Vandermonde matrix generated by a vector $c_0 \in \mathbb{C}^n$ whose entries are on the unit circle and such that the last two entries coincide, i.e., $c_0(n-1) = c_0(n)$. The "noisy data" $V(\tilde{c})$ is another Vandermonde matrix generated by a vector $\tilde{c} \in \mathbb{C}^n$ of the form

$$\tilde{c}_j = \tilde{u}_j + i\tilde{v}_j, \quad \text{for all } j = 1, \ldots, n, \tag{2}$$

where $\tilde{u}_j, \tilde{v}_j, j = 1, \ldots, n$ are independent identically distributed (i.i.d.), zero mean, Gaussian with standard deviation σ.

Remark 1. *The normalization assumption on the "exact data" may seem restrictive, but it is necessary to avoid the vanishing/blow up of the entries of the Vandermonde matrix whenever large values of m are considered (see, e.g., Figure 1 about the consistency of the estimator). Although the constraint $|c_j| = 1$ for all j is needed, the estimation method does not impose it. This can be done in an extra step of projection on the unit circle.*

2. Construction of adjusted least squares estimator

In this section, we deal with a numerical solution of the system of equations $AX = B$. The classical least squares method is biased due to the noise in A [1, 13]. A classical way to approach the problem is the Total Least Squares solution, which yields (under standard assumptions) a maximum likelihood estimator [11].

The proposed approach is based on the solutions of deconvolution equations [12, 2] which guarantee that the corresponding estimators are unbiased. We analyze both the cases in which the value of σ (i.e., the standard deviation of the normal perturbation on the true data) is known and unknown. In particular, we describe in details how to build the proposed estimator in the case when σ is known, and we explain how to estimate σ in the case when it is unknown.

2.1. Known σ^2

We assume here σ is known, and we want to construct a solution to the system of equations $\hat{A}X = \hat{B}$, where the matrices $\hat{A}, \hat{B}$ are *corrections* of the matrices A, B coming from a partition of the columns of $V = (A, B)$ in (1). The score function for the starting system of equations $A_0X = B_0$ is

$$S(A_0, B_0; X) = (A_0^* A_0)X - A_0^* B_0 \tag{3}$$

associated with the classical least squares estimator. The idea of bias removal leads to the solution of the following problem: given the matrices A_0, B_0, compute a solution to a deconvolution equation of the form

$$\mathbf{E}(S_C(A, B; X)|A_0, B_0) = S(A, B; X). \tag{4}$$

Looking at the function in (3), a solution of the equation (4) is given by the solutions of the following two deconvolution equations

$$\begin{aligned} \mathbf{E}(f(A, B)|A_0, B_0) &= A_0^* B_0, \\ \mathbf{E}(g(A)|A_0) &= A_0^* A_0. \end{aligned} \tag{5}$$

The next theorem suggests how to get the sought solution.

Theorem 1. *We define the two covariance matrices*

$$\begin{aligned} V_{\tilde{A}\tilde{B}} &= \mathbf{E}(\tilde{A} - \mathbf{E}(\tilde{A}))^*(\tilde{B} - \mathbf{E}(\tilde{B})) = \mathbf{E}(\tilde{A}^*\tilde{B}) - \mathbf{E}(\tilde{A}^*)\mathbf{E}(\tilde{B}), \\ V_{\tilde{A}} &= \mathbf{E}(\tilde{A} - \mathbf{E}(\tilde{A}))^*(\tilde{A} - \mathbf{E}(\tilde{A})) = \mathbf{E}(\tilde{A}^*\tilde{A}) - \mathbf{E}(\tilde{A}^*)\mathbf{E}(\tilde{A}). \end{aligned}$$

The solution of the equations in (5) *is given by*

$$\begin{aligned} f(A, B) &= (A - \mathbf{E}(\tilde{A}))^*(B - \mathbf{E}(\tilde{B})) - V_{\tilde{A}\tilde{B}}, \\ g(A) &= (A - \mathbf{E}(\tilde{A}))^*(A - \mathbf{E}(\tilde{A})) - V_{\tilde{A}}. \end{aligned}$$

