Symbolic simplifications of exponentials of upper-triangular block matrices in Mathematica

Daniel Puzzuoli

December 15, 2018

This document describes the Mathematica notebook UTBSymbolicSimplifications. The main purpose of the notebook is the verification of instances of Conjecture 1 in [1]. We will state the conjecture and relevant facts associated with the functioning of the code, however, as this is only meant as a supplement, consult [1] for a full explanation and motivation.

Note that this code is written in Mathematica as this was the language we were using at the time, and I was in the mindset of using Mathematica's ability to directly work on symbolic expressions closely resembling the mathematical formulas we were working with. In retrospect, it would probably be simpler and more concise to use non-symbolic representations of these formulas, which could of course be implemented in any language.

1 Purpose

The work in [1] is centred around the computation of integrals of the form

$$U(t)\int_0^t dt_1 \cdots \int_0^{t_{m-1}} dt_m U^{-1}(t_1)A_1(t_1)U(t_1)\ldots U^{-1}(t_m)A_m(t_m)U(t_m), \tag{1}$$

for $A_i : [0,T] \to M_n$, and $U(t) = \mathcal{T}\exp\left(\int_0^t dt_1 G(t_1)\right)$ with $G : [0,T] \to M_n$. Building on the work of [2, 3], Theorem 1 in [1] demonstrates that such integrals may be computed as part of solutions to linear matrix differential equations, which are easy to solve, thereby avoiding potentially costly integral approximation methods. We will state Theorem 1 here; see [1] for a proof.

First, we introduce some notation. Let $B_{ij} : [0,T] \to M_n$ for $1 \leq i \leq j \leq m$. For $1 \leq i \leq m$, denote

$$U_i(t) = \mathcal{T}\exp\left(\int_0^t dt_1 B_{ii}(t_1)\right). \tag{2}$$

For $k \geq 2$ and indices $i_1, \ldots i_k$, denote

$$\begin{aligned}&\mathrm{Int}_{(i_1,\ldots,i_k)}(t)\\ &\quad = U_{i_1}(t)\int_0^t dt_1 \cdots \int_0^{t_{k-2}} dt_{k-1} U_{i_1}^{-1}(t_1)B_{i_1,i_2}(t_1)U_{i_2}(t_1)\ldots U_{i_{k-1}}^{-1}(t_{k-1})B_{i_{k-1},i_k}(t_{k-1})U_{i_k}(t_{k-1}),\end{aligned} \tag{3}$$

and for a single index i, denote

$$\text{Int}_{(i)}(t) = U_i(t). \tag{4}$$

Note that for $k \geq 2$ and indices $i_1, \ldots, i_k$, these definitions satisfy the recursive expression

$$\text{Int}_{(i_1,\ldots,i_k)}(t) = U_{i_1}(t) \int_0^t dt_1 U_{i_1}^{-1}(t_1) B_{i_1,i_2}(t_1) \text{Int}_{(i_2,\ldots,i_k)}(t_1). \tag{5}$$

Theorem 1. *Let $B_{ij} : [0,T] \to M_n$ for $1 \leq i \leq j \leq m$. Define $C_{ij} : [0,T] \to M_n$ implicitly by the equation*

$$\begin{pmatrix} C_{11}(t) & C_{12}(t) & \ldots & C_{1m}(t) \\ 0 & C_{22}(t) & \ldots & C_{2m}(t) \\ \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \ldots & C_{mm}(t) \end{pmatrix} = \mathcal{T} \exp \left[\int_0^t dt_1 \begin{pmatrix} B_{11}(t_1) & B_{12}(t_1) & \ldots & B_{1m}(t_1) \\ 0 & B_{22}(t_1) & \ldots & B_{2m}(t_1) \\ \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \ldots & B_{mm}(t_1) \end{pmatrix} \right]. \tag{6}$$

For all $1 \leq k \leq m$, $1 \leq j \leq m-k$, and $t \in [0,T]$, it holds that

$$C_{k,k}(t) = U_k(t), \tag{7}$$

and

$$C_{k,k+j}(t) = \text{Int}_{(k,k+j)}(t) + \sum_{r=1}^{j-1} \sum_{k<i_1<\cdots<i_r<k+j} \text{Int}_{(k,i_1,\ldots,i_r,k+j)}(t), \tag{8}$$

where the inner sum is over all indices $i_1, \ldots, i_r$ satisfying the relations, and U_i and Int *are as defined in the preceding discussion. Alternatively, these matrices can be given recursively as*

$$C_{k,k+j}(t) = \text{Int}_{(k,k+j)}(t) + \sum_{i=1}^{j-1} U_k(t) \int_0^t dt_1 U_k^{-1}(t_1) B_{k,k+i}(t_1) C_{k+i,k+j}(t_1). \tag{9}$$

