Displacement Structure of Matrices – An Introduction

Prof. Thomas Kailath

Hitachi America Professor of Engineering, Emeritus

Stanford University

SAM Workshop

A Coruña, Spain, June 2014

Some References

1. T. Kailath, S.Y. Kung, M. Morf, Displacement ranks of a matrix, Bull. Amer. Math. Soc., vol. 1, pp. 769-773, 1979.

2. T. Kailath, L. Ljung and M. Morf, Generalised Krein-Levinson equations for efficient calculation of Fredholm resolvents of nondisplacement kernels, in Topics in Functional Analysis, I.Gohberg and M. Kac, eds., pp. 168-184, Acad. Press, 1978.

3. T. Kailath and A H. Sayed, Displacement Structure: theory and applications, SIAM Review, pp. 297-386, 1995.

4. T. Kailath and A. H. Sayed, eds., Fast Reliable Algorithms for Matrices with Structure, SIAM Press, 1999.

5. T. Constantinescu, Schur Parameters, Factorization and Dilation problems, Birkhauser, 1999

6. V. Olshevsky, ed., Structured Matrices in Mathematics, Computer Science, and Engineering, American Mathematical Society, 2001.

7. D. A. Bini, G. Latouche and B. Meini, Numerical Methods for Structured Markov Chains, Oxford Press, 2005.

The Aim

The aim of displacement structure theory is to uncover and exploit implicit structure in the (dense) matrices encountered in applications in several fields:

Communications, Signal Processing, System Theory, Prediction and Filtering, Algebraic Coding Theory, Queuing Theory, Interpolation problems, Numerical Linear Algebra, Abstract Linear Algebra and Matrix Theory

Structured Matrices: Explicit

$$Toeplitz = [x_{i-j}] = \begin{bmatrix} x_0 & x_1 & x_2 \\ x_{-1} & x_0 & x_1 \\ x_{-2} & x_{-1} & x_0 \end{bmatrix}$$

$$Hankel = [x_{i+j}] = \begin{bmatrix} x_0 & x_1 & x_2 \\ x_1 & x_2 & x_3 \\ x_2 & x_3 & x_4 \end{bmatrix}$$

$$Vandermonde = \begin{bmatrix} 1 & 1 & 1 \\ x_1 & x_1^2 & x_1^3 \\ x_2 & x_2^2 & x_2^3 \end{bmatrix}$$

$$Cauchy = \left[\frac{1}{x_i - y_j}\right]$$

- Banded, Pick, Schur-Cohn, Routh-Hurwitz, Bezoutian, ...
- Controllability, observability, impulse response matrices for state-space descriptions, ...

Structured Matrices: Implicit

If A, B, C , D are all (explicitly) structured matrices, we would like the same to be true for various composites of these matrices that are often encountered in applications, e.g., the following that arise in least squares problems:

$$A^{-1},\ AB\ ,(A^{*}A)^{-1}A, D - CA^{-1}B$$

The composites unfortunately do not generally inherit the explicit structure of their constituent matrices. For many classes of matrices, this common structure turns out to be Displacement Structure, which can be exploited to design fast algorithms.

Application to Cell Phones

“It might interest you to know that part of my work here at Nokia involves the implementation of a pseudo-inverse of a **Toeplitz structured matrix** in hardware. We are using a version of **displacement structure based algorithms** with proprietary improvements for fast, parallel realization of the same. It has been a very interesting learning experience to consolidate the conflicting demands of precision, stability, complexity and real-time constraints into a working receiver structure."

