A Lanczos-like method for the time-ordered exponential

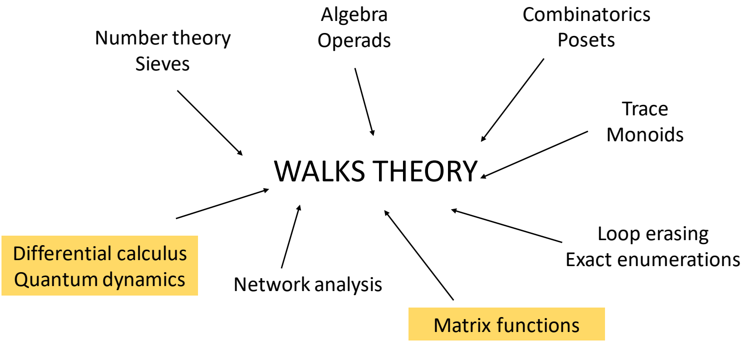
Pierre-Louis Giscard[1] Stefano Pozza[2]

[1]Université du Littoral Côte d'Opale, Calais

[2]Charles University, Prague

WACA workshop, Calais
May 27, 2021

P.-L.Giscard's program



Outline

- The $*$-product (a simple scalar example)
- The Path-sum method for the solution of a system of ODEs
- Lanczos as a way to transform a graph into a path.
- $*$-Lanczos + Path-sum: a new expression for the solution of a system of linear ODEs (time ordered exponential)
- Conclusion

Projects:

- PRIMUS research poject *A Lanczos-like Method for the Time-Ordered Exponential*, 2021-2023. **www.starlanczos.cz**
- ANR research project *MAGICA – MAGnetic resonance techniques and Innovative Combinatorial Algebra*, P.I. C. Bonhomme, France, 2021-2025.

Scalar ODE

Consider the "simple" problem:

$$\frac{d}{dt'} u(t') = a(t') u(t'), \quad u(t) = 1,$$

a smooth and bounded.
The (obvious) solution is given by

$$u(t') = \exp\left(\int_t^{t'} a(\tau)\, \mathrm{d}\tau\right).$$

We are going to show and alternative expression for the solution, which will be useful for large system of ODEs.

A new approach: the $*$-product

Let $\widetilde{d}(t',t), \widetilde{d}_i(t',t)$ smooth functions in $t', t \in I$. Consider the class $\mathrm{D}(I)$ of the distributions

$$d(t',t) = \widetilde{d}(t',t)\Theta(t'-t) + \sum_{i=0}^{N} \widetilde{d}_i(t',t)\delta^{(i)}(t'-t),$$

with $\widetilde{d}(t',t)$ smooth functions in t', t, Θ the Heaviside function. For $f_1, f_2 \in \mathrm{D}(I)$ we define the convolution-like $*$-product as

$$(f_1 * f_2)(t',t) := \int_{-\infty}^{\infty} f_1(t',\tau) f_2(\tau,t)\, \mathrm{d}\tau,$$

with identity $1_* := \delta(t'-t)$; c.f., [Volterra, Pérès, '28], [Schwartz, '78].

Remark: Volterra and Pérès did not have the distribution theory by Schwartz at their time!

Basics of the $*$-product

- $1_* := \delta(t' - t)$ is the identity;
- The dirac derivatives work nicely:

$$\delta^{(i)}(t'-t)*\delta^{(j)}(t'-t) = \delta^{(j)}(t'-t)*\delta^{(i)}(t'-t) = \delta^{(i+j)}(t'-t)$$

- The $*$-inverse of δ' is the Heaviside function, we use the convention:

$$\Theta(t'-t) = \begin{cases} 1, & t' \geq t \\ 0, & t' < t \end{cases};$$

i.e.,

$$\Theta(t'-t) * \delta'(t'-t) = \delta'(t'-t) * \Theta(t'-t) = \delta(t'-t).$$

Basics of the $*$-product

Consider the subclass $\mathrm{Sm}_\Theta(I)$ of $\mathrm{D}(I)$:

$$f(t',t) = \tilde{f}(t',t)\Theta(t'-t).$$

For $f_1,\ f_2 \in \mathrm{Sm}_\Theta(I)$, the $*$-product between f_1, f_2 simplifies to

$$\big(f_2 * f_1\big)(t',t) = \Theta(t'-t)\int_t^{t'} \tilde{f}_2(t',\tau)\tilde{f}_1(\tau,t)\,\mathrm{d}\tau.$$