The conjecture concerns using the above theorem to compute integrals of the form

$$U(t) \int_0^t dt_1 \int_0^{t_1} dt_2 p(t_1,t_2) U^{-1}(t_1) A_1(t_1) U(t_1) U^{-1}(t_2) A_2(t_2) U(t_2), \tag{10}$$

where $p(t_1,t_2)$ is some polynomial in t_1 and t_2, and again, $U(t) = \mathcal{T} \exp\left(\int_0^t dt_1 G(t_1)\right)$ for some function $G : [0,T] \to M_n$. That is, the goal is to find an upper triangular block matrix whose time-ordered exponential contains the above integral. Ideally, such an upper triangular block matrix will not have any additional time-dependence outside of what is already contained in $G(t)$, $A_1(t)$, and $A_2(t)$, so as to not introduce any new difficulties in the computation.

As described in [1], we simplify the problem by first asking, for non-negative integers k_1, k_2, how to compute the special case $p(t_1,t_2) = t_1^{k_1} t_2^{k_2}$, i.e. when p is just a monomial. Due to the linearity of integration, computing the integral for an arbitrary p would thus only require taking a linear combination of the result of such computations.

In [1], a procedure is presented for constructing such a generator, which we denote $L_{k_1,k_2}(t)$. The generator is of the form $L_{k_1,k_2} : [0,T] \to M_{3k_1+k_2+3}(M_n)$, for which all blocks are 0 except for:

- Every diagonal block is $G(t)$,
- For the first off diagonal, the first $k_1 + 1$ blocks are 0, and the remaining blocks by

$$(k_1, k_1 - 1, \ldots, 0, k_1 + k_2, k_1 + k_2 - 1, \ldots, 1), \tag{11}$$

  and
- The $(k_1 + 1)^{th}$ off diagonal is given by $k_1 + 1$ repetitions of $A_1(t)$, then $k_1 + 1$ repetitions of $A_2(t)$, followed by zeros.

As described in [1], $L_{k_1,k_2}(t)$ was systematically constructed so that the top right block of $\mathcal{T}\exp\left(\int_0^t dt_1 L_{k_1,k_2}(t_1)\right)$ is exactly the integral in Equation (10) for the monomial case $p(t_1, t_2) = t_1^{k_1} t_2^{k_2}$.

By chance, we observed that $\mathcal{T}\exp\left(\int_0^t dt_1 L_{k_1,k_2}(t_1)\right)$ seems to contain much more than that. Denote the integral

$$\mathcal{D}(t^i A_1, t^j A_2) = U(t)\int_0^t dt_1 \int_0^{t_1} dt_2 t_1^i t_2^j U^{-1}(t_1) A_1(t_1) U(t_1) U^{-1}(t_2) A_2(t_2) U(t_2). \tag{12}$$

Any integral of the form in Equation (10) for a polynomial whose powers of t_1 never exceed k_1, and whose powers of t_2 never exceed k_2, can be written as an element of

$$\operatorname{span}\left\{\mathcal{D}(t^i A_1, t^j A_2) : 0 \le i \le k_1, 0 \le j \le k_2\right\}. \tag{13}$$

Our conjecture is that such an integral, for *any* polynomial with powers restricted as above, can be extracted from $\mathcal{T}\exp\left(\int_0^t dt_1 L_{k_1,k_2}(t_1)\right)$. In particular, we conjecture the following.

Conjecture 1. *It holds that the top right* $(k_1 + 1) \times (k_2 + 1)$ *blocks of*

$$\mathcal{T}\exp\left(\int_0^t dt_1 L_{k_1,k_2}(t_1)\right) \tag{14}$$

is a basis for the vector space of expressions

$$\operatorname{span}\{\mathcal{D}(t^i A_1, t^j A_2) : 0 \le i \le k_1, 0 \le j \le k_2\}. \tag{15}$$

Hence, assuming the conjecture is true, the generator $L_{k_1,k_2}(t)$ provides a very convenient way for computing the integral in Equation (10) for any polynomial p of the form

$$p(t_1, t_2) = \sum_{i=0}^{k_1}\sum_{j=0}^{k_2} \alpha_{ij} t_1^i t_2^j. \tag{16}$$

2 The `Mathematica` notebook overview

The `Mathematica` notebook is written to verify Conjecture 1 for a given value of k_1 and k_2. Doing so requires understanding the blocks of $\mathcal{T}\exp\left(\int_0^t dt_1 L_{k_1,k_2}(t_1)\right)$, and given the generator $L_{k_1,k_2}(t)$, the form of the blocks of the time-ordered exponential can be deduced

by applying Theorem 1. Hence, the primary function of the notebook is to take an upper triangular block generator, and deduce the structure of its time-ordered exponential.