Proof. The function $f(A, B)$ can be expanded as

$$f(A, B) = A^*B - \mathbf{E}(\tilde{A}^*)B - A^*\mathbf{E}(\tilde{B}) + \mathbf{E}(\tilde{A}^*)\mathbf{E}(\tilde{B}) - V_{AB}.$$

By computing the expected value of each term in the sum we get

$$\begin{aligned} \mathbf{E}(A^*B|A_0, B_0) &= A_0^*B_0 + \mathbf{E}(\tilde{A}^*)B_0 + A_0^*\mathbf{E}(\tilde{B}) + \mathbf{E}(\tilde{A}^*\tilde{B}), \\ \mathbf{E}(A|A_0, B_0) &= A_0 + \mathbf{E}(\tilde{A}), \\ \mathbf{E}(B|A_0, B_0) &= B_0 + \mathbf{E}(\tilde{B}). \end{aligned} \tag{6}$$

Similar computations hold true for the function $g(A)$ by replacing B with A in (6). The claim follows by replacing and summing up the correct terms. □

Finally we get the estimator as

$$\hat{X} = g(A)^+ f(A, B). \tag{7}$$

Once we have found the expression of the adjusted least squares estimator, we switch to the computational issue: how to build the matrices $g(A)$ and $f(A, B)$.

Computation of covariance matrices. We show in the following how to build all the matrices $\mathbf{E}(A), \mathbf{E}(B), V_{AB}, V_A$ necessary to build the estimator $\hat{X}$ in (7). Given σ, the entries of the matrices $V_{\tilde{A}\tilde{B}}$ and $V_{\tilde{A}}$ (defined in Theorem 1) are polynomials in the parameters $\tilde{c}_j$ because of the Vandermonde structure (1) of the matrices A, B. The expected value of a polynomial is split into the expected value of monomials; in particular a monomial of the form $\bar{c}_k^p c_l^q$ has expected value [1]

$$\begin{aligned}
&\mathbf{E}((\tilde{u}_k - i\tilde{v}_k)^p(\tilde{u}_l + i\tilde{v}_l)^q) = \\
&\sum_{j=0}^{p}\sum_{t=0}^{q} \tilde{c}_k^j \tilde{c}_l^t (-i)^{p-j} i^{q-t} \mathbf{E}(\tilde{u}_k^j \tilde{u}_l^t)\mathbf{E}(\tilde{v}_k^{p-j}\tilde{v}_l^{q-t}) = \\
&\sigma^{p+q}\sum_{j=0}^{p}\sum_{t=0}^{q} \tilde{c}_k^j \tilde{c}_l^t (-i)^{p-j} i^{q-t} \mathbf{E}(\gamma_k^j \gamma_l^t)\mathbf{E}(\gamma_k^{p-j}\gamma_l^{q-t}),
\end{aligned} \tag{8}$$

where γ_i are i.i.d. random variables coming from a normal distribution with zero mean and unit variance. Their expected values are

$$\begin{aligned}
&\mu_0 = \mathbf{E}(\gamma_1^0) = 1 \\
&\mu_n = \mathbf{E}(\gamma_1^n) = \begin{cases} 0 & \text{if } n \text{ odd} \\ (2k-1)!! & \text{if } n \text{ even}, \ n = 2k \end{cases}
\end{aligned} \tag{9}$$

Replacing (9) in (8) we get the following expressions:

$$\mathbf{E}(\bar{c}_k^p c_l^q) = \begin{cases} \sigma^{p+q}\sum_{j=0}^{p}\sum_{t=0}^{q} \tilde{c}_k^j \tilde{c}_l^t (-i)^{p-j} i^{q-t} \mu_{j+t}\mu_{p+q-j-t} & \text{if } k = l \\ \sigma^{p+q}\sum_{j=0}^{p}\sum_{t=0}^{q} \tilde{c}_k^j \tilde{c}_l^t (-i)^{p-j} i^{q-t} \mu_j \mu_t \mu_{p-j}\mu_{q-t} & \text{if } k \neq l \end{cases} \tag{10}$$

Hence the covariance matrices V_{AB}, V_A are obtained as the sum of terms of the form (10). As we see from (10), each term depends on σ, the powers p, q and the indices k, l. By setting $p = 0$ in (10) we automatically find the expression for $\mathbf{E}(c_l^q)$, i.e., the expected values of the entries of a Vandermonde matrix: these values are exactly the entries of the matrices $\mathbf{E}(A), \mathbf{E}(B)$.