- Anand.Kannan@nokia.com

An Application in Communications

A Reduced-Complexity Partial-Interference-Cancellation Group Decoder for STBCs

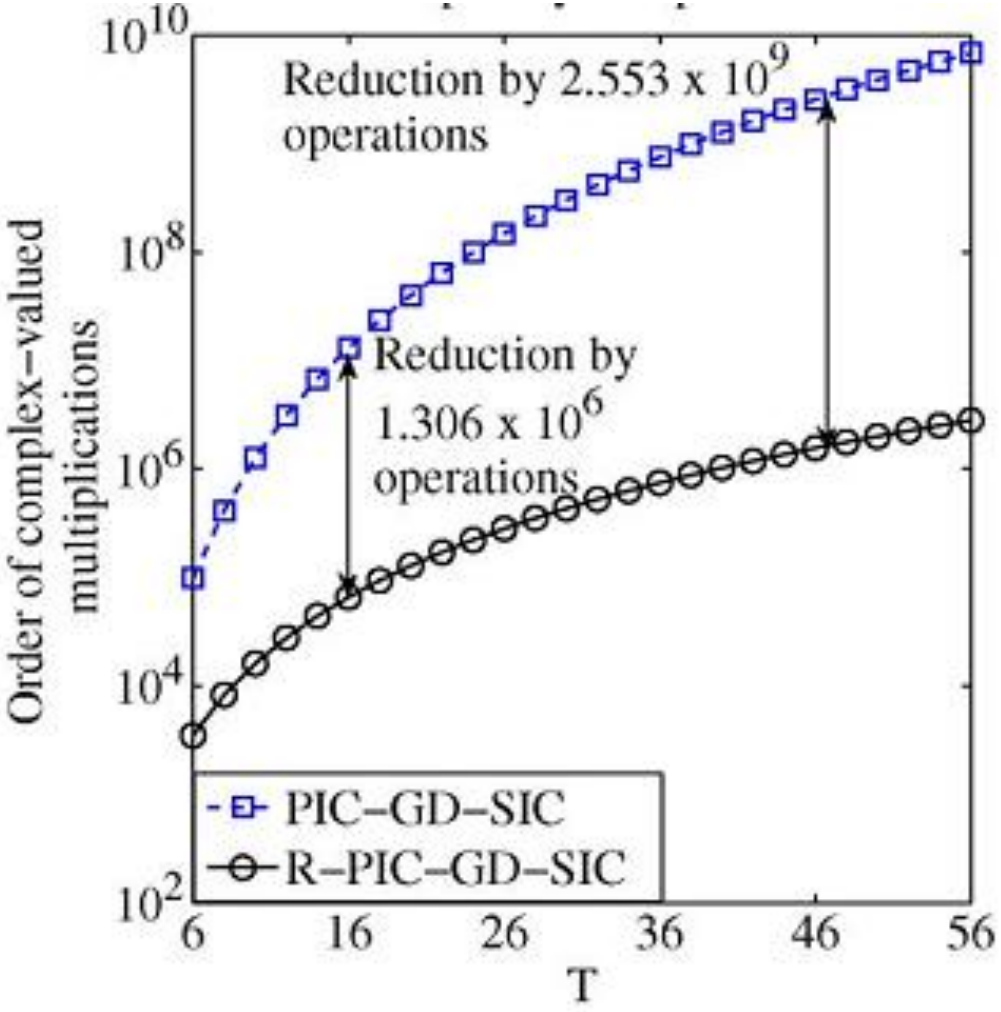
Rakshith Rajashekar, Student Member, IEEE , K.V.S.Hari , Senior Member, IEEE,and L.Hanzo, Fellow, IEEE

IEEE SIGNAL PROCESSING LETTERS, VOL. 20, NO. 10, OCTOBER 2013 92

Abstract—In this letter, we propose a reduced-complexity implementation of partial interference cancellation group decoder with successive interference cancellation (PIC-GD-SIC) by employing the theory of displacement structures.

The proposed algorithm exploits the block-Toeplitz structure of the effective matrix and chooses an ordering of the groups such that the zero-forcing matrices associated with the various groups are obtained through Schur recursions without any approximations. We show using an example that the proposed implementation offers a significantly reduced computational complexity compared to the direct approach without any loss in performance.

Complexity Comparison



Motivation – Toeplitz Matrices

Because a (symmetric) Toeplitz matrix

$$T = \begin{bmatrix} a & b & c \\ b & a & b \\ c & b & a \end{bmatrix}$$

can be specified by n rather than n^2 entries, we can expect that linear equations with T as coefficient matrix can be solved with $O(n^2)$ elementary computations. In fact this is true as first shown by N. Levinson (1949)

However, consider solving $Sx = b$, where S is not Toeplitz, but is known to be the inverse of some Toeplitz matrix. S will not be constant along diagonals, so the fast algorithms for T cannot be used. However, since S is not completely arbitrary, one can imagine that with sufficient effort, we can get $O(n^2)$ algorithms for S as well. In fact, this is true.

So the question arises : If it is not the constancy along diagonals that S & T share, what property do they have in common, that enables fast algorithms for both matrices?

The answer is that they have the same ***Displacement structure.***

Displacement Matrices

A preliminary definition introduces a displacement matrix.

$$\nabla R = R - ZRZ^*$$

where Z is the shift matrix with 1's on the first sub-diagonal and zeros everywhere else.

To see the reason for the name, first note that

$$\begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 \\ a & b & c \\ d & e & f \end{bmatrix}$$

while

$$\begin{bmatrix} 0 & 0 & 0 \\ a & b & c \\ d & e & f \end{bmatrix} \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & a & b \\ 0 & d & e \end{bmatrix}$$

Therefore,

$$R - ZRZ^* = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} - \begin{bmatrix} 0 & 0 & 0 \\ 0 & a & b \\ 0 & d & e \end{bmatrix}$$

which explains the name.

