Macaulay matrices, low displacement rank, and the efficient computation of null spaces

Nithin Govindarajan

Back to the roots seminar, March 5th 2024

KU LEUVEN

Overview

Motivation: Macaulay method for solving (overdetermined) polynomial systems
Low displacement rank (LDR) matrices and the GKO algorithm
A fast null space algorithm for Macaulay matrices exploiting LDR properties
Challenges of extending LDR theory in higher dimensions
Conclusions and additional remarks

Overview

Motivation: Macaulay method for solving (overdetermined) polynomial systems

Low displacement rank (LDR) matrices and the GKO algorithm

A fast null space algorithm for Macaulay matrices exploiting LDR properties

Challenges of extending LDR theory in higher dimensions

Conclusions and additional remarks

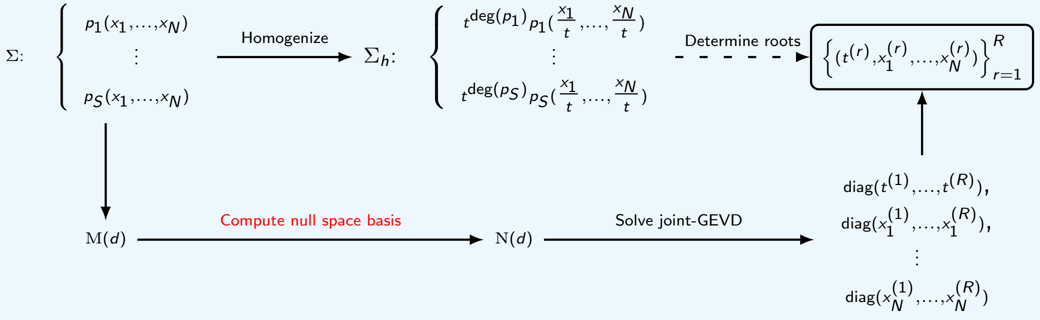
Solving system of multivariate polynomials

Let $S \geq N$, find *isolated* roots of the polynomial system

$$\Sigma : \quad \begin{cases} p_1 & = & p_1(x_1, x_2, \ldots, x_N) \\ & \vdots & \\ p_S & = & p_S(x_1, x_2, \ldots, x_N) \end{cases} \tag{1}$$

NOTE: In practice, we may be only interested in the affine roots...

One approach to compute roots (Vanderstukken and De Lathauwer 2021)



The rows of the Macaulay matrix $\mathrm{M}(d)$ span the *set of polynomial combinations*:

$$\left\{ \sum_{s=1}^{S} h_s \cdot p_s : \quad \deg(h_s) = d - d_s \right\}.$$

Computing null spaces of Macaulay-type matrices is important!

Null spaces of Macaulay-type matrices are also required in:

1. The GEVD method of *Dreesen*, *Batselier*, and *De Moor*.
2. Truncated normal form methods of *Telen*, *Mourrain*, *Van Barel*.
3. The Julia package EigenvalueSolver.jl of *Bender* and *Telen*.
4. *Vermeersch* and *De Moor*'s block-Macaulay method for multiparameter eigenvalue problems.
5. ... and other related methods ...

Macaulay matrices of univariate systems are block-Toeplitz

$$
\begin{aligned}
p_1(x) &= 4 + 2x + 1x^2 + 2x^3 + 5x^4 \\
p_2(x) &= 2 + 1x + 4x^2 + 3x^3 + 2x^4
\end{aligned}
$$

$$
M(8) = \begin{array}{c} \\ p_1 \\ p_2 \\ xp_1 \\ xp_2 \\ x^2p_1 \\ x^2p_2 \\ x^3p_1 \\ x^3p_2 \\ x^4p_1 \\ x^4p_2 \end{array}
\begin{array}{c} \begin{array}{ccccccccc} 1 & x & x^2 & x^3 & x^4 & x^5 & x^6 & x^7 & x^8 \end{array} \\
\left[\begin{array}{ccccccccc}
4 & 2 & 1 & 2 & 5 & & & & \\
2 & 1 & 4 & 3 & 2 & & & & \\
 & 4 & 2 & 1 & 2 & 5 & & & \\
 & 2 & 1 & 4 & 3 & 2 & & & \\
 & & 4 & 2 & 1 & 2 & 5 & & \\
 & & 2 & 1 & 4 & 3 & 2 & & \\
 & & & 4 & 2 & 1 & 2 & 5 & \\
 & & & 2 & 1 & 4 & 3 & 2 & \\
 & & & & 4 & 2 & 1 & 2 & 5 \\
 & & & & 2 & 1 & 4 & 3 & 2
\end{array}\right] \end{array}
$$

Macaulay matrices of bivariate systems are (almost) Toeplitz block-(block-)Toeplitz!

$$
\begin{aligned}
p_1(x,y) &= 1 + 6x + 4x^2 + 2y + 5xy + 3y^2 \\
p_2(x,y) &= 9 + 1x + 3x^2 + 8y + 7xy + 2y^2
\end{aligned}
$$

$M(4) =$

	1	x	x^2	x^3	x^4	y	xy	x^2y	x^3y	y^2	xy^2	x^2y^2	y^3	xy^3	y^4
p_1	1	6	4			2	5			3					
p_2	9	1	3			8	7			2					
xp_1		1	6	4			2	5			3				
xp_2		9	1	3			8	7			2				
x^2p_1			1	6	4			2	5			3			
x^2p_2			9	1	3			8	7			2			
yp_1						1	6	4		2	5		3		
yp_2						9	1	3		8	7		2		
xyp_1							1	6	4		2	5		3	
xyp_2							9	1	3		8	7		2	
y^2p_1										1	6	4	2	5	3
y^2p_2										9	1	3	8	7	2

Macaulay matrices of systems with $N \geq 3$ variables

- In trivariate case: Block-Toeplitz where each is Toeplitz-block-(block-)Toeplitz
- For $N \geq 4$: higher-order generalizations of the above...

Overview

Motivation: Macaulay method for solving (overdetermined) polynomial systems

Low displacement rank (LDR) matrices and the GKO algorithm

A fast null space algorithm for Macaulay matrices exploiting LDR properties

Challenges of extending LDR theory in higher dimensions

Conclusions and additional remarks

Low displacement rank (LDR) matrices

LDR matrix of the Sylvester-type

A matrix $\mathrm{X} \in \mathbb{F}^{n \times n}$ has *low displacement rank* with respect to displacement matrices $\mathrm{A}, \mathrm{B} \in \mathbb{F}^{n \times n}$ if rank $\mathcal{D}_{\mathrm{A},\mathrm{B}}\{\mathrm{X}\} \ll n$ with

$$\mathcal{D}_{\mathrm{A},\mathrm{B}} : \quad \mathrm{X} \mapsto \mathrm{AX} - \mathrm{XB}.$$

One could also work with the Stein equation $\mathrm{X} - \mathrm{AXB}$, leading to different family of algorithms...