Remark 2. *When q is odd, we have $\mathbf{E}(c_l^q) = 0$; this is because of the symmetry of the centered Gaussian distribution.*

2.2. Unknown σ^2

We saw in the previous section how to build the estimator for given σ; therefore, in the case σ is unknown, we focus only on the estimation of σ. Once σ is estimated, we plug-in the estimate to the solution (7).

The penalty function we consider is

$$Q(A_0, B_0; X) = \|A_0 X - B_0\|_F^2, \tag{11}$$

[1] The following formula refers to Gaussian noise; each kind of error distribution has its own formula depending on the expression of the moments.

where $\|\cdot\|_F$ denotes the Frobenius norm induced by the inner product $\langle A,B\rangle = \mathrm{tr}(A^*B)$. As in the previous case, we look for a correction on the penalty function Q_c which removes the bias, i.e.,

$$\mathbf{E}(Q_c(A,B;X)|A_0,B_0) = Q(A_0,B_0;X). \tag{12}$$

Observe that the function (11) can be rewritten as

$$\begin{aligned} Q(A_0,B_0;X) &= \mathrm{tr}((A_0X-B_0)^*(A_0X-B_0)) = \\ &\mathrm{tr}(X^*A_0^*A_0X) - \mathrm{tr}(X^*A_0^*B_0) - \mathrm{tr}(B_0^*A_0X) + \mathrm{tr}(B_0^*B_0). \end{aligned} \tag{13}$$

Similar computations to the ones in Section 2 allow to replace the matrices $A_0^*A_0, A_0^*B_0$ with the functions f,g from Theorem 1. The matrix $B_0^*B_0$ is replaced by the function

$$\begin{aligned} h(B) :&= (B - \mathbf{E}(\tilde{B}))^*(B-\mathbf{E}(\tilde{B})) - V_{\tilde{B}}, \\ V_{\tilde{B}} &= \mathbf{E}(\tilde{B}^*\tilde{B}) - \mathbf{E}(\tilde{B}^*)\mathbf{E}(\tilde{B}). \end{aligned}$$

Doing so, a solution to (12) is given by the following penalty function

$$Q_c = \mathrm{tr}(X^*g(A)X) - \mathrm{tr}(X^*f(A,B)) - \mathrm{tr}(f(A,B)^*X) + \mathrm{tr}(h(B)). \tag{14}$$

However, we observe that the function in (14) depends on two variables: X and σ. But we saw in Section 2 that the estimator of X actually depends on σ, so we can write

$$\inf_{X,\sigma^2} Q_c(X,\sigma^2) = \inf_{\sigma^2}\inf_{X} Q_c(X,\sigma^2) = \inf_{\sigma^2} Q_c(\hat{X}_{\sigma^2},\sigma^2), \tag{15}$$

where $\hat{X}_{\sigma^2} = \hat{X}|\sigma^2$ denotes the value of the estimator $\hat{X}$ for given value of σ^2; moreover it is reasonable to consider only the values of σ^2 for which the matrix $g(A)$ is positive semidefinite, as it happens for the matrix $A_0^*A_0$. Finally, the estimation of σ ends up with the solution of the (univariate) optimization problem

$$\hat{\sigma}^2 = \inf_{\substack{\sigma^2 \\ g(A) \text{ positive semidefinite}}} Q_c(\hat{X}_{\sigma^2},\sigma^2). \tag{16}$$

2.3. About the estimator

We describe here some properties of the proposed estimator $\hat{X}$. First of all, we have to mention that both the error of the estimator and the estimate of $\hat{\sigma}$ depend on the dimension m of the matrix $V(c)$ (1) (for n fixed), as well as on the noise distribution. Indeed we expect that bigger values of m (more information on the data) correspond to better approximations in terms of solution (if X_0 is the solution of $A_0X_0 = B_0$, the error on the estimator is measured via the relative distance $\|\hat{X}-X_0\|_2/\|X_0\|_2$). But we see in Figure 1 what happens if we run a numerical experiment with $n = 4$ parameters and increasing values of m ($\sigma = 10^{-1}$ is known). The error decreases only up to a certain value of m[2], while we would like the value of $\hat{X}$ to approach the exact solution of the

[2]In Figure 1 we are not able to further increase m because the function in (9) blows up in the machine arithmetic.