The **displacement rank** is the rank of the matrix ∇R

The above definition is to be restricted to the case where $\boldsymbol{R}$ is symmetric, or in the complex case, self adjoint. In this case, the displacement matrix will only have real eigenvalues and we can define the displacement inertia as $\{\boldsymbol{n}_+, \boldsymbol{n}_-\}$ the number of positive (non-zero) and negative eigenvalues. The displacement rank is then, $\boldsymbol{n}_+ + \boldsymbol{n}_-$.

The Toeplitz Case

When $R = T = \left[c_{|i-j|}\right]$, we see that the displacement $\nabla T = T - ZTZ^*$ has the form

$$\nabla T = \begin{bmatrix} c_0 & c_1 & c_2 \\ c_1 & c_0 & c_1 \\ c_2 & c_1 & c_0 \end{bmatrix} - \begin{bmatrix} 0 & 0 & 0 \\ 0 & c_0 & c_1 \\ 0 & c_1 & c_0 \end{bmatrix}$$

$$= \begin{bmatrix} c_0 & c_1 & c_2 \\ c_1 & 0 & 0 \\ c_2 & 0 & 0 \end{bmatrix}$$

which will have rank two (or less) no matter how large the rank of T is.

A simple exercise : Show that the displacement inertia of T is $\{1,1\}$

Some Composite Matrices

Let a Toeplitz matrix $T = \begin{bmatrix} A & B \\ C & D \end{bmatrix}$, where $\{A, B, C, D\}$ will also all be Toeplitz.

Then,

$$\text{Displacement rank}(T) \leq 2$$
$$\text{Displacement rank}(A) \leq 2$$
$$\text{Displacement rank}(A^{-1}) \leq 2$$
$$\text{Displacement rank}(AB) \leq 4$$
$$\text{Displacement rank}(D - CA^{-1}B) \leq 2$$

In general, it is true, and a very important fact that the Schur complements inherit the displacement structure of the original matrix

Theorem 1

If a matrix has displacement rank r, then standard $O(n^3)$ algorithms for a variety of associated matrix problems - solving linear equations, inversion, triangular and orthogonal factorization, Schur complementation, ... - can be replaced by $O(n^2 r)$ algorithms and with more effort, we can make them $O(rn\ logn)$.

Theorem 1:

$$\text{rank}(\nabla R) = \text{rank}(\nabla \tilde{I} R^{-1} \tilde{I})$$

where $\tilde{I}$ is the reverse identity matrix.

The proof follows from easily proved block matrix triangularization formulas and will be given later.

A Representation Theorem

Theorem 2:
Let

$$R - ZRZ^* = \sum_1^r x_i y_i^*$$

where, x_i and y_i are $n \times 1$ vectors and r is the displacement rank. Then,

$$R = \sum_1^r L(x_i)L^*(y_i)$$

where $L(x) =$ a lower-triangular Toeplitz matrix with first column x.

Corollary:
When

$$R = T = \left[c_{|i-j|}\right]$$

a symmetric Toeplitz matrix,

$$T = L(x)L^*(x) - \mathrm{L(y)L^*(y)}$$

where $x^* = [c_0 \quad c_1 \quad \dots \quad c_n]$, $y^* = [0 \quad c_0 \quad \dots \quad c_{n-1}]$.

Moreover, by Theorem 1, T^{-1} will have the same form, but with different x and y :

$$T^{-1} = L(a)L^*(a) - L(b)L^*(b).$$

With a little more effort, we can specify the {a,b}, thus obtaining a very simple proof of a famous result of Gohberg and Senencul.

An Application of Theorem 2

First check that

$$\boldsymbol{a}L(\boldsymbol{x}) = [a_1 \quad a_2 \quad a_3]\begin{bmatrix} x_1 & 0 & 0 \\ x_2 & x_1 & 0 \\ x_3 & x_2 & x_1 \end{bmatrix}$$

is the **convolution** of the vectors $\boldsymbol{a}$ and $\boldsymbol{x}$.

Since convolutions can be carried out by FFTs, this can be done with $O(nlog(n))$ computations.

In **signal detection** problems, and in fact in several statistical calculations, we often need to compute the quadratic form $\boldsymbol{a}^* T^{-1}\boldsymbol{a}$. If we ignore the displacement structure of T^{-1}, this will require $O(n^2)$ computations but using the displacement structure of T^{-1}, we see that we can do this with $O(nlog(n))$ computations, a very great reduction when n is large.