An example: Toeplitz matrices

$$\mathrm{Z}_{n,\varphi} := \begin{bmatrix} & & & \varphi \\ 1 & & & \\ & \ddots & & \\ & & 1 & \end{bmatrix} \in \mathbb{C}^{n\times n}.$$

$$\begin{bmatrix} & & & 1 \\ 1 & & & \\ & 1 & & \\ & & 1 & \end{bmatrix} \begin{bmatrix} t_0 & t_1 & t_2 & t_3 \\ t_{-1} & t_0 & t_1 & t_2 \\ t_{-2} & t_{-1} & t_0 & t_1 \\ t_{-3} & t_{-2} & t_{-1} & t_0 \end{bmatrix} - \begin{bmatrix} t_0 & t_1 & t_2 & t_3 \\ t_{-1} & t_0 & t_1 & t_2 \\ t_{-2} & t_{-1} & t_0 & t_1 \\ t_{-3} & t_{-2} & t_{-1} & t_0 \end{bmatrix} \begin{bmatrix} & & & \varphi \\ 1 & & & \\ & 1 & & \\ & & 1 & \end{bmatrix}$$

$$=$$

$$\begin{bmatrix} t_{-3} - t_1 & t_{-2} - t_2 & t_{-1} - t_3 & t_0 - \varphi t_0 \\ 0 & 0 & 0 & t_3 - \varphi t_{-1} \\ 0 & 0 & 0 & t_2 - \varphi t_{-2} \\ 0 & 0 & 0 & t_1 - \varphi t_{-3} \end{bmatrix}$$

The displacement operator produces a compact representation of the Toeplitz matrix

If $\varphi \neq 1$, U and V are generators of the Toeplitz matrix

$$\begin{bmatrix} t_{-3}-t_1 & t_{-2}-t_2 & t_{-1}-t_3 & t_0-\varphi t_0 \\ 0 & 0 & 0 & t_3-\varphi t_{-1} \\ 0 & 0 & 0 & t_2-\varphi t_{-2} \\ 0 & 0 & 0 & t_1-\varphi t_{-3} \end{bmatrix} = \underbrace{\begin{bmatrix} 1 & 0 \\ 0 & t_3-\varphi t_{-1} \\ 0 & t_2-\varphi t_{-2} \\ 0 & t_1-\varphi t_{-3} \end{bmatrix}}_{=:\mathrm{U}} \underbrace{\begin{bmatrix} t_{-3}-t_1 & t_{-2}-t_2 & t_{-1}-t_3 & t_0-\varphi t_0 \\ 0 & 0 & 0 & 1 \end{bmatrix}}_{=:\mathrm{V}^*}$$

Not all generators produce Toeplitz matrices!

LDR matrices associated with $Z_{n,1}X - XZ_{n,\varphi}$ are called (in this context) Toeplitz-like

As *an example*, $T_1^{-1}T_2$ with $T_1, T_2 \in \mathbb{C}^{n\times n}$ Toeplitz is *Toeplitz-like*.

$$\begin{bmatrix} Z_{n,1} & \\ & Z_{n,1} \end{bmatrix} \begin{bmatrix} T_1 & T_2 \\ -I & 0 \end{bmatrix} - \begin{bmatrix} T_1 & T_2 \\ -I & 0 \end{bmatrix} \begin{bmatrix} Z_{n,\varphi} & \\ & Z_{n,\varphi} \end{bmatrix} \equiv \text{rank } 4$$

$$\downarrow$$

$$G_1 \begin{bmatrix} Z_{n,1} & \\ & Z_{n,1} \end{bmatrix} G_1^{-1} G_1 \begin{bmatrix} T_1 & T_2 \\ -I & 0 \end{bmatrix} G_2 - G_1 \begin{bmatrix} T_1 & T_2 \\ -I & 0 \end{bmatrix} G_2 G_2^{-1} \begin{bmatrix} Z_{n,\varphi} & \\ & Z_{n,\varphi} \end{bmatrix} G_2^{-1} \equiv \text{rank } 4$$

$$\downarrow$$

$$\begin{bmatrix} Z_{n,1} & \\ * & Z_{n,1} \end{bmatrix} \begin{bmatrix} T_1 & \\ & T_1^{-1}T_2 \end{bmatrix} - \begin{bmatrix} T_1 & \\ & T_1^{-1}T_2 \end{bmatrix} \begin{bmatrix} Z_{n,\varphi} & * \\ & Z_{n,\varphi} \end{bmatrix} \equiv \text{rank } 4$$

$$\downarrow$$

$$Z_{n,1}\left(T_1^{-1}T_2\right) - \left(T_1^{-1}T_2\right) Z_{n,\varphi} \equiv \text{rank } 4$$

Another example: a pure Cauchy matrix

$$
\begin{bmatrix} \mu_1 & & & \\ & \mu_2 & & \\ & & \mu_3 & \\ & & & \mu_4 \end{bmatrix}
\begin{bmatrix} \frac{1}{\mu_1-\nu_1} & \frac{1}{\mu_1-\nu_2} & \frac{1}{\mu_1-\nu_3} & \frac{1}{\mu_1-\nu_4} \\ \frac{1}{\mu_2-\nu_1} & \frac{1}{\mu_2-\nu_2} & \frac{1}{\mu_2-\nu_3} & \frac{1}{\mu_2-\nu_4} \\ \frac{1}{\mu_3-\nu_1} & \frac{1}{\mu_3-\nu_2} & \frac{1}{\mu_3-\nu_3} & \frac{1}{\mu_3-\nu_4} \\ \frac{1}{\mu_4-\nu_1} & \frac{1}{\mu_4-\nu_2} & \frac{1}{\mu_4-\nu_3} & \frac{1}{\mu_4-\nu_4} \end{bmatrix}
-
\begin{bmatrix} \frac{1}{\mu_1-\nu_1} & \frac{1}{\mu_1-\nu_2} & \frac{1}{\mu_1-\nu_3} & \frac{1}{\mu_1-\nu_4} \\ \frac{1}{\mu_2-\nu_1} & \frac{1}{\mu_2-\nu_2} & \frac{1}{\mu_2-\nu_3} & \frac{1}{\mu_2-\nu_4} \\ \frac{1}{\mu_3-\nu_1} & \frac{1}{\mu_3-\nu_2} & \frac{1}{\mu_3-\nu_3} & \frac{1}{\mu_3-\nu_4} \\ \frac{1}{\mu_4-\nu_1} & \frac{1}{\mu_4-\nu_2} & \frac{1}{\mu_4-\nu_3} & \frac{1}{\mu_4-\nu_4} \end{bmatrix}
\begin{bmatrix} \nu_1 & & & \\ & \nu_2 & & \\ & & \nu_3 & \\ & & & \nu_4 \end{bmatrix}
$$

$$
=
$$

$$
\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix}
$$

If $\mu_i \neq \mu_j$, the displacement operator is invertible.