- $\mathrm{D}(I)$ is closed under $*$ multiplication;
- $f \in \mathrm{Sm}_\Theta(I)$ is $*$-invertible in $\mathrm{D}(I)$ if $f(t',t') \neq 0$ for $t' \in I$ (restrictive) and $\widetilde{f}$ is separable.

The kth $*$-power f^{*k} as the k $*$-products $f * f * \cdots * f$, $(f^{*0} = \delta)$.

$$\Theta^{*k} = \frac{(t'-t)^{k-1}}{(k-1)!}\Theta(t'-t); \quad \left(\delta^{(j)}\right)^{*k} = \delta^{(kj)}(t'-t).$$

ODE solution by $*$-product

$$\frac{d}{dt'}u(t') = \widetilde{a}(t')u(t'), \quad u(t) = 1, \quad t' \geq t.$$

Then the solution can be given as ([Giscard & al., 2015])

$$u(t') = \Theta(t', t) * R_*(a)(t', t),$$

with $a(t', t) = \widetilde{a}(t')\Theta(t' - t)$, and R_* the $*$-resolvent

$$R_*(a)(t', t) := (1_* - a)^{*-1}(t', t) = \sum_{k=0}^{\infty} a^{*k}(t', t); \quad \text{Neumann series}$$

Note that the series converges when a is bounded.

Example

Consider the simple case $\widetilde{a}(t') = 1$, $a(t', t) = \Theta(t' - t)$. Then

$$R_*(a) = \sum_{k=0}^{\infty} \Theta^{*k}.$$

Hence

$$u(t') = \Theta * R_*(a) = \sum_{k=0}^{\infty} \Theta^{*(k+1)} = \sum_{k=0}^{\infty} \frac{(t' - t)^k}{(k)!} \Theta(t' - t).$$

As expected, the solution is

$$u(t') = \exp(t' - t)\Theta(t' - t).$$

Systems of ODEs

Let $t' \geq t \in I \subseteq \mathbb{R}$, $\mathrm{A}(t')$ a time dependent matrix. The time-ordered exponential is the unique solution $\mathrm{U}(t',t)$ of

$$\widetilde{\mathrm{A}}(t')\mathrm{U}(t',t) = \frac{d}{dt'}\mathrm{U}(t',t), \quad \mathrm{U}(t,t) = \mathrm{I}_N.$$

If $\widetilde{\mathrm{A}}(\tau_1)\widetilde{\mathrm{A}}(\tau_2) = \widetilde{\mathrm{A}}(\tau_2)\widetilde{\mathrm{A}}(\tau_1)$ for all $\tau_1, \tau_2 \in I$, then

$$\mathrm{U}(t',t) = \exp\left(\int_t^{t'} \widetilde{\mathrm{A}}(\tau)\, d\tau\right).$$

U has generally no explicit form. Expression by ([Dyson, 1948])

$$\mathrm{U}(t',t) = \mathcal{T}\exp\left(\int_t^{t'} \widetilde{\mathrm{A}}(\tau)\, d\tau\right).$$

with $\mathcal{T}$ the time-ordering operator.

Time-ordered exponential

The time-ordering expression is more a notation as the action of the time-ordering operator is very difficult to evaluate.

- Applications: System dynamics (quantum dynamics); e.g., [Blanes & al., 2009], [Giscard, Bonhomme, 2020]. Differential Riccati matrix equations (control theory, filter design); e.g., [Abou-Kandil et al., 2003].

We focus on expressions for U in terms of scalar integrals and differential equations. We will not talk about numerical methods.

- Classical approaches Perturbative methods (Floquet-based and Magnus series techniques), often prohibitively involved, e.g., [Blanes & al., 2009];
- Path-sum approach: The expression has a finite number of scalar integro-differential equations, but its complexity can be too large; [Giscard & al., 2015].