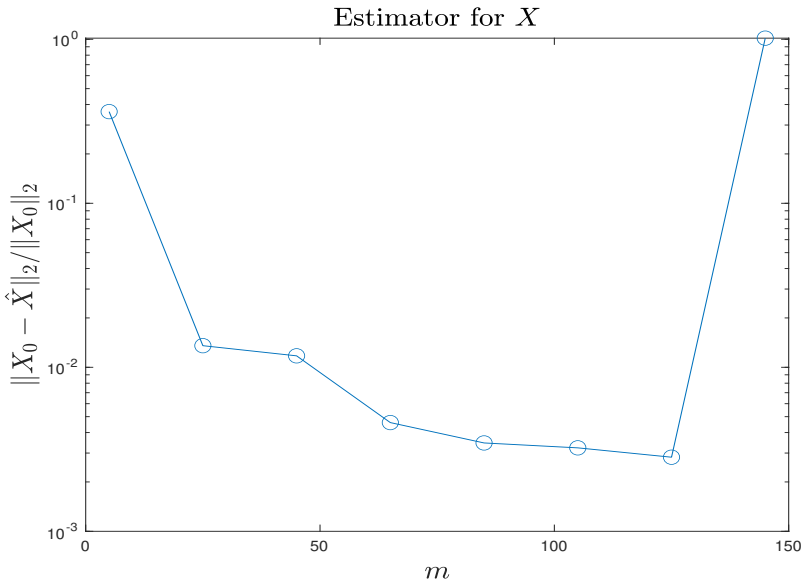

Figure 1: Relative distance between the estimator $\hat{X}$ and the true value X_0 for increasing values of m: error averaged over 50 different perturbation on the same data.

system $A_0X = B_0$ as $m \to \infty$. The observed result is actually due to the choice of the Gaussian noise on the data. It happens that the Gaussian noise makes the Vandermonde signal vanishing under the perturbation, and this is because the moments of order $2k$ of Gaussian random variables grow with k faster than the exponent of k; therefore, the solution associated to the Vandermonde system of equations can vanish under the considered perturbation.

Remark 3. *If the Gaussian distribution of the noise is replaced by a symmetrical distribution with compact support, the results for the error of the estimator $\hat{X}$ would be different with respect to the ones shown in Figure 1. This is because the error moments of order $2k$ would grow not faster than the exponent of k in this case. However, we believe that the noise with compact support (e.g., uniformly distributed noise) is not realistic in an application setting, hence the choice of the Gaussian noise.*

From the previous experiment we observe that, up to a certain point, the increasing of the value of m actually leads to an improvement in the quality of the estimator. However, the numerical values suggest that such improvement is not so relevant, therefore, a wide range of values of m should lead to numerical results which are very close.

3. Estimation of the parameters

The algorithm described in Section 2 only computes $\hat{X}$ (and estimates σ if this is unknown); however the corrected matrices $\hat{A}, \hat{B}$ do not preserve the Vandermonde

structure (1) of $(\hat{A},\hat{B})$, hence we need to refine the computed solution in order to get a perturbed parameter vector $\hat{c}$ generating a rank-deficient Vandermonde matrix. The goal of this section is to estimate the sought Vandermonde matrix $V(\hat{c}), \hat{c} \in \mathbb{C}^n$ starting from the computed solution $\hat{A},\hat{B},\hat{X}$. We explain in the following the proposed algorithm.