LDR matrices associated with $\mathrm{diag}(\boldsymbol{\mu})\mathrm{X} - \mathrm{X}\,\mathrm{diag}(\boldsymbol{\nu})$ are called Cauchy-like

A rank-r *Cauchy-like* matrix $\mathrm{C} \in \mathbb{C}^{n\times n}$ has entries:

$$[C]_{ij} = \frac{\boldsymbol{u}_i^* \boldsymbol{v}_j}{\mu_i - \nu_j}, \qquad \boldsymbol{u}_i, \boldsymbol{v}_j \in \mathbb{C}^r.$$

E.g., Loewner matrices with entries $\frac{\xi_i - \eta_j}{\mu_i - \nu_j}$ are rank-2 *Cauchy-like*.

The Gohberg-Kailath-Olshevsky (GKO) algorithm for Cauchy-like matrices

$$\begin{bmatrix} \mathrm{diag}(\boldsymbol{\mu}_1) & \\ & \mathrm{diag}(\boldsymbol{\mu}_2) \end{bmatrix} \begin{bmatrix} \mathrm{A} & \mathrm{G}^* \\ \mathrm{F} & \mathrm{B} \end{bmatrix} - \begin{bmatrix} \mathrm{A} & \mathrm{G}^* \\ \mathrm{F} & \mathrm{B} \end{bmatrix} \begin{bmatrix} \mathrm{diag}(\boldsymbol{\nu}_1) & \\ & \mathrm{diag}(\boldsymbol{\nu}_2) \end{bmatrix} = \begin{bmatrix} \mathrm{R}_1 \\ \mathrm{R}_2 \end{bmatrix} \begin{bmatrix} \mathrm{S}_1 \\ \mathrm{S}_2 \end{bmatrix}^*$$

$$\downarrow$$

$$\mathrm{diag}(\boldsymbol{\mu}_2) \left(\mathrm{B} - \mathrm{F}\mathrm{A}^{-1}\mathrm{G}^*\right) - \left(\mathrm{B} - \mathrm{F}\mathrm{A}^{-1}\mathrm{G}^*\right) \mathrm{diag}(\boldsymbol{\nu}_2) = \left(\mathrm{R}_2 - \mathrm{F}\mathrm{A}^{-1}\mathrm{R}_1\right) \left(\mathrm{S}_2 - \mathrm{G}(\mathrm{A}^*)^{-1}\mathrm{S}_1\right)^*$$

Main principles behind $O(n^2)$ GKO algorithm:

- *Schur complements* preserve Cauchy → use *generators* for Gauss elimination.
- *Permutations* preserve Cauchy → partial pivoting for improved numerical stability

Stability of GKO can be further enhanced through *approximate* complete pivoting

$$\mathrm{diag}(\boldsymbol{\mu})\mathrm{C} - \mathrm{C}\mathrm{diag}(\boldsymbol{\nu}) =: \mathrm{G} = \begin{bmatrix} | & & | \\ \boldsymbol{q}_1 & \cdots & \boldsymbol{q}_r \\ | & & | \end{bmatrix} \begin{bmatrix} w_{11} & w_{12} & \cdots & w_{1n} \\ \vdots & \vdots & & \vdots \\ w_{11} & w_{12} & \cdots & w_{1n} \end{bmatrix}.$$

(Gu 1998, Lemma 3.1)

The column with the *largest* 2-norm in G (denoted $j_{\max}$) satisfies:

$$\max_{1\le i\le n} |c_{ij_{\max}}| \ge \frac{1}{K(\boldsymbol{\mu},\boldsymbol{\nu})\sqrt{n}} \max_{1\le i,j\le n} |c_{ij}|.$$

If $r \ll n$ and Q is orthonormal, finding $j_{\max}$ is cheap

Gauss updates *destroy* orthonormality $\rightarrow$ **extra cost:** QR-decomposition at each step!

Many LDR matrices are efficiently converted to Cauchy-like (Heinig 1995)

Assume A, B have *"fast" eigendecompositions*

$$\mathrm{AX} - \mathrm{XB} = \mathrm{UV}^*$$

$$\downarrow$$

$$\left(\mathrm{W}_A \Lambda_A \mathrm{W}_A^{-1}\right) \mathrm{X} - \mathrm{X} \left(\mathrm{W}_B \Lambda_B \mathrm{W}_B^{-1}\right) = \mathrm{UV}^*$$

$$\downarrow$$

$$\Lambda_A \mathrm{W}_A^{-1} \mathrm{X} \mathrm{W}_B - \mathrm{W}_A^{-1} \mathrm{X} \mathrm{W}_B \Lambda_B = \left(\mathrm{W}_A^{-1} \mathrm{U}\right) \left(\mathrm{W}_B^* \mathrm{V}\right)^*$$

$$\downarrow$$

$$\mathrm{diag}(\boldsymbol{\mu}) \mathrm{C} - \mathrm{C} \mathrm{diag}(\boldsymbol{\nu}) = \mathrm{RS}^*$$

$\mathrm{Z}_{n,\varphi}$ has a fast eigendecomposition, thus Toeplitz-like systems are solved fast using GKO!

Denote $\omega_n := \exp(-2\pi\iota/n)$ and $\mathrm{F}_n \in \mathbb{C}^{n\times n}$ the (unitary) DFT matrix. Then

$$\mathrm{Z}_{n,\varphi} = (\mathrm{D}_{n,\varphi} F_n)(\varphi^{1/n}\Omega_n)(\mathrm{D}_{n,\varphi}\mathrm{F}_n)^{-1},$$

where $\mathrm{D}_{n,\varphi} := \mathrm{diag}(1, \varphi^{-1/n}, \ldots, \varphi^{-(n-1)/n})$, $\Omega_n := \mathrm{diag}(1, \bar{\omega}_n, \ldots, \bar{\omega}_n^{n-1})$,.