Solution by $*$-product

Nevertheless, the $*$-resolvent expression for the solution remains:

$$U(t', t) = \Theta(t' - t) * R_*(A),$$

with

$$R_*(A)(t', t) = \sum_{k=0}^{\infty} \left(\widetilde{A}(t')\Theta(t' - t)\right)^{*k},$$

where the $*$-product here is extended in the matrix-product sense.

The probelm is: How do we compute $R_*(A)$?

A possible solution: Path-sum method [Giscard & al., 2015].

Example: the 2×2 case

Consider the time-dependent matrix

$$A(t', t) = \begin{bmatrix} \widetilde{a}(t') & \widetilde{b}(t') \\ \widetilde{c}(t') & \widetilde{d}(t') \end{bmatrix} \Theta(t', t).$$

In this simple case the Path-sum method gives:

$$p = b * R_*(d) * c,$$

$$R_*(A)_{1,1} = R_*(p + a),$$

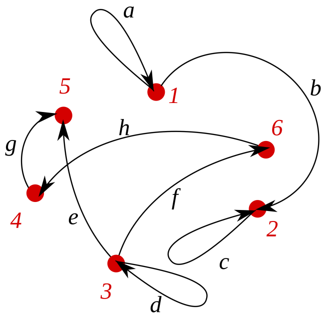
from which we get the $(1, 1)$ entry of the solution

$$\begin{aligned} U_{1,1}(t', t) &= \Theta(t' - t) * R_*(A)_{1,1}(t', t) \\ &= \Theta * (1_* - a - b * (1_* - d)^{*-1} * c)^{*-1}. \end{aligned}$$

Note the continued fraction structure of the solution.

Path-sum: main ideas

In general, Path-sum looks at A as an adjacency matrix of a weighted, generally undirected, time dependent graph.



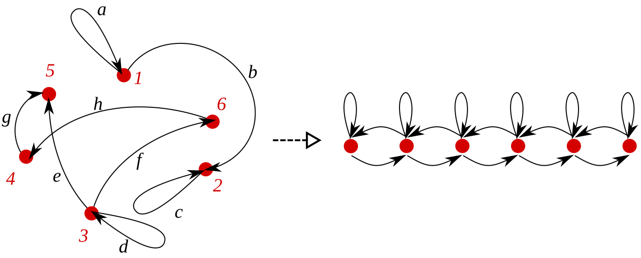
$$\widetilde{A}(t') = \begin{bmatrix} a & b & & & & \\ & c & & & & \\ & & d & & e & f \\ & & & & g & \\ & & & & & \\ & & & h & & \end{bmatrix}(t')$$

Path-sum requires one to find all the simple cycles and simple paths of the graph. Then it expresses each entry of U as a branched continued fraction of finite depth and breadth.

The problem with Path-sum and a possible solution

When A is large and has no exploitable structure, finding all the simple cycles and simple paths is too expensive (#P-complete).

Idea: Tridiagonalization



$$\widetilde{A}(t') = \begin{bmatrix} a & b & & & & \\ & c & & & & \\ & & d & & e & f \\ & & & & g & \\ & & & & & \\ & & & h & & \end{bmatrix}(t') \quad \rightarrow \quad \widetilde{T}(t', t) = \begin{bmatrix} \star & \star & & & & \\ \star & \star & \star & & & \\ & \star & \star & \star & & \\ & & \star & \star & \star & \\ & & & \star & \star & \star \\ & & & & \star & \star \end{bmatrix}(t', t)$$

ODEs with constant coefficients

Tridiagonalization for the simpler case where A is constant.