For an arbitrary upper triangular block matrix, the expressions resulting from Theorem 1 are likely very complex and uninformative. However, as all blocks in $L_{k_1,k_2}(t)$ are either 0, $G(t)$, $A_1(t)$, $A_2(t)$, or multiples of the identity, the expressions appearing in Theorem 1 can be greatly simplified, as many terms will be 0, and many of the integrals can be explicitly carried out.[1]

Hence, the most general functionality in the notebook is to take in a symbolic specification of an upper triangular block matrix, and output a symbolic description of the blocks of the time-ordered exponential, where simplifications of the expressions are carried out where possible.

We apply to the generator $L_{k_1,k_2}(t)$, and then look at the top right $(k_1+1)\times(k_2+1)$ blocks. On each block, we then perform a mapping from the symbolic expressions for the integrals $\mathcal{D}(t^i A_1, t^j A_2)$ to elementary vectors of dimension $(k_1+1)(k_2+1)$, and finally verify that the resultant vectors are in fact linearly independent, and therefore a basis.[2] This constitutes a verification of Conjecture 1 for a given pair (k_1,k_2).

The notebook contains the output of this verification procedure for all pairs (k_1,k_2) with $0\leq k_2\leq 15$ and $0\leq k_2\leq 15$. It is found that the conjecture is true for all such pairs. For applications it is unlikely that powers beyond this would be considered, though if a pair (k_1,k_2) not in this set is desired, the conjecture may simply be verified for this pair using this code.

The last piece of the code assumes Conjecture 1 is true, and outputs an equation for how to combine the blocks of $\mathcal{T}\exp\left(\int_0^t dt_1 L_{k_1,k_2}(t_1)\right)$ to compute

$$U(t)\int_0^t dt_1\int_0^{t_1} dt_2 p(t_1,t_2)U^{-1}(t_1)A_1(t_1)U(t_1)U^{-1}(t_2)A_2(t_2)U(t_2). \tag{17}$$

for any polynomial with degree (k_1,k_2).

3 Code design and description

The functionality fo the code is mainly implemented via `Mathematica` replacement rules, then calling the built-in `Simplify` function with these rules.

3.1 Reserved symbols and primitive expressions

The first step is specifying a representation for expressions appearing in Theorem 1 in `Mathematica`. We reserve the symbol "t" to represent integration variables, and the symbol $\mathbb{1}$ to represent the identity matrix, which can be produced in `Mathematica` by typing `Esc d s 1 Esc`.

There are two types of expressions which can appear in simplifications of Theorem 1 that we consider. In this appendix we call these *primitive expressions*, and use the term to refer to both the mathematical objects, as well as their representation in the code. The first

[1] For the generators appearing in quantum control applications, the blocks are often sparse and repetitive, and so an automated process for simplifying the outputs of Theorem 1 may be useful for other purposes as well.

[2] Of course, when doing the mapping, we also need to check that the blocks *only* contain integrals of the form $\mathcal{D}(t^i A_1, t^j A_2)$, as if they did, it would disprove the conjecture.

kind of primitive expression is the nested integral (note the additional appearance of the pre-factor t^m):

$$t^m U_1(t) \int_0^t dt_1 \cdots \int_0^{t_{n-1}} dt_n t_1^{k_1} \ldots t_n^{k_n} U_1^{-1}(t_1) A_1(t_1) V_1(t_1) \ldots U_n^{-1}(t_n) A_n(t_n) V_n(t_n), \quad (18)$$

which is represented in the code using the head "Int":

$$\mathrm{Int}\Big[t^m, \{U_1, t^{k_1} * A_1, V_1\}, \ldots, \{U_n, t^{k_n} * A_n, V_n\}\Big]. \quad (19)$$

The second primitive is simply an expression not appearing in an integral, for example $t^m U(t)$, which is represented in the code using the head "Ex":

$$\mathrm{Ex}[t^m, U]. \quad (20)$$

We remark that the programmatic representation of nested integrals is slightly redundant for representing expressions resulting from Theorem 1, as primitives from this theorem will always have $V_{i-1} = U_i$, but there is no harm in this redundancy.

A final technical detail is a special representation for time-independent scalar multiples of matrices, i.e. $aA(t)$, for $a \in \mathbb{C}$ and A a matrix valued function. These are represented in the code in the following way:

$$\mathrm{SM}[a, A], \quad (21)$$

which eases the discrimination between symbols representing (potentially time-dependent) matrices and time-independent scalars.