Overview

Motivation: Macaulay method for solving (overdetermined) polynomial systems

Low displacement rank (LDR) matrices and the GKO algorithm

A fast null space algorithm for Macaulay matrices exploiting LDR properties

Challenges of extending LDR theory in higher dimensions

Conclusions and additional remarks

$\mathrm{M}(d)$ is LDR for univariate systems

Assuming $\deg(p_s) = d_\Sigma$ for $s = 1, 2, \ldots, S$,

$$\mathrm{M}(d) := \begin{bmatrix} \boldsymbol{c}_{0i} & \boldsymbol{c}_{1i} & \cdots & \boldsymbol{c}_{(d_\Sigma - i)i} & & & \\ & \boldsymbol{c}_{0i} & \boldsymbol{c}_{1i} & \cdots & \boldsymbol{c}_{(d_\Sigma - i)i} & & \\ & & \ddots & \ddots & & \ddots & \\ & & & \boldsymbol{c}_{0i} & \boldsymbol{c}_{1i} & \cdots & \boldsymbol{c}_{(d_\Sigma - i)i} \end{bmatrix} \in \mathbb{C}^{S(\Delta d+1)\times(d+1)}.$$

For $\mathcal{D} : \mathrm{X} \mapsto (\mathrm{Z}_{d-d_\Sigma,1} \otimes \mathrm{I}_S)\,\mathrm{X} - \mathrm{X}\mathrm{Z}_{d+1,\varphi}$, we have

$$\operatorname{rank} \mathscr{D}\{\mathrm{M}(d)\} = 2S, \quad \text{irrespective of } d.$$

The Macaulay matrix for the general bivariate case

Assuming $\deg(p_s) = d_\Sigma$ for $s = 1, 2, \ldots, S$,

$$\mathrm{M}(d) := \begin{bmatrix} \mathrm{M}_{0,0} & \mathrm{M}_{1,0} & \cdots & \mathrm{M}_{d_\Sigma,0} & & & \\ & \mathrm{M}_{0,1} & \mathrm{M}_{1,1} & \cdots & \mathrm{M}_{d_\Sigma,1} & & \\ & & \ddots & \ddots & & \ddots & \\ & & & \mathrm{M}_{0,\Delta d} & \mathrm{M}_{1,\Delta d} & \cdots & \mathrm{M}_{d_\Sigma,\Delta d} \end{bmatrix} \in \mathbb{C}^{\frac{S}{2}(\Delta d+1)(\Delta d+2)\times\frac{1}{2}(d+1)(d+2)},$$

with $\Delta d := d - d_\Sigma$ and

$$\mathrm{M}_{i,j} := \begin{bmatrix} \boldsymbol{c}_{0i} & \boldsymbol{c}_{1i} & \cdots & \boldsymbol{c}_{(d_\Sigma-i)i} & & & \\ & \boldsymbol{c}_{0i} & \boldsymbol{c}_{1i} & \cdots & \boldsymbol{c}_{(d_\Sigma-i)i} & & \\ & & \ddots & \ddots & & \ddots & \\ & & & \boldsymbol{c}_{0i} & \boldsymbol{c}_{1i} & \cdots & \boldsymbol{c}_{(d_\Sigma-i)i} \end{bmatrix} \in \mathbb{C}^{S(\Delta d+1-j)\times(d+1-i-j)}.$$

$\mathrm{M}(d)$ is *relatively* LDR for bivariate systems

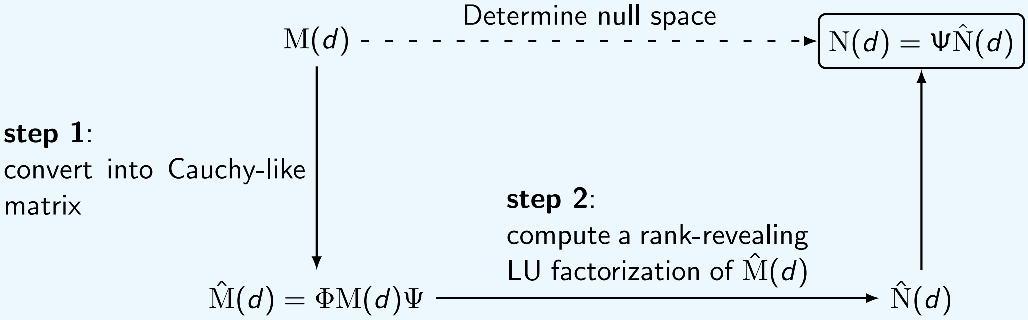
$$\mathscr{D}\left\{\mathrm{M}(d)\right\} = \begin{bmatrix} \mathrm{Z}_{\Delta d+1,1} \otimes \mathrm{I}_S & & \\ & \ddots & \\ & & \mathrm{Z}_{1,1} \otimes \mathrm{I}_S \end{bmatrix} \mathrm{M}(d) - \mathrm{M}(d) \begin{bmatrix} \mathrm{Z}_{d+1,\varphi_{d+1}} & & \\ & \ddots & \\ & & \mathrm{Z}_{1,\varphi_1} \end{bmatrix}$$

Dimensions of $\mathrm{M}(d) \in \mathbb{C}^{\frac{S}{2}(\Delta d+1)(\Delta d+2) \times \frac{1}{2}(d+1)(d+2)}$ grow *quadratically* w.r.t. d, but

$$\operatorname{rank} \mathscr{D}\left\{\mathrm{M}(d)\right\} \leq S(\Delta d + 1) = S\left(d + 1 - d_\Sigma\right).$$

grows only *linearly* with d.

An overview of the fast algorithm for computing the null space



Both steps can be done *fast*!

Rank-revealing LU (RRLU) factorization (Miranian and Gu 2003)

Apply permutation matrices Π_1, Π_2 such that $\hat{\mathrm{M}}_{22} - \hat{\mathrm{M}}_{21}\hat{\mathrm{M}}_{11}^{-1}\hat{\mathrm{M}}_{12} \approx 0$ in

$$\begin{aligned}
\Pi_1 \hat{\mathrm{M}}(d) \Pi_2 &= \begin{bmatrix} \mathrm{I} & \\ \hat{\mathrm{M}}_{21}\hat{\mathrm{M}}_{11}^{-1} & \mathrm{I} \end{bmatrix} \begin{bmatrix} \hat{\mathrm{M}}_{11} & \\ & \hat{\mathrm{M}}_{22} - \hat{\mathrm{M}}_{21}\hat{\mathrm{M}}_{11}^{-1}\hat{\mathrm{M}}_{12} \end{bmatrix} \begin{bmatrix} \mathrm{I} & \hat{\mathrm{M}}_{11}^{-1}\hat{\mathrm{M}}_{12} \\ & \mathrm{I} \end{bmatrix} \\
&\approx \begin{bmatrix} \hat{\mathrm{M}}_{11} \\ \hat{\mathrm{M}}_{21} \end{bmatrix} \begin{bmatrix} \mathrm{I} & \hat{\mathrm{M}}_{11}^{-1}\hat{\mathrm{M}}_{12} \end{bmatrix}
\end{aligned}$$