Let $\mathrm{A} \in \mathbb{C}^{N\times N}$ be a square matrix and $t' \geq t \in I \subseteq \mathbb{R}$. The solution $\mathrm{U}(t',t) \in \mathbb{C}^{N\times N}$ of the system of ODEs

$$\mathrm{A}\,\mathrm{U}(t',t) = \frac{d}{dt'}\mathrm{U}(t',t), \quad \mathrm{U}(t,t) = \mathrm{I_N},$$

can be expressed as

$$\mathrm{U}(t',t) = \exp\left(\mathrm{A}\cdot(t'-t)\right).$$

(Symmetric) Lanczos method

Given a symmetric matrix $A \in \mathbb{R}^{N \times N}$ and a vector $\mathrm{v} \neq 0$, Lanczos produces the orthogonal matrix $U_m = [\mathrm{u}_1, \dots, \mathrm{u}_m]$, basis of the (polynomial) Krylov subspace

$$\mathcal{K}_m(A, \mathrm{v}) := \mathrm{span}\left\{\mathrm{v}, A\,\mathrm{v}, \dots, A^{m-1}\,\mathrm{v}\right\}.$$

Starting with $\mathrm{u}_1 = \mathrm{v}/\|\mathrm{v}\|$, Lanczos is a Gram-Schmidt orthogonalization process defined by the recurrences

$$\begin{aligned} t_{j+1,j}\mathrm{u}_{j+1} &= A\mathrm{u}_j - \sum_{i=1}^{j} t_{i,j}\mathrm{u}_i \\ &= Au_j - t_{j,j}u_j - t_{j-1,j}u_{j-1}, \quad j = 1, \dots, m. \end{aligned}$$

$$t_{i,j} = \mathrm{u}_i^* A\mathrm{u}_j, \quad t_{j+1,j} = \|\mathrm{u}_{j+1}\|.$$

(Symmetric) Lanczos method

The recurrences have the matrix form:

$$AU_m = U_m T_m + t_{m+1,m} \mathrm{u}_{m+1} \mathrm{e}_m^T,$$

with T_m the $m \times m$ tridiagonal matrix with entries $t_{i,j}$ (e_m the mth vector of the canonical basis). By orthogonality we get

$$T_m = U_m^* A\, U_m.$$

The matrix T_m plays two roles in the algorithm:

- It represents the orthogonalization process (coefficients $t_{i,j}$);
- It represents the action of A in the Krylov subspace $\mathcal{K}_m(A, \mathrm{v})$, i.e.,

$$U_m T_m U_m^* = U_m U_m^* A\, U_m U_m^*.$$

Non-Hermitian Lanczos algorithm

Assume for simplicty $t = 0$ and let v, w be vectors so that $\mathrm{w}^H \mathrm{v} \neq 0$. We aim to approximate

$$\mathrm{w}^H \mathrm{U}(t', 0)\, \mathrm{v} = \mathrm{w}^H \exp(\mathrm{A} t')\, \mathrm{v}.$$

Consider the Krylov subspaces

$$\mathrm{span}\{\mathrm{v}, \mathrm{A}\,\mathrm{v}, \dots, \mathrm{A}^{n-1}\,\mathrm{v}\}, \quad \mathrm{span}\{\mathrm{w}, \mathrm{A}^H \mathrm{w}, \dots, (\mathrm{A}^H)^{n-1} \mathrm{w}\}.$$

Assuming the algorithm does not breakdown, non-Hermitian Lanczos computes the matrices

$$\mathrm{V}_n = [\mathrm{v}_0, \dots, \mathrm{v}_{n-1}], \quad \mathrm{W}_n = [\mathrm{w}_0, \dots, \mathrm{w}_{n-1}]$$

bases of the Krylov subspaces so that $\mathrm{W}_n^H \mathrm{V}_n = \mathrm{I}_n$.

Remark: Arnoldi does not produce a tridiagonal matrix.

Non-Hermitian Lanczos algorithm

The $n \times n$ tridiagonal matrix defined as

$$J_n = W_n^H A V_n,$$

satisfies the matching moment property[1]

$$w^H A^k v = e_1^H (J_n)^k e_1, \quad k = 0, \dots, 2n-1.$$

We get the approximation (model reduction)