1. First of all, we stack the matrices $\hat{A},\hat{B}$ in a matrix $\hat{C} = (\hat{A},\hat{B})$. Consider then the matrix $\hat{D} = \hat{C}^*\hat{C}$: by construction the matrix $\hat{D}$ satisfies the deconvolution equation
$$\mathbf{E}(\hat{D}|C_0) = C_0^* C_0. \tag{17}$$
2. We exploit the structure of the matrix $\hat{D}$ and equation (17) to estimate the modulus $\hat{r}_j = |\hat{c}_j|$ for all j of each complex number. If we denote as $\hat{d}_{ij}$ the entries of $\hat{D}$, using equation (17), we need to solve the equation
$$\hat{d}_{jj} = \sum_{k=1}^{n} \hat{r}_j^{2(k-1)}. \tag{18}$$
This is because, if C is a Vandermonde matrix generated by a vector $c \in \mathbb{C}^n$, the i-th diagonal entry of the square matrix C^*C is a (univariate) polynomial in c_i. It holds that, if $\hat{d}_{jj} > 1$, the equation has a unique real solution (we are looking for a modulus, i.e., a positive real number); if $\hat{d}_{jj} < 1$ we set the corresponding $\hat{r}_j$ equal to zero.
3. We set $\hat{r}_{j^*} = \max_j \hat{r}_j$. Assuming $r_{j^*} > 0$, we choose $\hat{c}_{j^*} = r_{j^*}$ (this is a real number while $\hat{c}_{j^*}$ should be a complex number; this choice is only temporary though). We notice then that, if C is a Vandermonde matrix generated by a vector $c \in \mathbb{C}^n$, the entry indexed by i, j with $i \neq j$ (which are the entries out of the diagonal) is a polynomial in the two variables c_i, c_j. By adopting the polar representation for complex numbers, we can write the estimators of sought parameters as $\hat{c}_j = \exp\big(i\hat{\phi}\big)\hat{r}_j$, $j \neq j^*$, where $\hat{r}_j$ comes from the solution computed at step 2, while $0 \leq \hat{\phi} < 2\pi$ is a solution of the equation
$$\hat{d}_{j^* j} = \sum_{k=1}^{n} (\hat{r}_{j^*}\hat{r}_j)^{k-1} \exp\big(i\hat{\phi}_j(k-1)\big). \tag{19}$$
Observe that the solution of this last equation is not unique.
4. As last step, we rotate all the current estimates of the parameters by an angle θ: we set $\hat{c}_j = \hat{r}_j \exp\big(i\hat{\phi}_j\big)\exp\big(i\hat{\theta}\big)$, where $\hat{\theta}$ is chosen as a scalar between 0 and 2π which solves the optimization problem
$$\min_{\hat{\theta}} \|\hat{A}(\hat{c}(\hat{\theta}))\hat{X} - \hat{B}(\hat{c}(\hat{\theta}))\|_2,$$
where $(\hat{A}(\hat{c}(\hat{\theta})),\hat{B}(\hat{c}(\hat{\theta}))) = V(\hat{c}(\hat{\theta}))$ and $\hat{X}$ is the estimator computed through the algorithm in Section 2.
5. Once we get a solution, we need to observe that the possible multiple solutions of equation (19) can lead to a set of parameters whose real and imaginary parts have different signs and/or are swapped. As last refinement we compare the computed solution with the starting parameters c in order to possibly swap the real and imaginary parts or changing some signs.

Remark 4. *We note that it is not easy to recover exactly some complex numbers α, β from a polynomial value $p(\alpha,\beta)$. While the estimation of the modulus is an easier problem, the estimation of the phase is more complicated also in the noise-free case (we actually did not run the algorithm at all if the starting matrix is rank-deficient). This is due to the sampling of the interval $(0,2\pi)$, or in general to the several numerical approximations due to the machine arithmetic.*

4. Numerical experiments

In this section, we report results of numerical experiments that verify the properties and illustrate the performance of the proposed estimation method. We do experiments in both the cases when the value of σ is known and unknown. The setup of the experiments is as follows:

1. we generate $n-1$ (random) complex roots $c_1,\ldots,c_{n-1}$ on the unit circle and then we repeat one of them (assume it is the last); the vector $(c_1,\ldots,c_{n-1},c_{n-1})$ generates a rank-deficient Vandermonde matrix;
2. we perturb the starting Vandermonde matrix by adding a second Vandermonde matrix generated by a random complex vector whose entries are i.i.d. white Gaussian coming from a distribution of the form $N(0,\sigma^2)$;
3. we show the estimation results averaged over different perturbations (of increasing noise levels) on the starting parameters.