Expression for null space:

$$\hat{\mathrm{N}}(d) = \Pi_2 \begin{bmatrix} \tilde{\mathrm{N}} \\ \mathrm{I} \end{bmatrix}, \qquad \tilde{\mathrm{N}} := -\hat{\mathrm{M}}_{11}^{-1}\hat{\mathrm{M}}_{12}.$$

Extending the GKO algorithm to compute null spaces

- $\tilde{\mathrm{N}}$ is also Cauchy-like of the same LDR rank as $\hat{\mathrm{M}}(d)$
- Adapt GKO algorithm to directly determine generators for $\tilde{\mathrm{N}}$
- Use (approximate) complete pivoting strategy!
- Bivariate systems: QR too expensive $\rightarrow$ update strategy using Householder!
- Special provisions for stable calculation of $\tilde{\mathrm{N}}$.
- For more technical details (Govindarajan, Widdershoven, et al. 2024)

Flop count is reduced from $\mathcal{O}(d_\Sigma^6)$ to $\mathcal{O}(d_\Sigma^5)$ for bivariate systems

Assumptions:

- Number of roots $\sim d_\Sigma^2$ (i.e., Bezout bound),
- $2 \leq S \ll d_\Sigma$.

A quick complexity overview for each step

- Step 1: $\mathcal{O}(S \cdot d_\Sigma \cdot \Delta d \cdot d \log d)$
- Step 2: $\mathcal{O}((\operatorname{rank} \mathrm{M}(d)) \cdot S^2 d^3)$, $\operatorname{rank} \mathrm{M}(d) \sim d^2$

$$d \leq 2d_\Sigma - 2 \text{ to find a null space containing all system roots}$$
$$\downarrow$$
$$\mathcal{O}(d_\Sigma^5)$$

Stability experiments: error grows linearly with problem size

$$\epsilon := \frac{\|\mathrm{M}(d)\mathrm{Q}\|_2}{\|\mathrm{M}(d)\|_2}, \qquad \mathrm{Q} \text{ is an orthonormal basis for col } \mathrm{N}(d)$$

	d_Σ				
	2	4	8	16	32
SVD on $\mathrm{M}(d)$	2.23e-16	3.75e-16	5.70e-16	7.94e-16	9.51e-16
SVD on $\hat{\mathrm{M}}(d)$	2.57e-16	4.77e-16	7.54e-16	9.97e-16	1.15e-15
GECP on $\mathrm{M}(d)$	1.40e-16	3.11e-16	8.33e-16	1.02e-14	1.40e-13
GECP on $\hat{\mathrm{M}}(d)$	2.08e-16	4.65e-16	1.03e-15	9.73e-15	1.21e-13
GECP on $\mathscr{C}$	4.35e-16	1.51e-15	1.35e-14	1.72e-13	2.81e-12
GEAP on $\mathscr{C}$	4.21e-16	3.63e-15	3.88e-14	3.19e-13	4.48e-12

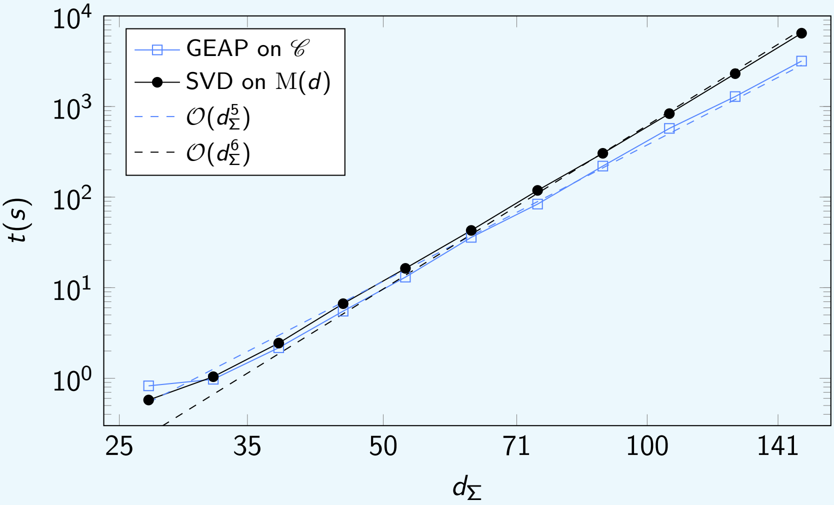
Median error ϵ over 100 runs for *square* systems with different methods and degrees.

Sources of error:

- switching to LU instead of an SVD
- working with the compact Cauchy representation $\mathscr{C}$
- switching to approximate pivoting $\leftarrow$ Surprisingly not so bad!

Our experiments indicate that the flop complexity is indeed $O(d_\Sigma^5)$

The measurements are the median of an adapted number of runs after warmup.



The fast algorithm extends to Polynomial systems in the Chebyshev basis

$$\Sigma : \quad \begin{cases} p_1(x,y) & := \displaystyle\sum_{i=0}^{d_\Sigma} \sum_{j=0}^{d_\Sigma - i} b_{1ij} T_i(x) T_j(y) = 0 \\ & \vdots \\ p_S(x,y) & := \displaystyle\sum_{i=0}^{d_\Sigma} \sum_{j=0}^{d_\Sigma - i} b_{Sij} T_i(x) T_j(y) = 0 \end{cases}$$

- *Toeplitz-plus-Hankel* structures instead of just Toeplitz.
- Apply same techniques but with a *modified* displacement equation.

One-variable case

Since $T_k(x)T_l(x) = \frac{1}{2}(T_{k+l}(x) + T_{|k-l|}(x))$, we have:

$$\mathrm{W}(d) = \mathrm{W}^{\mathrm{tpz}}(d) + \mathrm{W}^{\mathrm{hnk}}(d) \in \mathbb{C}^{S(\Delta d+1)\times d}$$