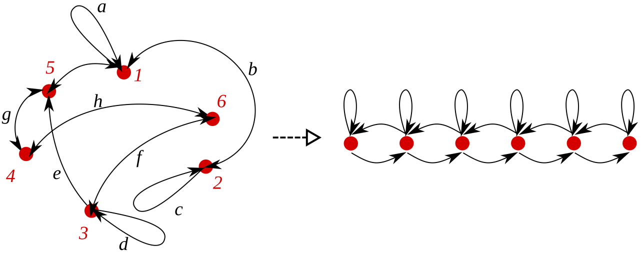
$$w^H \exp(A\, t')\, v \approx e_1^H \exp(J_n\, t')\, e_1;$$

e.g., [Golub, Meurant, 2010].
For $n = N$, we get the tridiagonalization of A, [Parlett, 1992],

$$A = V_N J_N W_N^H \implies \exp(A\, t') = V_N \exp(J_n\, t')\, W_N^H.$$

[1] Gragg, Lindquist (1983); Cybenko (1987); Freund, Hochbruck (1993); Strakoš (2009); P., Pranić, Strakoš (2017).

Graph interpretation of Lanczos

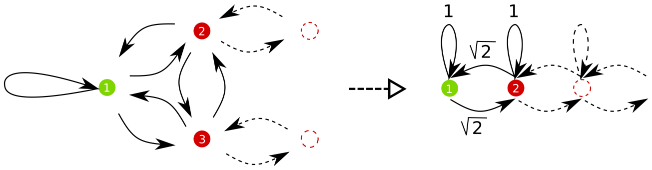


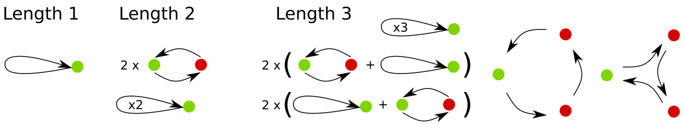
G P

- If $w = v = e_j$, the moment $e_j^T A^k e_j$ is the weighted # of closed walks of length k from j to j in the original graph G; e.g., [Estrada, Rodriguez-Velazquez, 2005].
- The matching moment property says that the weighted # of such closed walks is the same in G and P.

(We are working on this with Francesca Arrigo)

Simple example





$$A = \begin{bmatrix} 1 & 1 & 1 & \cdots \\ 1 & 0 & 1 & \cdots \\ 1 & 1 & 0 & \cdots \\ \vdots & \vdots & \vdots & \end{bmatrix} \quad \rightarrow \quad T_n = \begin{bmatrix} 1 & \sqrt{2} & \\ \sqrt{(2)} & 1 & \ddots \\ & \ddots & \ddots \end{bmatrix}$$

A new approach: the $*$-Lanczos algorithm

Let $\mathrm{A}(t')$ an $N \times N$ time-dependent matrix smooth in $t' \in I$, and let v, w be time-independent vectors ($\mathrm{w}^H \mathrm{v} \neq 0$). Assuming no breakdown, the nth step of the $*$-Lanczos method gives

$$V_n(t', t), W_n(t', t) \in \mathrm{D}(I)^{N \times n}, \quad T_n(t', t) \in \mathrm{D}(I)^{n \times n},$$

so that $W_n(t', t)^H * V_n(t', t) = 1_* \mathrm{I}_n$. T_n is tridiagonal and so that

$$\mathrm{T}_n(t', t) = \mathrm{W}_n^H(t', t) * \mathrm{A}(t')\Theta(t' - t) * \mathrm{V}_n(t', t).$$

It also satisfies the $*$-matching moment property

$$\mathrm{w}^H \left(\mathrm{A}(t', t)\right)^{*k} \mathrm{v} = \mathrm{e}_1^H \left(\mathrm{T}_n(t', t)\right)^{*k} \mathrm{e}_1, \quad k = 0, \dots, 2n-1.$$

[Giscard, P., 2020]. Hence:

$$\sum_{k=0}^{\infty} w^H A^{*k} v \approx \sum_{k=0}^{\infty} e_1^H (T_n)^{*k} e_1.$$

$*$-Lanczos Algorithm

Initialize: $\mathrm{v}_{-1} = \mathrm{w}_{-1} = 0$, $\mathrm{v}_0 = \mathrm{v}\, 1_*$, $\mathrm{w}_0^H = \mathrm{w}^H 1_*$.