The standard deviation of the noise distribution varies in the interval $(0,0.3)$; the considered estimation error is the norm of the difference between the computed parameters $\hat{c}$ and the starting true parameters c_0: $\|c_0-\hat{c}\|$. We run each time several experiments with different perturbations and the same noise levels on the parameters c_0, and we consider the average error (the number of experiments is different and it depends on the computational time needed by the algorithm). Moreover we plot also the (averaged) smallest singular value of the Vandermonde matrix associated with the computed solution.

Estimation of σ^2 We check first how the estimate of σ^2 performs in the case σ is unknown. In this paragraph we focus on the estimation of σ only, by solving the univariate optimization problem (15), while we deal with the parameters estimation later. This problem is associated with the estimator built in Section 2. The following experiment is built following the setup described in Section 4, but we compare the true value of σ (the one we used to generate the data) with the estimated one. Despite we observed that the estimator is consistent (up to a certain value of m), we consider a smaller matrix since the experiment would be time consuming otherwise; as it emerges from Figure 1 the numerical values corresponding to bigger values of m would be more accurate but not so far.

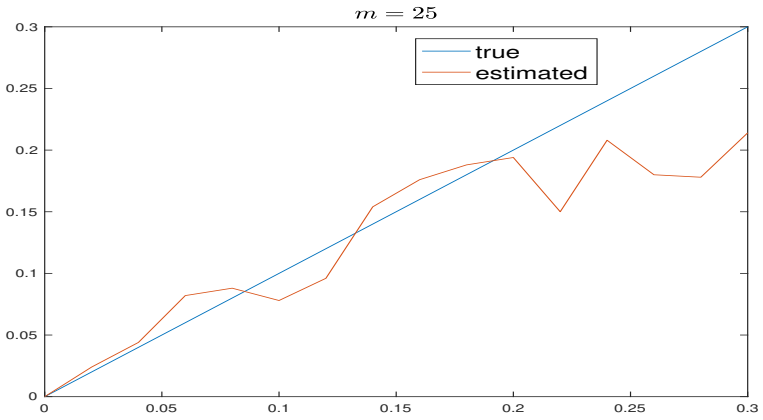

Figure 2: Estimation of σ: result averaged over 10 different perturbation on the data associated with normal distributions with the same standard deviation.

The size of the considered Vandermonde matrix is $m = 25$, so it is a medium value in between the values of m which make the estimator consistent. We did not consider bigger values of m since the experiment is time consuming: each time we run an experiment with known σ, for all $\sigma \in (0, 0.3)$, by using the available data and then we solve the optimization problem (16).

We observe that, for small values of σ (approximately the first half of the plot), the estimated values are very close to the exact ones. The estimates become worse for higher values of σ, but the difference from the exact values is not so high.

Parameters estimation Once we know the value σ^2 (it can be known or estimated through the solution of (15)) we can run some experiments in order to test the algorithm for estimation of the parameters (as described in Section 3). In the experiments we fix the number of parameters n and we run the estimation algorithm for different values of σ (in the interval $(0, 0.3)$).

We plot in the following the error on the parameters ($\|\hat{c} - c\|_2$) and the (relative) distance to singularity for the computed matrices (we remember they are Vandermonde matrices without the first row). This is because we want to understand how close the matrix $V(\hat{c})$ is to the rank-deficiency. The following figures show the results for 4 parameters with $m = 5$ (Figure 3).

We see how the error slightly increases with σ. The value of m is small in order to avoid slow computations with big matrices (observe we ran 500 experiments for each σ). If we look at the second plot, i.e., the relative distance to singularity, we see that such numerical values are closer to 1 than 0. This is because, in the estimated parameters, there are no two overlapping roots but only two close roots; moreover Vandermonde-like matrices are known to be ill conditioned, indeed the condition numbers of the (relatively small) involved matrices are quite big.

