Scientific Research Publishing

Journal of Applied Mathematics and Physics, 2026, 14(11), 4671-4697
https://www.scirp.org/journal/jamp
ISSN Online: 2327-4379
ISSN Print: 2327-4352

Matrix Generators of GL(n) and SU(n), for Order n up to Six

Richard Shurtleff

Wentworth Institute of Technology, Boston, USA
Email: shurtleffr@wit.edu, momentum.matrix@gmail.com

How to cite this paper: Shurtleff, R. (2026) Matrix Generators of GL(n) and SU(n), for Order n up to Six. *Journal of Applied Mathematics and Physics*, **14**, 4671-4697.
https://doi.org/10.4236/jamp.2026.1411229

Received: October 13, 2026
Accepted: November 21, 2026
Published: November 24, 2026

Copyright © 2026 by author(s) and Scientific Research Publishing Inc.
This work is licensed under the Creative Commons Attribution International License (CC BY 4.0).
http://creativecommons.org/licenses/by/4.0/

Open Access

Abstract

Formulas available in the literature produce matrix generators for an irreducible representation (irrep) of the general linear Lie algebra gl(n). A well-known linear transformation with complex coefficients yields a basis for the special unitary Lie algebra su(n). This article simplifies the task further by presenting a Mathematica notebook that calculates matrix generators for irreps of gl(n) and su(n), $n = 2, 3, 4, 5, 6$.

Keywords

Lie Algebra, Matrix Representations, General Linear Group, Special Unitary Group

1. Introduction

The Lie algebra gl(n) can be represented by $n \times n$ matrices, with scalar multiplication by real numbers and matrix multiplication defined so that the matrices may be added and multiplied as usual. The matrix commutator serves as the Lie bracket. Many other algebras can be represented by $n \times n$ matrices with various restrictions. We consider the Lie algebra su(n), which can be represented by the set of traceless Hermitian $n \times n$ matrices [1]-[3].

A computer program that provides matrix representations of these Lie algebras offers a convenient utility, an aid in the study of the Lie groups and their applications in the sciences. With the certainty of arithmetic, matrices can quickly check logic and calculations and immediately evaluate invariants. Having such a utility available is a potential asset.

The Gel'fand-Tsetlin (GT) formalism [1] [4] [5] provides a method for calculating matrix generators. In accordance with the complexity of the task, the formulas appear to be complicated. We have coded the formulas to avoid the need to

confront them. The computer program implements the Louck-Biedenharn formulas [6] [7], formulas that extend the GT formulas in a way that avoids employing commutation relations (CR) and thereby simplifies the coding.

The Wolfram Mathematica notebook [8] *Matrix Generators for Lie Algebras gl*(n) *and su*(n), *the Mathematica Notebook*, included in Section 5, is a computer program that calculates matrix generators for irreducible representations (irreps) of gl(n) and su(n) for $n = 2,3,4,5,6$. We use the Gel'fand-Tsetlin (GT) formalism [1] [4] [5] with GT patterns to determine the irrep and to parameterize the vector space upon which the matrices act.

The order n of gl(n) and su(n) is restricted to $n = 2,3,4,5,6$. The matrices for $n = 1$ are numbers, making gl(n) abelian, unlike the algebras for $n \geq 2$. At the high end, we stop at $n = 6$. Following the progression set by $n = 4,5,6$, excessive additional code would be required for $n = 7$. However, with unlimited program length, one expects that the progression could be continued to orders n larger than six.

Section 2 briefly explains the interplay of Lie groups and algebras, includes an example of the matrices produced by the notebook, and sets some of the notation.

By matrix multiplication, a matrix acts on a vector to produce a vector. Section 3 discusses the vector spaces, their bases, and the Gel'fand-Tsetlin patterns that parameterize the base vectors.

The order of entries in the rows and columns of matrices depends on the sequence of vectors in the basis of the vector space. In the literature, for example, in [7], the sequence is defined in words in a way that applies to all orders n.

However, to code a sequence function for the Mathematica notebook, we prefer the sequence functions to be algebraic. The sequence functions are found by iteration of the order n of gl(n). An algebraic function of GT patterns is assumed for $n = 2$. Then a rule that defines the function for n in terms of the function for $n-1$ gives the sequence functions for $n \geq 3$. It should be noted that our sequence functions yield different sequences in some sets of GT patterns than those in [7].