Proof of Theorem 2

Let

$$R_3 - ZR_3Z^* = \boldsymbol{x}\boldsymbol{y}^*.$$

We can solve this equation by successive substitution and the fact that Z^3 is identically zero (Z is a nilpotent matrix):

$$\begin{aligned} R_3 - ZR_3Z^* &= \boldsymbol{x}\boldsymbol{y}^* \\ ZR_3Z^* - Z^2R_3Z^{*^2} &= Z\boldsymbol{x}\boldsymbol{y}^*Z^* \\ Z^2R_3Z^{*^2} - Z^3R_3Z^{*^3} &= Z^2\boldsymbol{x}\boldsymbol{y}^*Z^{*^2} \end{aligned}$$

Adding these three equations and using $Z^3 \equiv 0$ gives:

$$\begin{aligned} R_3 &= \boldsymbol{x}\boldsymbol{y}^* + Z\boldsymbol{x}\boldsymbol{y}^*Z^* + Z^2\boldsymbol{x}\boldsymbol{y}^*Z^{*^2} \\ &= \begin{bmatrix} x_1 & 0 & 0 \\ x_2 & x_1 & 0 \\ x_3 & x_2 & x_1 \end{bmatrix} \begin{bmatrix} y_1 & y_2 & y_3 \\ 0 & y_1 & y_2 \\ 0 & 0 & y_1 \end{bmatrix} \\ &= L(\boldsymbol{x})L^*(\boldsymbol{y}) \end{aligned}$$

Proof of Theorem 1

First derive (or just verify), the useful block triangularization formulas

$$\begin{bmatrix} R & Z \\ Z^* & R^{-1} \end{bmatrix} = \begin{bmatrix} I & ZR \\ 0 & I \end{bmatrix} \begin{bmatrix} R - ZRZ^* & 0 \\ 0 & R^{-1} \end{bmatrix} \begin{bmatrix} I & ZR \\ 0 & I \end{bmatrix}^*$$
$$= \begin{bmatrix} I & 0 \\ Z^*R^{-1} & I \end{bmatrix} \begin{bmatrix} R & 0 \\ 0 & R^{-1} - Z^*R^{-1}Z \end{bmatrix} \begin{bmatrix} I & 0 \\ Z^*R^{-1} & I \end{bmatrix}^*$$

Since the block triangular matrices are non-singular, the triple products define congruent matrices (Now recall our assumption that $R = R^*$). Since congruence preserves inertia and therefore, also rank, we have that rank

$$rank(R - ZRZ^*) + rank(R^{-1}) = rank(R) + rank(R^{-1} - Z^*R^{-1}Z).$$

Since $rank(R) = rank(R^{-1})$, we have

$$rank(R - ZRZ^*) = rank(R^{-1} - Z^*R^{-1}Z) = rank(\tilde{I}R^{-1}\tilde{I} - Z\tilde{I}R^{-1}\tilde{I})$$

where $\tilde{I}$ is the reverse identity matrix and this completes the proof.

(Note that scalar Toeplitz matrices and their inverses are persymmetric, so $\tilde{I}T^{-1}\tilde{I} = T^{-1}$.)

Very important remark: Note that the form of Z does not enter into the proof. So, Z can be replaced by an arbitrary matrix, say F. However for the theory, we generally require that F be lower triangular.
This fact is very useful in studying the displacement structure of composite matrices.

We now proceed to describe fast algorithms for triangular matrix factorization, a key ingredient in the solution of problems in many different fields. We will need one more definition.

Generators of Structured Matrices

- We mention again that the displacement of an $n \times n$ Hermitian matrix R was originally defined in the late 1970s as

$$\nabla R = R - ZRZ^*$$

- When ∇R has low rank, then R will be said to be a structured matrix. Now note that whether r is low or not, we can non-uniquely factor ∇R as

$$\nabla R = R - ZRZ^* = GJG^*$$

where $J = \left(I_p \oplus -I_q\right)$ is a signature matrix that specifies the displacement inertia of R, and G is $n \times r$

- **The pair $\{\boldsymbol{G}, \boldsymbol{J}\}$ is called a generator of $\boldsymbol{R}$**. Note that G has rn elements as compared to n^2 in R, and usually $r \ll n$.

- This is the main point about structured matrices: use the generator matrix G, instead of R, in order to reduce algorithm complexity.