$\alpha_0 = \mathrm{w}^H \mathrm{A}\, \mathrm{v}$,

$\mathrm{w}_1^H = \mathrm{w}^H \mathrm{A} - \alpha_0\, \mathrm{w}^H$,

$\widehat{\mathrm{v}}_1 = \mathrm{A}\, \mathrm{v} - \mathrm{v}\, \alpha_0$,

$\beta_1 = \mathrm{w}^H \mathrm{A}^{*2}\, \mathrm{v} - \alpha_0^{*2}$,

If β_1 is not $*$-invertible, then stop, otherwise,

$\mathrm{v}_1 = \widehat{\mathrm{v}}_1 * \beta_1^{*-1}$,

For $n = 2, \dots$

$\alpha_{n-1} = \mathrm{w}_{n-1}^H * \mathrm{A} * \mathrm{v}_{n-1}$,

$\mathrm{w}_n^H = \mathrm{w}_{n-1}^H * \mathrm{A} - \alpha_{n-1} * \mathrm{w}_{n-1}^H - \beta_{n-1} * \mathrm{w}_{n-2}^H$,

$\widehat{\mathrm{v}}_n = \mathrm{A} * \mathrm{v}_{n-1} - \mathrm{v}_{n-1} * \alpha_{n-1} - \mathrm{v}_{n-2}$,

$\beta_n = \mathrm{w}_n^H * \mathrm{A} * \mathrm{v}_{n-1}$,

If β_n is not $*$-invertible, then stop, otherwise,

$\mathrm{v}_n = \widehat{\mathrm{v}}_n * \beta_n^{*-1}$,

end.

$*$-Lanczos + Path Sum method

We get the approximation (model reduction)

$$w^H U(t',t)v \approx \Theta(t'-t) * R_*(T_n)_{11}(t',t),$$

The model reduction has to be interpreted in two way

- The size of T_n is much smaller, $n << N$;
- T_n is tridiagonal.

Going back to Path-sum

$$
\mathsf{T}_N(t',t) := \begin{bmatrix} \alpha_0(t',t) & \delta(t'-t) & & \\ \beta_1(t',t) & \alpha_1(t',t) & \ddots & \\ & \ddots & \ddots & \delta(t'-t) \\ & & \beta_{N-1}(t',t) & \alpha_{N-1}(t'-t) \end{bmatrix}
$$

From Path-sum method we have

$$
R_*(T_n)_{1,1}(t',t) = R_*(\alpha_0 + R_*(\alpha_1 + R_*(\alpha_2 + \dots) * \beta_2) * \beta_1);
$$

For $n = N$, we have exactness!

$$
\begin{aligned} \mathsf{A}(t',t)\Theta(t'-t) &= V_N(t',t) * \mathsf{T}_N(t',t) * W_N^H(t',t), \\ U(t',t) &= \Theta(t'-t) * R_*(T_N)_{11}(t',t). \end{aligned}
$$

Properties

A time-dependent matrix $\mathrm{A}(t')$ can be $*$-tridiagonalized in $\mathrm{D}(I)$ if

- $\mathrm{A}(t')$ is smooth in I.
- $\mathrm{A}(t')$ can be tridiagonalized (in the classical sense) starting from the same initial vectors w, v for every $t' \in I$; [Parlett, '92].

Error bound

Under the previous assumptions,

$$\left|\mathrm{w}^H \mathrm{U}\,\mathrm{v} - \Theta * \mathrm{R}_*(\mathrm{T}_n)_{1,1}\right| \leq \left(C^{2n} + D_n^{2n}\right) \frac{(t'-t)^{2n}}{(2n)!},$$

with the finite coefficients

$$C := \sup_{t' \in I} \|\tilde{\mathrm{A}}(t')\|_\infty, \quad D_n := \sup_{t',t \in I^2} \max_{0 \leq j \leq n-1} \left\{|\tilde{\alpha}_j(t',t)|, |\tilde{\beta}_j(t',t)|\right\}.$$

Example: time-dependent matrix

Consider the matrix

$$A = \begin{pmatrix} \cos(t') & 0 & 1 & 2 & 1 \\ 0 & \cos(t') - t' & 1 - 3t' & t' & 0 \\ 0 & t' & 2t' + \cos(t') & 0 & 0 \\ 0 & 1 & 2t' + 1 & t' + \cos(t') & t' \\ t' & -t' - 1 & -6t' - 1 & 1 - 2t' & \cos(t') - 2t' \end{pmatrix}.$$

The matrix does not commute with itself at different times and the corresponding differential system has no known analytical solution.