3.2 Simplification of primitives via replacement rules

The simplification of primitive expressions is carried out by defining *replacement rules*, then applying those rules to primitives using the internal functions of `Mathematica`. The replacement rules we specify correspond to basic simplifications of the nested integrals arising in Theorem 1. The rules are broken into groups.

- *Zeroes*

  The detection of zero matrices:

  $$\mathrm{Int}[x\_\_\_, \{U\_, 0, V\_\}, y\_\_\_] \to 0. \quad (22)$$

- *Linearity of integration*

  Factoring scalars:

  $$\mathrm{Int}[x\_\_\_, \{U\_, \mathrm{SM}[a\_, A\_], V\_\}, y\_\_\_] \to a\mathrm{Int}[x, \{U, A, V\}, y], \quad (23)$$

  and linear combinations:

  $$\mathrm{Int}[x\_\_\_, \{U\_, A\_ + B\_, V\_\}, y\_\_\_] \to \mathrm{Int}[x, \{U, A, V\}, y] + \mathrm{Int}[x, \{U, B, V\}, y]. \quad (24)$$

- *Performing integrals*

In some cases, an integral can be explicitly performed. The case that we handle is when $U_i = V_i$ and $A_i(t) = t^m \mathbb{1}$ for some index i in Equation (18). As an explicit example, it holds that

$$\begin{aligned}\int_0^t dt_1 \int_0^{t_1} dt_2 \int_0^{t_2} dt_3 t_2^m U_1^{-1}(t_1) A_1(t_1) V_1(t_1) U_3^{-1}(t_3) A_3(t_3) V_3(t_3) \\ = \frac{1}{m+1}\Bigg(\int_0^t dt_1 \int_0^{t_1} dt_3 t_1^{m+1} U_1^{-1}(t_1) A_1(t_1) V_1(t_1) U_3^{-1}(t_3) A_3(t_3) V_3(t_3) \\ - \int_0^t dt_1 \int_0^{t_1} dt_3 t_3^{m+1} U_1^{-1}(t_1) A_1(t_1) V_1(t_1) U_3^{-1}(t_3) A_3(t_3) V_3(t_3) \Bigg).\end{aligned} \tag{25}$$

Handling all possible versions of this simplification with replacement rules must be broken into four cases based on the order of the nested integral, as well as where the simplification appears in the nest. Some care must also be taken with powers of t; the pattern $t^{m\_}$ will detect powers of t when $m \geq 2$, but not the expressions "1" and "t" as, symbolically, they are not powers of t. Thus, one must create replacement rules for these cases separately.

- Case 1: A first order integral that can be performed.

$$\begin{aligned}\text{Int}[x\_, \{U\_, \mathbb{1}, U\_\}] &\to \text{Ex}[x * t, U] \\ \text{Int}[x\_, \{U\_, t\mathbb{1}, U\_\}] &\to \frac{1}{2}\text{Ex}[x * t^2, U] \\ \text{Int}[x\_, \{U\_, t^{m\_}\mathbb{1}, U\_\}] &\to \frac{1}{m+1}\text{Ex}[x * t^{m+1}, U]\end{aligned} \tag{26}$$

- Case 2: An integral of order ≥ 2 where the last integral can be performed.

$$\begin{aligned}\text{Int}[x\_\_\_, \{U1\_, A\_, V1\_\}, \{U\_, \mathbb{1}, U\_\}] &\to \text{Int}[x, \{U1, tA, V\}] \\ \text{Int}[x\_\_\_, \{U1\_, A\_, V1\_\}, \{U\_, t\mathbb{1}, U\_\}] &\to \frac{1}{2}\text{Int}[x, \{U1, t^2 A, V\}] \\ \text{Int}[x\_\_\_, \{U1\_, A\_, V1\_\}, \{U\_, t^{m\_}\mathbb{1}, U\_\}] &\to \frac{1}{m+1}\text{Int}[x, \{U1, t^{m+1} A, V\}]\end{aligned} \tag{27}$$

- Case 3: An integral of order ≥ 2 where the first integral can be performed.

$$\begin{aligned}&\text{Int}[y\_, \{U\_, \mathbb{1}, U\_\}, \{U1\_, B\_, V1\_\}, x\_\_\_] \\ &\qquad \to \text{Int}[y * t, \{U1, B, V1\}, x] - \text{Int}[y, \{U1, tB, V1\}, x] \\ &\text{Int}[y\_, \{U\_, t\mathbb{1}, U\_\}, \{U1\_, B\_, V1\_\}, x\_\_\_] \\ &\qquad \to \frac{1}{2}\left(\text{Int}[y * t^2, \{U1, B, V1\}, x] - \text{Int}[y, \{U1, t^2 B, V1\}, x]\right) \\ &\text{Int}[y\_, \{U\_, t^{m\_}\mathbb{1}, U\_\}, \{U1\_, B\_, V1\_\}, x\_\_\_] \\ &\qquad \to \frac{1}{m+1}\left(\text{Int}[y * t^{m+1}, \{U1, B, V1\}, x] - \text{Int}[y, \{U1, t^{m+1} B, V1\}, x]\right)\end{aligned}$$