- Consider again the symmetric Toeplitz matrix,

$$\boxed{\mathbf{T} = \left[c_{|i-j|}\right]_{i,j=0}^{n-1}} \quad \text{with} \quad c_0 = 1.$$

- *It is easy to verify that*

$$\mathbf{T} - Z\mathbf{T}Z^* = \begin{bmatrix} 1 & 0 \\ c_1 & c_1 \\ c_2 & c_2 \\ \vdots & \vdots \\ c_{n-1} & c_{n-1} \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ c_1 & c_1 \\ c_2 & c_2 \\ \vdots & \vdots \\ c_{n-1} & c_{n-1} \end{bmatrix}^* .$$

- Hence, $\mathbf{T}$ has displacement rank 2 ($c_i \neq 0$), independent of n, with

$$J = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}, \quad G = \begin{bmatrix} 1 & c_1 & \dots & c_{n-1} \\ 0 & c_1 & \dots & c_{n-1} \end{bmatrix}^{\mathbf{T}} .$$

A major result concerning such structured matrices is that the successive Schur complements of R, denoted by R_i, inherit the same displacement structure:

$$\boxed{R_i - ZR_iZ^* = G_iJG_i^*}$$

Moreover,

- There is an efficient (Generalized Schur) procedure for computing the successive $\{G_i\}$ from the original G.
- This procedure also checks whether $R > 0$.

Generalized Schur Algorithm: $G_i \longrightarrow G_{i+1}$:

$$G_i = \begin{bmatrix} \times & \times & \times \\ \times & \times & \times \\ \times & \times & \times \\ \vdots & & \vdots \end{bmatrix} \overset{\Theta_i}{\rightarrow} \begin{bmatrix} \delta_i & 0 & 0 \\ \times' & \times' & \times' \\ \times' & \times' & \times' \\ \vdots & & \vdots \end{bmatrix} \overset{\text{shift}}{\rightarrow} \begin{bmatrix} 0 & 0 & 0 \\ \delta_i & \times' & \times' \\ \times' & \times' & \times' \\ \vdots & & \vdots \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ G_{i+1} \end{bmatrix}$$

where Θ_i such that $\Theta_i J \Theta_i^* = J$ is chosen to null out all but the first entry of the top row of G_i:

$$\boxed{\begin{bmatrix} 0 \\ G_{i+1} \end{bmatrix} = G_i \Theta_i \begin{bmatrix} 0 & 0 \\ 0 & I \end{bmatrix} + Z G_i \Theta_i \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}}$$

Generalized Schur Test: If these recursions can be performed for all i, then $R > 0$.

- The entries of L in the factorization $R = LDL^*$ are given by

$$\boxed{l_i = \tfrac{1}{\delta_i} G_i \Theta_i \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \quad d_i = |\delta_i|^2}$$

- Moreover, we can associate a first-order system, in state-space form, with each step of the algorithm by combining the expressions for $\{G_{i+1}, l_i\}$:

$$\begin{bmatrix} l_i & 0 \\ & G_{i+1} \end{bmatrix} = \begin{bmatrix} Zl_i & G_i \end{bmatrix} \begin{bmatrix} 0 & \frac{\delta_i}{d_i} \begin{bmatrix} 1 & 0 \end{bmatrix} \\ \Theta_i \begin{bmatrix} \delta_i \\ 0 \end{bmatrix} & \Theta_i \begin{bmatrix} 0 & 0 \\ 0 & I \end{bmatrix} \end{bmatrix}$$

$\{G_{i+1}, G_i\}$: regarded as output and input of the system.
$\{l_i, Zl_i\}$: regarded as future and current states.

- In many situations, however, it is necessary to use a more general definition of displacement structure.
- *Consider, for example, the extended block matrix* M

$$M = \begin{bmatrix} \mathbf{T} & I \\ I & \mathbf{0} \end{bmatrix}, \quad \textit{with a Toeplitz } \mathbf{T}.$$

- It is easy to verify that M has displacement rank 4 with respect to $M - Z_{2n} M Z_{2n}^*$, and displacement rank 2 with respect to $M - (Z_n \oplus Z_n) M (Z_n \oplus Z_n)^*$.
- *This, and many other examples, motivate the introduction of a general displacement defined by*

$$\boxed{\nabla R \equiv R - FRF^* = GJG^*}$$

where F *is lower triangular, with diagonal entries* $\{f_i\}_{i=0}^{n-1}$, *and* $J = (I_p \oplus -I_q)$. (We limit ourselves in this talk to positive-definite matrices R and $|f_i| < 1$).

- The recursions can also be extended to this more general structure. In this case, the successive Schur complements of R will also inherit similar displacement structure.