$$\mathrm{W}^{\mathrm{tpz}}(d) := \begin{bmatrix} \boldsymbol{b}_{0j} & \boldsymbol{b}_{1j} & \cdots & \boldsymbol{b}_{d_\Sigma j} & & & & & \\ \boldsymbol{b}_{1j} & \boldsymbol{b}_{0j} & \boldsymbol{b}_{1j} & \cdots & \boldsymbol{b}_{d_\Sigma j} & & & & \\ \vdots & \ddots & \ddots & \ddots & & \ddots & & & \\ \boldsymbol{b}_{d_\Sigma j} & \cdots & \boldsymbol{b}_{1j} & \boldsymbol{b}_{0j} & \boldsymbol{b}_{1j} & \cdots & \boldsymbol{b}_{d_\Sigma j} & & \\ & \ddots & & \ddots & \ddots & \ddots & & \ddots & \\ & & \boldsymbol{b}_{d_\Sigma j} & \cdots & \boldsymbol{b}_{1j} & \boldsymbol{b}_{0j} & \boldsymbol{b}_{1j} & \cdots & \boldsymbol{b}_{d_\Sigma j} \end{bmatrix}, \quad \mathrm{W}^{\mathrm{hnk}}(d) := \begin{bmatrix} \boldsymbol{b}_{0j} & \boldsymbol{b}_{1j} & \cdots & \boldsymbol{b}_{d_\Sigma j} & \\ \boldsymbol{b}_{1j} & & \iddots & & \\ \vdots & \iddots & & & \\ \boldsymbol{b}_{d_\Sigma j} & & & & \\ & & & & \\ & & & & \end{bmatrix}$$

$$\operatorname{rank}\left((\mathrm{Y}_{\Delta d,0} \otimes \mathrm{I}_S)\,\mathrm{W}(d) - \mathrm{W}(d)\mathrm{Y}_{d,1}\right) = 4S, \qquad \mathrm{Y}_{n,\delta} := \begin{bmatrix} \delta & 1 & & & \\ 1 & 0 & \ddots & & \\ & 1 & \ddots & 1 & \\ & & \ddots & 0 & 1 \\ & & & 1 & \delta \end{bmatrix}$$

Overview

The algorithm does not nicely extend for general N-variable systems

- The idea of "GKO+complete pivoting" naturally extends to N-variable system
- Current set-up however has diminishing returns: $\mathcal{O}(d^{3N})$ to $\mathcal{O}(d^{3N-1})$.
- Issue: we only exploit one direction of the multi-Toeplitz structure!

How to exploit multi-level Toeplitz structures? A big open question!

What we are searching for!

Given a Toeplitz-block toeplitz matrix $\mathrm{T} \in \mathbb{C}^{n^2 \times n^2}$, does there exist an $\mathrm{A}, \mathrm{B} \in \mathbb{C}^{n^2 \times n^2}$ such that

$$\operatorname{rank}(\mathrm{AT} - \mathrm{TB}) \in O(1)$$

and A, B have "fast" eigendecompositions?

If answer is no, we need a radically different approach...

Could composing displacement operators possibly resolve things?

$$\mathrm{T} = \left[\begin{array}{cccc|cccc|cccc|cccc}
t_0^0 & t_1^0 & t_2^0 & t_3^0 & t_0^1 & t_1^1 & t_2^1 & t_3^1 & t_0^2 & t_1^2 & t_2^2 & t_3^2 & t_0^3 & t_1^3 & t_2^3 & t_3^3 \\
t_{-1}^0 & t_0^0 & t_1^0 & t_2^0 & t_{-1}^1 & t_0^1 & t_1^1 & t_2^1 & t_{-1}^2 & t_0^2 & t_1^2 & t_2^2 & t_{-1}^3 & t_0^3 & t_1^3 & t_2^3 \\
t_{-2}^0 & t_{-1}^0 & t_0^0 & t_1^0 & t_{-2}^1 & t_{-1}^1 & t_0^1 & t_1^1 & t_{-2}^2 & t_{-1}^2 & t_0^2 & t_1^2 & t_{-2}^3 & t_{-1}^3 & t_0^3 & t_1^3 \\
t_{-3}^0 & t_{-2}^0 & t_{-1}^0 & t_0^0 & t_{-3}^1 & t_{-2}^1 & t_{-1}^1 & t_0^1 & t_{-3}^2 & t_{-2}^2 & t_{-1}^2 & t_0^2 & t_{-3}^3 & t_{-2}^3 & t_{-1}^3 & t_0^3 \\
\hline
t_0^{-1} & t_1^{-1} & t_2^{-1} & t_3^{-1} & t_0^0 & t_1^0 & t_2^0 & t_3^0 & t_0^1 & t_1^1 & t_2^1 & t_3^1 & t_0^2 & t_1^2 & t_2^2 & t_3^2 \\
t_{-1}^{-1} & t_0^{-1} & t_1^{-1} & t_2^{-1} & t_{-1}^0 & t_0^0 & t_1^0 & t_2^0 & t_{-1}^1 & t_0^1 & t_1^1 & t_2^1 & t_{-1}^2 & t_0^2 & t_1^2 & t_2^2 \\
t_{-2}^{-1} & t_{-1}^{-1} & t_0^{-1} & t_1^{-1} & t_{-2}^0 & t_{-1}^0 & t_0^0 & t_1^0 & t_{-2}^1 & t_{-1}^1 & t_0^1 & t_1^1 & t_{-2}^2 & t_{-1}^2 & t_0^2 & t_1^2 \\
t_{-3}^{-1} & t_{-2}^{-1} & t_{-1}^{-1} & t_0^{-1} & t_{-3}^0 & t_{-2}^0 & t_{-1}^0 & t_0^0 & t_{-3}^1 & t_{-2}^1 & t_{-1}^1 & t_0^1 & t_{-3}^2 & t_{-2}^2 & t_{-1}^2 & t_0^2 \\
\hline
t_0^{-2} & t_1^{-2} & t_2^{-2} & t_3^{-2} & t_0^{-1} & t_1^{-1} & t_2^{-1} & t_3^{-1} & t_0^0 & t_1^0 & t_2^0 & t_3^0 & t_0^1 & t_1^1 & t_2^1 & t_3^1 \\
t_{-1}^{-2} & t_0^{-2} & t_1^{-2} & t_2^{-2} & t_{-1}^{-1} & t_0^{-1} & t_1^{-1} & t_2^{-1} & t_{-1}^0 & t_0^0 & t_1^0 & t_2^0 & t_{-1}^1 & t_0^1 & t_1^1 & t_2^1 \\
t_{-2}^{-2} & t_{-1}^{-2} & t_0^{-2} & t_1^{-2} & t_{-2}^{-1} & t_{-1}^{-1} & t_0^{-1} & t_1^{-1} & t_{-2}^0 & t_{-1}^0 & t_0^0 & t_1^0 & t_{-2}^1 & t_{-1}^1 & t_0^1 & t_1^1 \\
t_{-3}^{-2} & t_{-2}^{-2} & t_{-1}^{-2} & t_0^{-2} & t_{-3}^{-1} & t_{-2}^{-1} & t_{-1}^{-1} & t_0^{-1} & t_{-3}^0 & t_{-2}^0 & t_{-1}^0 & t_0^0 & t_{-3}^1 & t_{-2}^1 & t_{-1}^1 & t_0^1 \\
\hline
t_0^{-3} & t_1^{-3} & t_2^{-3} & t_3^{-3} & t_0^{-2} & t_1^{-2} & t_2^{-2} & t_3^{-2} & t_0^{-1} & t_1^{-1} & t_2^{-1} & t_3^{-1} & t_0^0 & t_1^0 & t_2^0 & t_3^0 \\
t_{-1}^{-3} & t_0^{-3} & t_1^{-3} & t_2^{-3} & t_{-1}^{-2} & t_0^{-2} & t_1^{-2} & t_2^{-2} & t_{-1}^{-1} & t_0^{-1} & t_1^{-1} & t_2^{-1} & t_{-1}^0 & t_0^0 & t_1^0 & t_2^0 \\
t_{-2}^{-3} & t_{-1}^{-3} & t_0^{-3} & t_1^{-3} & t_{-2}^{-2} & t_{-1}^{-2} & t_0^{-2} & t_1^{-2} & t_{-2}^{-1} & t_{-1}^{-1} & t_0^{-1} & t_1^{-1} & t_{-2}^0 & t_{-1}^0 & t_0^0 & t_1^0 \\
t_{-3}^{-3} & t_{-2}^{-3} & t_{-1}^{-3} & t_0^{-3} & t_{-3}^{-2} & t_{-2}^{-2} & t_{-1}^{-2} & t_0^{-2} & t_{-3}^{-1} & t_{-2}^{-1} & t_{-1}^{-1} & t_0^{-1} & t_{-3}^0 & t_{-2}^0 & t_{-1}^0 & t_0^0
\end{array}\right]$$