– Case 4: An integral of order ≥ 3 where an "internal" integral can be performed.

$$\begin{aligned}
&\text{Int}[x_{___}, \{U1_{_}, A1_{_}, V1_{_}\}, \{U_{_}, \mathbb{1}, U_{_}\}, \{U2_{_}, A2_{_}, V2_{_}\}, y_{___}] \\
&\quad \to \text{Int}[x, \{U1, tA1, V1\}, \{U2, A2, V2\}, y] - \text{Int}[x, \{U1, A1, V1\}, \{U2, tA2, V2\}, y] \\
&\text{Int}[x_{___}, \{U1_{_}, A1_{_}, V1_{_}\}, \{U_{_}, t\mathbb{1}, U_{_}\}, \{U2_{_}, A2_{_}, V2_{_}\}, y_{___}] \\
&\quad \to \frac{1}{2}\left(\text{Int}[x, \{U1, t^2A1, V1\}, \{U2, A2, V2\}, y] - \text{Int}[x, \{U1, A1, V1\}, \{U2, t^2A2, V2\}, y]\right) \\
&\text{Int}[x_{___}, \{U1_{_}, A1_{_}, V1_{_}\}, \{U_{_}, t^{m\text{-}}\mathbb{1}, U_{_}\}, \{U2_{_}, A2_{_}, V2_{_}\}, y_{___}] \\
&\quad \to \frac{1}{m+1}\left(\text{Int}[x, \{U1, t^{m+1}A1, V1\}, \{U2, A2, V2\}, y] - \text{Int}[x, \{U1, A1, V1\}, \{U2, t^{m+1}A2, V2\}, y]\right)
\end{aligned}$$

3.3 Computing general expressions from Theorem 1

Computing general expressions arising from Theorem 1 consists of programmatically implementing the mapping

$$\begin{pmatrix} B_{11}(t) & B_{12}(t) & \dots & B_{1n}(t) \\ 0 & B_{22}(t) & \dots & B_{2n}(t) \\ \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \dots & B_{nn}(t) \end{pmatrix} \to \begin{pmatrix} C_{11}(t) & C_{12}(t) & \dots & C_{1n}(t) \\ 0 & C_{22}(t) & \dots & C_{2n}(t) \\ \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \dots & C_{nn}(t) \end{pmatrix}, \tag{28}$$

where the right-hand side is the time-ordered exponential of the left-hand side. The function `TOExponential` implements this mapping. As

$$C_{ii}(t) = U_i(t) = \mathcal{T} \exp\left\{ \int_0^t dt_1 B_{ii}(t_1) \right\} \tag{29}$$

(i.e. $C_{ii}(t)$ depends only on $B_{ii}(t)$) rather than specifying $B_{ii}(t)$, the user specifies the symbols for the $U_i(t)$ matrices as an input. That is, `TOExponential` takes in two lists of symbols:

$$\begin{aligned} \{U_1, \dots, U_n\}, \text{ and } \{&\{B_{12}, \dots, B_{1n}\}, \\ &\{B_{23}, \dots, B_{2n}\}, \\ &\vdots, \\ &\{B_{n-1,n}\}\}, \end{aligned} \tag{30}$$

where the first list of symbols represents the diagonal blocks of the time-ordered exponential, and the second list represents the off-diagonal blocks of the matrix to be time-ordered exponentiated. The output is a list of symbols

$$\begin{aligned} C = \{&\{C_{11}, \dots, C_{1n}\}, \\ &\{C_{22}, \dots, C_{2n}\}, \\ &\vdots, \\ &\{C_{nn}\}\}, \end{aligned} \tag{31}$$

representing the upper triangle of matrices in the right-hand side of Equation (28). In the notation of Theorem 1, we have that the expression for $C_{k,k+j}(t)$ is contained in $C[[k, j+1]]$.