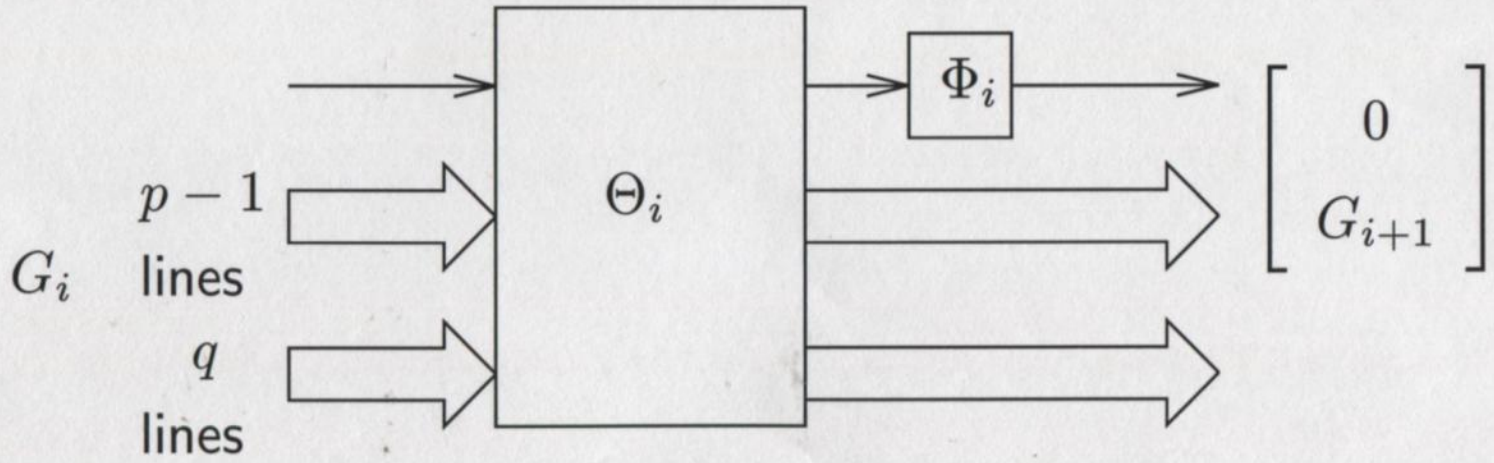
- That is, if R_i is the Schur complement of the leading $i \times i$ submatrix of R. Then R_i also exhibits displacement structure of the form

$$\boxed{R_i - F_i R_i F_i^* = G_i J G_i^*}$$

- Here, F_i is the submatrix obtained after deleting the first i rows and columns of F, and the generator G_i satisfies a recursive construction that we now describe.

- Using $\Phi_i = (F_i - f_i I)(I - f_i^* F_i)^{-1}$,

$$G_i = \begin{bmatrix} \times & \times & \times \\ \times & \times & \times \\ \times & \times & \times \\ \vdots & & \vdots \end{bmatrix} \overset{\Theta_i}{\to} \begin{bmatrix} \delta_i & 0 & 0 \\ \times' & \times' & \times' \\ \times' & \times' & \times' \\ \vdots & & \vdots \end{bmatrix} \overset{\Phi_i}{\to} \begin{bmatrix} 0 & 0 & 0 \\ \times'' & \times' & \times' \\ \times'' & \times' & \times' \\ \vdots & & \vdots \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ G_{i+1} \end{bmatrix}$$



- Each step of the algorithm now leads to

$$\boxed{\Theta_i(z) = \Theta_i \begin{bmatrix} \frac{z-f_i}{1-zf_i^*} & \mathbf{0} \\ \mathbf{0} & I \end{bmatrix}}$$

- As before, we can introduce the feedforward cascade:

$$\Theta(z) = \Theta_0(z)\Theta_1(z)\dots\Theta_n(z) = \begin{bmatrix} \Theta_{11}(z) & \Theta_{12}(z) \\ \Theta_{21}(z) & \Theta_{22}(z) \end{bmatrix}$$

and

$$\Sigma(z) = \begin{bmatrix} \Theta_{11}(z) - \Theta_{12}(z)\Theta_{22}^{-1}(z)\Theta_{21}(z) & -\Theta_{12}(z)\Theta_{22}^{-1}(z) \\ \Theta_{22}^{-1}(z)\Theta_{21}(z) & \Theta_{22}^{-1}(z) \end{bmatrix}$$

which maps Schur (contractive) functions to Schur (contractive) functions.

- This extension from Z to F allows us to nicely solve rational interpolation problems, as we now briefly demonstrate.

Displacement Structure Theory

Over the last 35 years, the initial ideas have been developed into an extensive theory of Displacement Structure. See for example, the survey paper, Kailath & Sayed SIAM review (1995).