$$\mathcal{D}_1 : \quad \mathrm{T} \mapsto (\mathrm{I}_n \otimes \mathrm{Z}_{n,\varphi_a})\,\mathrm{T} - \mathrm{T}\,(\mathrm{I}_n \otimes \mathrm{Z}_{n,\varphi_b})$$

$$\mathcal{D}_2 \circ \mathcal{D}_1 : \quad \mathcal{D}_1\{\mathrm{T}\} \mapsto (\mathrm{Z}_{n,\varphi_c} \otimes \mathrm{I}_n)\,\mathcal{D}_1\{\mathrm{T}\} - \mathcal{D}_1\{\mathrm{T}\}\,(\mathrm{Z}_{n,\varphi_d} \otimes \mathrm{I}_n)$$

Could composing displacement operators possibly resolve things?

$$\mathcal{D}_1\{\mathrm{T}\} = \left[\begin{array}{cccc|cccc|cccc|cccc} * & * & * & * & * & * & * & * & * & * & * & * & * & * & * & * \\ 0 & 0 & 0 & * & 0 & 0 & 0 & * & 0 & 0 & 0 & * & 0 & 0 & 0 & * \\ 0 & 0 & 0 & * & 0 & 0 & 0 & * & 0 & 0 & 0 & * & 0 & 0 & 0 & * \\ 0 & 0 & 0 & * & 0 & 0 & 0 & * & 0 & 0 & 0 & * & 0 & 0 & 0 & * \\ \hline * & * & * & * & * & * & * & * & * & * & * & * & * & * & * & * \\ 0 & 0 & 0 & * & 0 & 0 & 0 & * & 0 & 0 & 0 & * & 0 & 0 & 0 & * \\ 0 & 0 & 0 & * & 0 & 0 & 0 & * & 0 & 0 & 0 & * & 0 & 0 & 0 & * \\ 0 & 0 & 0 & * & 0 & 0 & 0 & * & 0 & 0 & 0 & * & 0 & 0 & 0 & * \\ \hline * & * & * & * & * & * & * & * & * & * & * & * & * & * & * & * \\ 0 & 0 & 0 & * & 0 & 0 & 0 & * & 0 & 0 & 0 & * & 0 & 0 & 0 & * \\ 0 & 0 & 0 & * & 0 & 0 & 0 & * & 0 & 0 & 0 & * & 0 & 0 & 0 & * \\ 0 & 0 & 0 & * & 0 & 0 & 0 & * & 0 & 0 & 0 & * & 0 & 0 & 0 & * \\ \hline * & * & * & * & * & * & * & * & * & * & * & * & * & * & * & * \\ 0 & 0 & 0 & * & 0 & 0 & 0 & * & 0 & 0 & 0 & * & 0 & 0 & 0 & * \\ 0 & 0 & 0 & * & 0 & 0 & 0 & * & 0 & 0 & 0 & * & 0 & 0 & 0 & * \\ 0 & 0 & 0 & * & 0 & 0 & 0 & * & 0 & 0 & 0 & * & 0 & 0 & 0 & * \end{array}\right] \sim \text{rank} - O(n) \text{ matrix}$$

$$\mathcal{D}_1 : \quad \mathrm{T} \mapsto (\mathrm{I}_n \otimes \mathrm{Z}_{n,\varphi_a})\,\mathrm{T} - \mathrm{T}\,(\mathrm{I}_n \otimes \mathrm{Z}_{n,\varphi_b})$$

$$\mathcal{D}_2 \circ \mathcal{D}_1 : \quad \mathcal{D}_1\{\mathrm{T}\} \mapsto (\mathrm{Z}_{n,\varphi_c} \otimes \mathrm{I}_n)\,\mathcal{D}_1\{\mathrm{T}\} - \mathcal{D}_1\{\mathrm{T}\}\,(\mathrm{Z}_{n,\varphi_d} \otimes \mathrm{I}_n)$$

Could composing displacement operators possibly resolve things?

$$\mathcal{D}_2 \circ \mathcal{D}_1\{\mathrm{T}\} = \left[\begin{array}{cccc|cccc|cccc|cccc}
* & * & * & * & * & * & * & * & * & * & * & * & * & * & * & * \\
0 & 0 & 0 & * & 0 & 0 & 0 & * & 0 & 0 & 0 & * & 0 & 0 & 0 & * \\
0 & 0 & 0 & * & 0 & 0 & 0 & * & 0 & 0 & 0 & * & 0 & 0 & 0 & * \\
0 & 0 & 0 & * & 0 & 0 & 0 & * & 0 & 0 & 0 & * & 0 & 0 & 0 & * \\
\hline
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & * & * & * & * \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & * \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & * \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & * \\
\hline
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & * & * & * & * \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & * \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & * \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & * \\
\hline
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & * & * & * & * \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & * \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & * \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & *
\end{array}\right] \sim \text{rank} - O(n) \text{ matrix}$$

$$\mathcal{D}_1 : \quad \mathrm{T} \mapsto (\mathrm{I}_n \otimes \mathrm{Z}_{n,\varphi_a})\,\mathrm{T} - \mathrm{T}\,(\mathrm{I}_n \otimes \mathrm{Z}_{n,\varphi_b})$$

$$\mathcal{D}_2 \circ \mathcal{D}_1 : \quad \mathcal{D}_1\{\mathrm{T}\} \mapsto (\mathrm{Z}_{n,\varphi_c} \otimes \mathrm{I}_n)\,\mathcal{D}_1\{\mathrm{T}\} - \mathcal{D}_1\{\mathrm{T}\}\,(\mathrm{Z}_{n,\varphi_d} \otimes \mathrm{I}_n)$$