We wonder what happens if we increase the value of m. We saw in Figure 1 that the error on the estimator X decreases (up to a certain point), so it makes sense to inves-

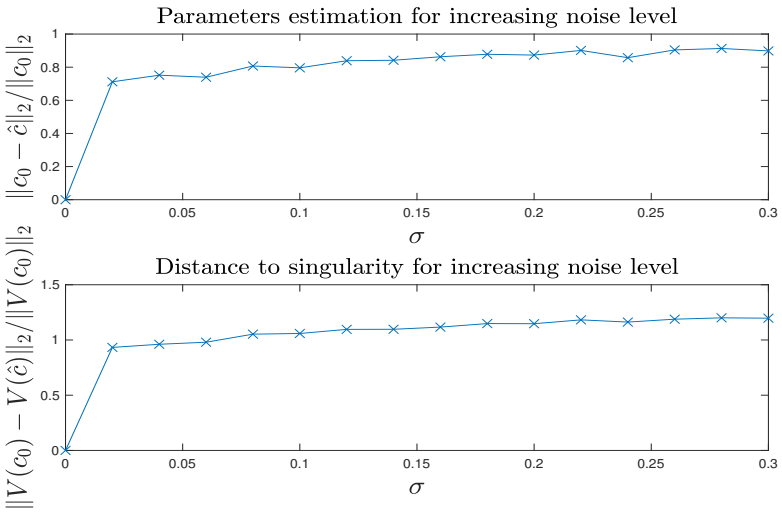

Figure 3: Vandermonde low-rank approximation for 4 complex parameters. Error on the generating parameters (top) and relative distance to singularity of the Vandermonde matrix (bottom) averaged over 500 different perturbations.

tigate what happens for the estimated parameters. We show in Figure 4 what happens for two different values of m (the two matrices are generated from the same parameter vector but the dimensions are different). We see that bigger values of m correspond to bigger errors on the estimated parameters and bigger distances to singularity. Moreover we needed to lower the considered value of σ since the distance to singularity of the computed Vandermonde matrix blows up if σ is too high. Concluding, increasing the value of m improves the value of the estimator $\hat{X}$ on one hand, but it leads to worse values for the parameters estimation problem, as well for the distance to singularity. A trade-off between the two errors would be the right balance.

Remark 5. *The estimation of the parameter works in a similar way even without the normalization assumption on the "exact data", as described in the paragraph Notation in Section 1. This may be useful in the application desscribed in Section 1. However, we need to pay attention to the value of m since, in this case, we could lose the consistency of the estimator because the possible vanishing/blow up of the entries of the matrix can cause misleading numerical results.*

5. Conclusion

We started from a rank-deficient Vandermonde matrix and, after the addition of some noise, we proposed a method for its recovery. This is based on the unbiased solution of a system of equations, followed by the estimation of the sought structured matrix. However, the problem is not so easy to deal with: if, on one hand, one can investigate different computational techniques for both the solution of the system of

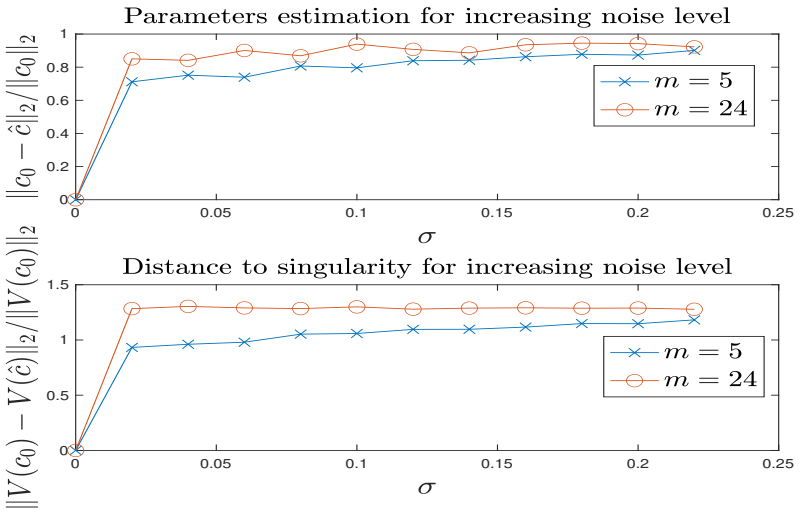

Figure 4: Vandermonde low-rank approximation for 4 complex parameters and two different values of m. Error on the generating parameters (top) and relative distance to singularity of the Vandermonde matrix (bottom) averaged over 500 different perturbation.

equations and the parameters estimation, on the other hand, the high condition numbers of Vandermonde matrices make hard the exact recovery of the parameters, even for small errors on the data.