Along the way several quite unexpected mathematical results were encountered and used, and some new ones developed. For example, a key reference is a 1917 paper by the famous mathematician, Issai Schur, on what would seem to be a purely mathematical topic: characterizing "power series that are bounded in the unit disc". Displacement Structure theory heavily builds on our generalizations of an algorithm found in that paper, which we have called Generalized Schur Algorithms.

While it can happen that fast algorithms lose some measure of numerical stability due to the accumulation of round-off errors, numerically stable variants of these algorithms can often be found.

Some References

1. T. Kailath, S.Y. Kung, M. Morf, Displacement ranks of a matrix, Bull. Amer. Math. Soc., vol. 1, pp. 769-773, 1979.

2. T. Kailath, L. Ljung and M. Morf, Generalised Krein-Levinson equations for efficient calculation of Fredholm resolvents of nondisplacement kernels, in Topics in Functional Analysis, I.Gohberg and M. Kac, eds., pp. 168-184, Acad. Press, 1978.

3. T. Kailath and A H. Sayed, Displacement Structure: theory and applications, SIAM Review, pp. 297-386, 1995.

4. T. Kailath and A. H. Sayed, eds., Fast Reliable Algorithms for Matrices with Structure, SIAM Press, 1999.

5. T. Constantinescu, Schur Parameters, Factorization and Dilation problems, Birkhauser, 1999

6. V. Olshevsky, ed., Structured Matrices in Mathematics, Computer Science, and Engineering, American Mathematical Society, 2001.

7. D. A. Bini, G. Latouche and B. Meini, Numerical Methods for Structured Markov Chains, Oxford Press, 2005.

Thank you for your attention

The following two results play a significant role in the developments to follow:

- If A is invertible and $\Delta = D - CA^{-1}B$ then

$$\boxed{\begin{bmatrix} A & B \\ C & D \end{bmatrix} - \begin{bmatrix} A \\ C \end{bmatrix} A^{-1} \begin{bmatrix} A & B \end{bmatrix} = \begin{bmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \Delta \end{bmatrix}.}$$

  The matrix Δ is called the Schur complement of A. That is, we can induce a zero column and row via a rank-one modification, or Schur complementation (or Gaussian elimination).

- Given two $n \times m$ $(n \leq m)$ matrices A and B such that $\boxed{AJA^* = BJB^*}$ is of full rank, for some $m \times m$ signature matrix J, then there exists an $m \times m$ $J-$unitary rotation Θ $(\Theta J \Theta^* = J)$ such that $\boxed{A = B\Theta}$

The key factorization algorithm in our analysis is based on combining Gaussian elimination with the notion of displacement structure.

Gaussian Elimination

The (misnamed) Gaussian elimination technique solves

$$Ax = b$$

by using elementary operations to bring A to upper triangular form.

Von Neumann pointed out that this was equivalent to factoring A as LDU, with L lower triangular, U upper triangular, both with unit diagonal, and D diagonal.

$$(Ax = b \; gives \; Ux = D^{-1}L^{-1}b)$$

Such factorizations play an important role in numerous problems. This seems trivial, but see the next slides.

Wiener-Hopf Equation

The story begins in 1931, when the astronomer Eberhard Hopf paid a visit to the summer home of the already famous mathematician Norbert Wiener of MIT. As was his wont, Wiener enquired of his guest what the most outstanding problem was in his field. The response was that no solution method was known for an integral equation put forward by the astronomers Milne and Schwarzschild to characterize certain problems of radiative transfer in an atmosphere of infinite height:

$$\int_{-\infty}^{\infty} w(\tau)K(t-\tau)d\tau = g(t), \qquad \boldsymbol{t \geq 0}$$

where $K(\cdot)$ and $g(\cdot)$ are known and $w(\cdot)$ is to be found.

The Wiener-Hopf Technique

At breakfast the next morning Wiener presented a solution! It was not quite correct, but the mistakes were easily fixed, and the paper on it was published by Wiener and Hopf in 1931.

So unexpectedly brilliant was the solution that the equation itself came to be known as the Wiener-Hopf equation and the solution method as the Wiener-Hopf technique.

The critical idea in the Wiener-Hopf technique requires something called spectral factorization, which is an infinite-dimensional version of writing a matrix as a product of upper- and lower-triangular matrices.

Solving Linear Equations

Motivation:

Consider the problem of solving the system of linear equations

$$Ax = b$$

where A is a known $n \times n$ matrix, b is a known n-vector, and x is an n-vector of unknowns.

There are many libraries of computational algorithms for this problem. However, for general matrices A the computational effort is $O(n^3)$, which is prohibitive for the large n found in many interesting problems. How do we get around this?