Generally, composition displacement operators yields generalized Sylvester equations

If $\mathcal{D}_{\mathrm{A,B}} : \mathrm{T} \mapsto \mathrm{AT} - \mathrm{TB}$ and $\mathcal{D}_{\mathrm{C,D}} : \mathrm{T} \mapsto \mathrm{CT} - \mathrm{TD}$, then

$$\mathcal{D}_{\mathrm{C,D}} \circ \mathcal{D}_{\mathrm{A,B}} : \quad \mathrm{T} \mapsto \mathrm{CAT} - \mathrm{CTB} - \mathrm{ATD} + \mathrm{TBD}$$

Solving generalized Sylvester equations

Solve

$$\sum_{i=1}^{p} (\mathrm{A}_i \otimes \mathrm{B}_i)\,\mathrm{vec}(\mathrm{X}) = \mathrm{vec}\,(\mathrm{Y}) \quad \Longleftrightarrow \quad \sum_{i=1}^{p} \mathrm{A}_i \mathrm{X} \mathrm{B}_i = \mathrm{Y}$$

Diagonalization of the system:

- For $p = 2$, substitute gen. eigendecomp. of pencils $(\mathrm{A}_1, \mathrm{A}_2)$ and $(\mathrm{B}_1, \mathrm{B}_2)$:

$$\begin{aligned} \mathrm{A}_1\mathrm{X}\mathrm{B}_1 + \mathrm{A}_2\mathrm{X}\mathrm{B}_2 &= \mathrm{Y} \\ \mathrm{V}_A \Lambda_{A_1} \mathrm{Q}_A^{-1} \mathrm{X} \mathrm{V}_B \Lambda_{B_1} \mathrm{Q}_B^{-1} + \mathrm{V}_A \Lambda_{A_2} \mathrm{Q}_A^{-1} \mathrm{X} \mathrm{V}_B \Lambda_{B_2} \mathrm{Q}_B^{-1} &= \mathrm{Y} \\ \Lambda_{A_1} \mathrm{Q}_A^{-1} \mathrm{X} \mathrm{V}_B \Lambda_{B_1} + \Lambda_{A_2} \mathrm{Q}_A^{-1} \mathrm{X} \mathrm{V}_B \Lambda_{B_2} &= \mathrm{V}_A^{-1} \mathrm{Y} \mathrm{Q}_B \end{aligned}$$

- For $p > 2$ we require $\{\mathrm{A}_i\}_{i=1}^{p}$ and $\{\mathrm{B}_i\}_{i=1}^{p}$ to be both jointly diagonizable.

Could this possibly be true?

Given a Toeplitz-block-Toeplitz matrix $\mathrm{T} \in \mathbb{C}^{n^2 \times n^2}$, does there exist for some $p > 2$ $\{\mathrm{A}_i\}_{i=1}^{p}$ and $\{\mathrm{B}_i\}_{i=1}^{p}$ such that

$$\operatorname{rank}\left(\sum_{i=1}^{p} \mathrm{A}_i \mathrm{T} \mathrm{B}_i\right) \in O(1)$$

and $\{\mathrm{A}_i\}_{i=1}^{p}$, $\{\mathrm{B}_i\}_{i=1}^{p}$ are jointly diagonalizable by “nice” matrices?

If answer is no, we again need a radically different approach...

Overview

Motivation: Macaulay method for solving (overdetermined) polynomial systems

Low displacement rank (LDR) matrices and the GKO algorithm

A fast null space algorithm for Macaulay matrices exploiting LDR properties

Challenges of extending LDR theory in higher dimensions

Conclusions and additional remarks

Conclusions and additional remarks

The Macaulay null space problem:

- The GKO algorithm allows for a *faster* algorithm than standard approaches.
- But the returns *diminish* as N becomes larger.
- Other LDR methods (Mastronardi, Van Barel, et al. 2009) suffer from *same* issue.
- Further improvements possible with HSS representations of Cauchy-like matrices
- They sadly come with large constants and do *not* resolve dimensionality issues.
- Root cause: *inability* to fully exploit multi-Toeplitz structure in Gauss elimination
- Any breakthrough on this front will have *great impact*, e.g., faster TBT solvers!

Acknowledgements

- Lieven De Lathauwer
- Raphaël Widdershoven
- Shivkumar Chandrasekaran
- Nicola Mastronardi

References I

- Govindarajan, Nithin et al. (2024). "A Fast Algorithm for Computing Macaulay Null Spaces of Bivariate Polynomial Systems". In: SIAM Journal on Matrix Analysis and Applications 45.1, pp. 368–396.
- Gu, Ming (1998). "Stable and Efficient Algorithms for Structured Systems of Linear Equations". In: SIAM J. Matrix Anal. Appl. 19.2, pp. 279–306.
- Heinig, Georg (1995). "Inversion of Generalized Cauchy Matrices and other Classes of Structured Matrices". In: Linear Algebra for Signal Processing. Ed. by A. Bojanczyk and G. Cybenko. Vol. 69. The IMA Volumes in Mathematics and its Applications. New York, NY, USA: Springer.
- Mastronardi, Nicola, Marc Van Barel, and Raf Vandebril (2009). "On the computation of the null space of Toeplitz-like matrices". In: Electron. Trans. Numer. Anal. 33, pp. 151–162.
- Miranian, L and Ming Gu (2003). "Strong rank revealing LU factorizations". In: Linear Algebra Appl. 367, pp. 1–16.

References II

- Vanderstukken, Jeroen and Lieven De Lathauwer (2021). "Systems of Polynomial Equations, Higher-Order Rensor Decompositions and Multidimensional Harmonic Retrieval: A Unifying Framework. Part I: The Canonical Polyadic Decomposition". eng. In: SIAM J. Matrix Anal. Appl. 42.2, pp. 883–912.
- Vanderstukken, Jeroen et al. (2021). "Systems of Polynomial Equations, Higher-Order Tensor Decompositions, and Multidimensional Harmonic Retrieval: A Unifying Framework. Part II: The Block Term Decomposition". In: SIAM J. Matrix Anal. Appl. 42.2, pp. 913–953.