Acknowledgements. The research leading to these results has received funding from the Fond for Scientific Research Vlaanderen (FWO) projects G028015N and G090117N and the FNRS-FWO under Excellence of Science (EOS) Project no 30468160 “Structured low-rank matrix / tensor approximation: numerical optimization-based algorithms and applications” and also from National Research Foundation of Ukraine under Project no 20DF038-02 “Parameter estimation, hypothesis testing and prediction in actual stochastic models".

References

[1] G.Q. Carapia, I. Markovsky, R. Pintelon, P. Csurcsia, and D. Verbeke. Bias and covariance of the least squares estimate in a structured errors-in-variables problem. *Computational Statistics & Data Analysis*, 144:106893, 2020.

[2] C.-L. Cheng and H. Schneeweiss. Polynomial regression with errors in the variables. *J. Royal Statist. Society Series B (Statist. Mathodol.)*, (60):189–199, 1998.

[3] M. T. Chu, R. E. Funderlic, and L. J. Plemmons. Structured low rank approximation. *Linear Algebra Its Appl.*, 366:157–172, 2003.

[4] V. De Iuliis, A. Germani, and C. Manes. Identification of forward and feedback transfer functions in closed-loop systems with feedback delay. *IFAC-PapersOnLine*, 50(1):12847 – 12852, 2017.

[5] A. Fazzi, N. Guglielmi, and I. Markovsky. An ODE based method for computing the Approximate Greatest Common Divisor of polynomials. *Numer. Algorithms*, 81:719–740, 2019.

[6] N. Guglielmi and I. Markovsky. An ODE based method for computing the distance of coprime polynomials to common divisibility. *SIAM J. Numer. Anal.*, 55:1456–1482, 2017.

[7] I. Markovsky, A. Fazzi, and N. Guglielmi. Applications of polynomial common factor computation in signal processing. In Y. Deville et al., editor, *Latent Variable Analysis and Signal Separation*, volume 10891 of *Lecture Notes in Computer Science*, pages 99–106. Springer, 2018.

[8] I. Markovsky, A. Kukush, and S. Van Huffel. Consistent least squares fitting of ellipsoids. *Numerische Mathematik*, 98(1):177–194, 2004.

[9] I. Markovsky and K. Usevich. Nonlinearly structured low-rank approximation. In Yun Raimond Fu, editor, *Low-Rank and Sparse Modeling for Vusual Analysis*, pages 1–22. Springer, 2014.

[10] I. Markovsky and S. Van Huffel. An algorithm for approximate common divisor computation. In *Proceedings of MTNS'06*, pages 274–279, Kyoto, Japan, 2006.

[11] I. Markovsky and S. Van Huffel. Overview of total least squares methods. *Signal Process*, 87:2283–2302, 2007.

[12] S.V. Masiuk, A. G. Kukush, S. V. Shklyar, M. I. Chepurny, and I. A. Likhtarov. *Radiation Risk Estimation: Based on Measurement Error Models*. de Gruyter series in Mathematics and Life Sciences, vol. 5. de Gruyter, 2nd edition, 2017.

[13] R. Meir. Bias, variance and the combination of least squares estimators. In G. Tesauro, D. S. Touretzky, and T. K. Leen, editors, *Advances in Neural Information Processing Systems 7*, pages 295–302. MIT Press, 1995.

[14] V. Y. Pan. Computation of approximate polynomial GCDs and an extension. *Information and Computation*, 167:71–85, 2001.

[15] W. Qiu, Y. Hua, and K. Abed-Meraim. A subspace method for the computation of the GCD of polynomials. *Automatica*, 33:741–743, 1997.

[16] S. Shklyar, A. Kukush, I. Markovsky, and S. Van Huffel. On the conic section fitting problem. *Journal of Multivariate Analysis*, 98:588–624, 2007.

[17] A. Terui. GPGCD: An iterative method for calculating approximate GCD of univariate polynomials. *Theoretical Computer Science*, 479:127–149, 2013.

[18] K. Usevich and I. Markovsky. Variable projection for affinely structured low-rank approximation in weighted 2-norms. *J. Comput. Appl. Math.*, 272:430–448, 2014.

[19] Z. Zeng. The numerical greatest common divisor of univariate polynomials. *Contemp. Math.*, 556:187–217, 2011.