The answer is that we often make simplifying modeling assumptions on the underlying physical situations, idealizations such as homogeneity, isotropy, time-invariance, infinite duration, finite bandwidth, etc.. These lead to special structures for the matrix A, which can allow the computational burden to be reduced to $O(n^2)$ or $O(n\ logn)$ or even $O(n)$.

Fast Algorithms for Linear Equations

The standard way of solving linear equations $A\boldsymbol{x} = b$ is by the method mistakenly called Gaussian elimination (probably first used by the Chinese in xxx and next apparently by Newton in xxx).

The processes to use elementary operations on the rows of the matrix A to reduce it to upper triangular form from which the components of the solution can easily be found by successive substitution.

It was pointed out by Von Neumann and repeated in Turing 1949 with acknowledgement that this procedure amounts to triangular factorization of the matrix A. One form is

$$A = LDU$$

where $L(U)$ is a lower(upper) triangular matrix with unit diagonal and D is diagonal.

Therefore fast algorithms for solving linear equations are essentially equivalent to fast algorithms for triangular factorization of matrices. We shall show how displacement structure can be exploited to obtain such fast algorithms.

Displacement Generators

We start with the definition of the displacement

$$\nabla R = R - ZRZ^*$$

Then $r = rank(\nabla R)$ is called the displacement rank. When $r \ll n$ (the size of the matrix R), we say that the matrix is structured.

Going forward, we shall, for a considerable period of time, assume that R is hermitian. Then, we can represent the displacement inertia of R by the matrix

$$J = \begin{bmatrix} I_p & 0 \\ 0 & -I_q \end{bmatrix}$$

where $p(q)$ is the number of non-zero positive(negative) eigenvalues of ∇R. Then we can write (non-uniquely)

$$R - ZRZ^* = GJG^*$$

where the $n \times r$ matrix G is called a generator matrix and the triple $\{\mathrm{Z}, G, J\}$ is called a Generator of R. Note that G will have rn elements as compared to n^2 in R.

This is the main point about structured matrices work with the generator in place of the matrix R in order to reduce the operation count of associated (esp. factorization) algorithms from $O(n^3)$ to $O(rn^2)$ The reason G is not unique is because we can always replace it by $G\Theta$, where Θ is J-unitary, $J = \Theta J \Theta^*$

Significance: If a matrix has displacement rank , then standard $O(n^3)$ algorithms for a variety of associated matrix problems - solving linear equations, inversion, triangular and orthogonal factorization, Schur complementation, ... - can be replaced by $O(n^2\alpha)$ algorithms where α is the displacement rank.

Along the way several quite unexpected mathematical results were encountered and used, and some new ones developed. For example, a key reference is a 1917 paper by the famous mathematician, Issai Schur, on what would seem to be a purely mathematical topic: characterizing \power series that are bounded in the unit disc". Displacement Structure theory heavily builds on our generalizations of an algorithm found in that paper, which we have called Generalized Schur Algorithms.

While it can happen that fast algorithms lose some measure of numerical stability due to the accumulation of round-off errors, numerically stable variants of these algorithms can often be found

Schur Complements

Given $\mathrm{M} = \begin{bmatrix} A & B \\ C & D \end{bmatrix}$,

The Schur complement

of A in M is $D - CA^{-1}B$

of D in M is $A - BD^{-1}C$

Schur complements have many nice properties (see a famous review paper by Cottle.)

They play a major role in triangulizing matrices, an operation of great importance in Matrix theory.

Block Matrix Triangularization

$$\begin{bmatrix} I & 0 \\ X & I \end{bmatrix} \begin{bmatrix} A & B \\ C & D \end{bmatrix} = \begin{bmatrix} A & B \\ XA + C & XB + D \end{bmatrix}$$
$$= \begin{bmatrix} A & B \\ 0 & D - CA^{-1}B \end{bmatrix}$$

by choosing $X = -CA^{-1}$. Similarly, we can see that

$$\begin{bmatrix} A & B \\ 0 & D - CA^{-1}B \end{bmatrix} \begin{bmatrix} I & -A^{-1}B \\ 0 & I \end{bmatrix} = \begin{bmatrix} A & 0 \\ 0 & D - CA^{-1}B \end{bmatrix}$$

Using $\begin{bmatrix} I & 0 \\ X & I \end{bmatrix}^{-1} = \begin{bmatrix} I & 0 \\ -X & 0 \end{bmatrix}, \begin{bmatrix} I & Y \\ 0 & I \end{bmatrix} = \begin{bmatrix} I & -Y \\ 0 & I \end{bmatrix}$

will give us one of the identities on the previous page