Example: time-dependent matrix

We get

$$\mathsf{T}_5 = \begin{pmatrix} \cos(t')\Theta & \delta & 0 & 0 & 0 \\ \frac{1}{2}(t'^2 - t^2)\Theta & \cos(t)\Theta & \delta & 0 & 0 \\ 0 & t(t'-t)\Theta & \widetilde{\alpha}_2(t',t)\Theta & \delta & 0 \\ 0 & 0 & -\frac{1}{2}(3t^2 - 4tt' + t'^2)\Theta & \widetilde{\alpha}_3(t',t)\Theta & \delta \\ 0 & 0 & 0 & (-2t^2 + 3tt' - t'^2)\Theta & \widetilde{\alpha}_4(t',t)\Theta \end{pmatrix},$$

with

$$\widetilde{\alpha}_2(t',t) = (t' - t)\sin(t) + \cos(t),$$

$$\widetilde{\alpha}_3(t',t) = \frac{1}{2}\left(4(t' - t)\sin(t) - \left((t - t')^2 - 2\right)\right)\cos(t),$$

$$\widetilde{\alpha}_4(t',t) = \frac{1}{6}\left(\left((t - t')^2 - 18\right)(t - t')\sin(t) + \left(6 - 9(t - t')^2\right)\cos(t)\right),$$

Example: time-dependent matrix

With the $*$-biorthonormal basis

$$\mathrm{V}_5 = \begin{pmatrix} \delta & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \delta^{(3)} & -2\delta^{(4)} \\ 0 & 0 & 0 & 0 & \delta^{(4)} \\ 0 & 0 & \delta'' & -\delta^{(3)} & \delta^{(4)} \\ 0 & \delta' & -2\delta'' & 2\delta^{(3)} & -3\delta^{(4)} \end{pmatrix}, \quad \mathrm{W}_5^H = \begin{pmatrix} \delta & 0 & 0 & 0 & 0 \\ 0 & 0 & \Theta & 2\Theta & \Theta \\ 0 & \Theta^{*2} & \Theta^{*2} & \Theta^{*2} & 0 \\ 0 & \Theta^{*3} & 2\Theta^{*3} & 0 & 0 \\ 0 & 0 & \Theta^{*4} & 0 & 0 \end{pmatrix}$$

The Dirac delta derivatives are coming from:

$$\beta_1^{*-1} = \frac{1}{t}\delta'(t'-t) * \delta'(t'-t), \qquad \beta_2^{*-1} = \frac{1}{t'}\delta'(t'-t) * \delta'(t'-t)$$

$$\beta_3^{*-1} = \frac{1}{t}\Theta(t'-t) * \delta^{(3)}(t'-t), \quad \beta_4^{*-1} = \frac{t'}{t^2}\Theta(t'-t) * \delta^{(3)}(t'-t).$$

Numerical outlook

To produce a numerical algorithm from $*$-Lanczos we need to:

- Approximate the $*$-product in $\mathrm{Sm}_\Theta(I)$ (numerical integration);
- Approximate the $*$-inverse (inverse of a quadrature formula?)
- How do such approximations work together?
- It is possible to formulate such approximation in terms of product and inversion of triangular matrix (cheap).

Warning

Rounding errors deeply affect (classical) Lanczos by loss of orthogonality. We expect a similar behavior in any numerical implementation of $*$-Lanczos. This must be investigated before confidently relying on the method in a computational setting.

References:

- P-L. G., S. P., *Lanczos-like method for the time-ordered exponential*, arXiv:1909.03437 [math.NA].
- P-L. G., S. P., *Lanczos-like algorithm for the time-ordered exponential: The $*$-inverse problem*, Applications of Mathematics, 65(6):807–827, 2020.
- P-L. G., S. P., *Tridiagonalization of systems of coupled linear differential equations with variable coefficients by a Lanczos-like method*, Linear Algebra and its Applications, 624:153–173, 2020.

Thank you for your attention!