The `Mathematica` notebook has several example applications of this code. One basic use-case is to show that:

$$\begin{pmatrix} U(t) & U(t)\int_0^t dt_1 U^{-1}(t_1)A(t_1)U(t_1) & U(t)\int_0^t dt_1 \int_0^{t_1} dt_2 U^{-1}(t_1)A(t_1)U(t_1)U^{-1}(t_2)B(t_2)U(t_2) \\ 0 & U(t) & U(t)\int_0^t dt_1 U^{-1}(t_1)B(t_1)U(t_1) \\ 0 & 0 & U(t) \end{pmatrix} = \mathcal{T}\exp\left\{\int_0 \right.$$

(32)

where $U(t) = \mathcal{T}\exp\{\int_0^t dt_1 G(t_1)\}$. In this case, the inputs to `TOExponential` are the lists

$$\{U, U, U\}, \text{ and } \{\{A, 0\}, \{B\}\}, \tag{33}$$

and the output is the list

$$\begin{aligned} \{\{\text{Ex}[1, U], \text{Int}[1, \{U, A, U\}], \text{Int}[1, \{U, A, U\}, \{U, B, U\}]\}, \\ \{\text{Ex}[1, U], \text{Int}[1, \{U, BU\}]\}, \\ \{\text{Ex}[1, U]\}\}. \end{aligned} \tag{34}$$

By interpreting the structure of this output according to the data representation for primitives in Section 3.1, we see that this output is correct.

3.3.1 Implementation of `TOExponential`

The implementation of `TOExponential` requires two pieces: the symbolic construction of expressions for the $C_{k,k+j}(t)$ matrices given in Theorem 1, and the simplification of these expressions via the replacement rules of Section 3.2. Due to the recursive structure of the matrices in Theorem 1, it is both conceptually natural and computationally more efficient to perform this computation in a recursive fashion. The recursion proceeds by computing successive off-diagonals of the $C_{k,k+j}(t)$ matrices (i.e. successive values of j for all k). As a reminder, the recursion relation for the $C_{k,k+j}(t)$ matrices for $j \geq 1$ is:

$$C_{k,k+j}(t) = \text{Int}_{(k,k+j)}(t) + \sum_{i=1}^{j-1} U_k(t) \int_0^t dt_1 U_k^{-1}(t_1) A_{k,k+i}(t_1) C_{k+i,k+j}(t_1). \tag{35}$$

Roughly, the computation goes as follows:

1. Base case: Initialize the case $j = 1$ using Equation (35) and apply replacement rules to simplify any expressions.
2. Recursive step: Construct expressions for $C_{k,k+j}(t)$ using Equation (35) and the already computed $C_{k,k+i}(t)$ matrices for $1 \leq i < j$. Apply replacement rules to simplify resulting expressions.

The most basic recursive construction step is to produce a single term in the sum in Equation (35). That is, given $C(t)$ (a linear combination of primitives) and two symbols U and A, we must programmatically produce the mapping

$$C(t) \to U(t) \int_0^t dt_1 U^{-1}(t_1) A(t_1) C(t_1). \tag{36}$$

This mapping is performed by two functions both named `RecursiveRule`, where the second handles the case when the A symbol is given as some scalar multiple $\mathrm{SM}[c, A]$. The replacement rules are

$$\mathrm{Ex}[q\_, z\_] \to \mathrm{Int}[1, \{U, q * A, z\}] \tag{37}$$

and

$$\mathrm{Int}[q\_, \{U1\_, A1\_, V1\_\}, z\_\_\_] \to \mathrm{Int}[1, \{U, q * A, U1\}, \{U1, A1, V1\}, z], \tag{38}$$

and for the version which handles scalar multiples they are

$$\mathrm{Ex}[q\_, z\_] \to c * \mathrm{Int}[1, \{U, q * A, z\}] \tag{39}$$

and

$$\mathrm{Int}[q\_, \{U1\_, A1\_, V1\_\}, z\_\_\_] \to c * \mathrm{Int}[1, \{U, q * A, U1\}, \{U1, A1, V1\}, z]. \tag{40}$$

A full recursion step (for computing the next off-diagonal of elements from the previous) is implemented by `RecursiveConstruct`, which simply uses the `RecursiveRule` functions to construct the whole of the right-hand-side of Equation (35).

Lastly, the implementation of `TOExponential` is to do the initialization step of constructing and simplifying the $C_{k,k+1}(t)$ matrices, then recursively calling `RecursiveConstruct` to populate the output consisting of all $C_{k,k+j}(t)$ matrices.

3.4 Bivariate polynomials and Conjecture 1

Here we describe code for working with and analyzing the generators $L_{k_1,k_2}(t)$ for the purposes of verifying Conjecture 1. A description of $L_{k_1,k_2}(t)$ is in Section 1, in the lead up to the statement of Conjecture 1.

The first step is simply to produce the generator $L_{k_1,k_2}(t)$ and its time-ordered exponential. The functions `BPODGenerator` and `BPGenerator` both serve the function of specifying $L_{k_1,k_2}(t)$, with the first producing the upper-off-diagonal pieces of the generator, and the second including the diagonal. The function `BPTOExponential` simply applies the function `TOExponential` of Section 3.3 to the generator given in `BPGenerator` and `BPODGenerator`.

The problems of verifying Conjecture 1, and providing explicit linear combinations of the blocks of $\mathcal{T} \exp\left(\int_0^t dt_1 L_{k_1,k_2}(t_1)\right)$ for computing integrals of the form

$$U(t) \int_0^t dt_1 \int_0^{t_1} dt_2 p(t_1, t_2) U^{-1}(t_1) A_1(t_1) U(t_1) U^{-1}(t_2) A_2(t_2) U(t_2) \tag{41}$$

for polynomials of degree at most (k_1, k_2), are fundamentally problems of linear algebra, and hence it is necessary to represent the upper right $(k_1 + 1) \times (k_2 + 1)$ blocks of $\mathcal{T} \exp\{\int_0^t dt_1 L_{k_1,k_2}(t)\}$ in a way that they may be analyzed using the linear algebra functions in `Mathematica`.

3.4.1 Linear algebraic representation

Due to the linearity of integration, for a polynomial p of degree (k_1, k_2), the expression in Equation (41) may be viewed as member of the vector space of expressions

$$\mathrm{span}\left\{\mathcal{D}(t^i A_1, t^j A_2) : 0 \le i \le k_1, 0 \le j \le k_2\right\}, \tag{42}$$

where, again

$$\mathcal{D}(t^i A_1, t^j A_2) = U(t) \int_0^t dt_1 \int_0^{t_1} dt_2 t_1^i t_2^j U^{-1}(t_1) A_1(t_1) U(t_1) U^{-1}(t_2) A_2(t_2) U(t_2), \quad (43)$$

and each $\mathcal{D}(t^i A_1, t^j A_2)$ is considered to be linearly independent from all others. Denoting this vector space of expressions as $P(k_1, k_2)$, we call the $\mathcal{D}(t^i A_1, t^j A_2)$ the *elementary basis* for this vector space.

We may represent $\mathrm{P}(k_1, k_2)$ as column vectors by defining a linear mapping $V : \mathrm{P}(k_1, k_2) \to \mathbb{C}^{(k_1+1)(k_2+1)}$ which acts on the standard basis as

$$V\mathcal{D}(t^i A_1, t^j A_2) = e_{(k_2+1)i+j+1}, \quad (44)$$

where e_n is the column vector with a 1 in the n^{th} position and a 0 everywhere else (i.e. V sends the standard basis of $\mathrm{P}(k_1, k_2)$ to the standard basis of $\mathbb{C}^{(k_1+1)(k_2+1)}$). Note that ordering of the images of the standard basis elements of $\mathrm{P}(k_1, k_2)$ according to this mapping is the *lexicographic ordering* of the basis elements according to the powers (i, j) of t_1 and t_2 appearing in the integral.

The function `BPVectorRep` constructs a list of replacement rules that corresponding to this mapping. The function `BPTopRightBlockVectors` produces a matrix whose column vectors correspond to the top-right $(k_1 + 1) \times (k_2 + 1)$ blocks of $\mathcal{T} \exp\left(\int_0^t dt_1 L_{k_1,k_2}(t)\right)$ under the above mapping, with the blocks ordered in terms of the lexicographic ordering of their indices (which is automatically implemented by the `Flatten` function in `Mathematica`).

3.4.2 Testing Conjecture 1

With the terminology of the previous section, the content of Conjecture 1 is simply that the top right $(k_1+1) \times (k_2+1)$ blocks of the time-ordered exponential of $\mathcal{T} \exp\left(\int_0^t dt_1 L_{k_1,k_2}(t)\right)$ are a basis for the formal vector space $\mathrm{P}(k_1, k_2)$. This is equivalent to the columns of the matrix Q output by `BPTopRightBlockVectors` being a basis for $\mathbb{C}^{(k_1+1)(k_2+1)}$. The function `TestClaim1` checks if this is the case by computing the rank of the matrix Q, with the claim being true if and only if the rank is $(k_1 + 1)(k_2 + 1)$. Note that before checking this, the code checks if any primitive expressions are present (i.e. whether the symbols "Ex" or "Int" remain, which would mean that there are expressions in the top right $(k_1 + 1) \times (k_2 + 1)$ blocks appearing that are not expected). If any such expressions are found, the function outputs the value ∞ indicating that Conjecture 1 has failed catastrophically. If no such expressions are found, the rank is checked, with an output of 1 meaning Conjecture 1 is verified, and 0 meaning it is invalidated. For example, the matrix Q in the $k_1 = k_2 = 1$ case is

$$\begin{pmatrix} 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}, \quad (45)$$

which is clearly full rank.

The function `TestClaim1Range` applies `TestClaim1` to a range of values of k_1 and k_2. We have tested this claim for $0 \leq k_1, k_2 \leq 15$ and have found it to be true in all cases. While

this doesn't constitute a proof, it does provide confidence that it is true in general, and in practical settings one can simply check its validity for a specific k_1 and k_2 of interest.

3.4.3 Computing integrals with bivariate polynomials

Given an arbitrary polynomial $p(t_1, t_2) = \sum_{i=0}^{k_1} \sum_{j=0}^{k_2} a_{i,j} t_1^i t_2^j$ we wish to write the integral

$$U(t) \int_0^T dt_1 \int_0^{t_1} dt_2 p(t_1, t_2) U^{-1}(t_1) A_1(t_1) U(t_1) U^{-1}(t_2) A_2(t_2) U(t_2) \tag{46}$$

$$= \sum_{i=0}^{k_1} \sum_{j=0}^{k_2} a_{i,j} \mathcal{D}(t^i A_1, t^j A_2) \tag{47}$$

as a linear combination of the top-right $(k_1+1) \times (k_2+1)$ blocks of $\mathcal{T} \exp\left(\int_0^t dt_1 L_{k_1,k_2}(t)\right)$. Assuming the truth of Conjecture 1, the top-right blocks of this time-ordered exponential are a basis for $\mathrm{P}(k_1, k_2)$, which we will call the *exponential basis*. Let $S : \mathrm{P}(k_1, k_2) \to \mathbb{C}^{(k_1+1)(k_2+1)}$ be the mapping which takes an element of $\mathrm{P}(k_1, k_2)$ and returns the column vector of coefficients corresponding to expanding it according to the exponential basis (in lexicographic ordering).

The matrix Q output from `BPTopRightBlockVectors` then simply represents a change of basis from the image of the exponential basis under S to the image of the standard basis under V. That is, for every $r \in \mathrm{P}(k_1, k_2)$, it holds that

$$Vr = QSr. \tag{48}$$

Hence, the vector of coefficients of r representing its expansion in the exponential basis is

$$Sr = Q^{-1} Vr. \tag{49}$$

As an example, in the $k_1 = k_2 = 1$ case, for an arbitrary $r \in \mathrm{P}(k_1, k_2)$ corresponding to the polynomial

$$p(t_1, t_2) = a_{0,0} + a_{0,1} t_2 + a_{1,0} t_1 + a_{1,1} t_1 t_2, \tag{50}$$

we have

$$Vr = \begin{pmatrix} a_{0,0} \\ a_{0,1} \\ a_{1,0} \\ a_{1,1} \end{pmatrix}, \text{ and } Q^{-1} = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 \end{pmatrix}, \tag{51}$$

and hence

$$Q^{-1} Vr = \begin{pmatrix} a_{1,0} \\ a_{1,1} \\ a_{0,0} \\ a_{0,1} - a_{1,0} \end{pmatrix}. \tag{52}$$

Denoting the blocks of $\mathcal{T}\exp\left(\int_0^t dt_1 L_{1,1}(t)\right)$ by $C_{i,j}(t)$, we may write the desired integral explicitly as a linear combination of $C_{i,j}(t)$ matrices by taking the following dot product

$$\begin{pmatrix} C_{1,6}(t) \\ C_{1,7}(t) \\ C_{2,6}(t) \\ C_{2,7}(t) \end{pmatrix} \cdot \begin{pmatrix} a_{1,0} \\ a_{1,1} \\ a_{0,0} \\ a_{0,1} - a_{1,0} \end{pmatrix} = a_{1,0}C_{1,6}(t) + a_{1,1}C_{1,7}(t) + a_{0,0}C_{2,6}(t) + (a_{0,1} - a_{1,0})C_{2,7}(t), \tag{53}$$

where the vector of $C_{i,j}(t)$ matrices is just the top 2×2 blocks of $\mathcal{T}\exp\left(\int_0^t dt_1 L_{1,1}(t)\right)$ in lexicographic ordering.

The function `PolynomialBlockDecomposition` outputs the right hand side of Equation (53) for an arbitrary k_1 and k_2 by performing the above computation in generality. An example when $k_1 = k_2 = 2$ is included in the code.

References

[1] H. Haas, D. Puzzuoli, and D. Cory. To appear, update once paper is done.

[2] C. Van Loan. Computing integrals involving the matrix exponential. *IEEE Transactions on Automatic Control*, 23(3):395–404, June 1978.

[3] F. Carbonell, J. C. Jímenez, and L. M. Pedroso. Computing multiple integrals involving matrix exponentials. *Journal of Computational and Applied Mathematics*, 213(1):300–305, March 2008.