The sequence function maps GT patterns onto integers between one and the dimension of the irrep. Inversely, each integer in that range, whether it is a row index or a column index, determines one GT pattern.

In Section 4, the Louck-Biedenharn formulas give the nonzero entry at the intersection of a row and a column as functions of the GT patterns of the entry's row and column. Thus, GT patterns implicitly determine the row, column, and value of the matrix generators' entries. All of this is encoded in the Mathematica notebook in Section 5.

The program in Section 5 runs on the Mathematica app. As discussed in Section 3, the desired gl(n) or su(n) irrep is determined by line n of its GT patterns. The program takes line n as input and calculates exact matrix generators for gl(n) and su(n). The matrices are saved to data files, one for the generators of gl(n) and one for the generators of su(n).

The Mathematica app allows a plain ASCII version of the notebook to run with the Wolfram Language Package.[8]. Instructions for running the notebook are provided in Section 5.

2. Lie Groups GL(n) and SU(n) and Their Algebras

This section briefly describes these groups and their algebras. For more thorough treatments, see, for example, References [1] and [2].

The matrices e_{ij} $i,j=1,\cdots,n$ with entries

$$e_{ij}^{rc} \equiv \delta_i^r \delta_j^c, \tag{1}$$

with row r and column c indices $r,c=1,\cdots,n$ form a basis for the general linear algebra gl(n), a Lie algebra [1] [2] [4] [5]. The commutation relations (CR) of the algebra are

$$\left[e_{ij},e_{kl}\right]^{rc} = e_{ij}^{rs}e_{kl}^{sc} - e_{kl}^{rt}e_{ij}^{tc} = \delta_{jk}e_{il}^{rc} - \delta_{il}e_{kj}^{rc},$$

where the Kronecker delta δ_{jk} is one for equal indices $j=k$ and zero otherwise. The CRs can be written as follows

$$\left[e_{ij},e_{kl}\right] = \delta_{jk}e_{il} - \delta_{il}e_{kj}, \tag{2}$$

where the row and column indices are understood.

There are other representations of the gl(n) Lie algebra. If a set of matrices E_{ij} satisfies the CRs (2), then the collection of matrices spanned by E_{ij} forms a representation of gl(n) [1] [2].

For example, the four 3×3 matrices

$$\begin{aligned} E_{11} &= \begin{pmatrix} 2 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \; E_{12} = \begin{pmatrix} 0 & \sqrt{2} & 0 \\ 0 & 0 & \sqrt{2} \\ 0 & 0 & 0 \end{pmatrix}, \\ E_{21} &= \begin{pmatrix} 0 & 0 & 0 \\ \sqrt{2} & 0 & 0 \\ 0 & \sqrt{2} & 0 \end{pmatrix}, \; E_{22} = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 2 \end{pmatrix} \end{aligned} \tag{3}$$

satisfy the CRs for gl(2), $\left[E_{ij},E_{kl}\right] = \delta_{jk}E_{il} - \delta_{il}E_{kj}$, as is quickly shown. They are the matrices calculated by the program in Section 5 when $\{2,0\}$ is entered as input for the quantity "*irrep*". Other choices give different results. For *irrep* set to $\{2+6,0+6\} = \{8,6\}$, we get the matrices in (3) but with six times the unit matrix added to E_{11} and E_{22}. These are two of an infinite number of irreps of gl(2) [1] [2].

Let E_{ij} denote the n^2 matrix generators in a basis for an irreducible representation (irrep) of gl(n). The group elements x of the associated irrep of the Lie group GL(n) can be found with the matrix exponent. Construct the matrix $M = \sum_{i=j}^{n}\sum_{i=1}^{n}\phi_{ij}E_{ij}$ using real-valued coefficients ϕ_{ij}. The matrices spanned by the E_{ij} include M and M is an element of the irrep of the gl(n) Lie algebra. We find a group element x of GL(n)

$$x = \mathrm{e}^{M} = \mathrm{e}^{\sum_{i=j}^{n}\sum_{i=1}^{n}\phi_{ij}E_{ij}}, \tag{4}$$

where the matrix exponent is defined by its series expansion, $\exp M = \mathbf{1} + M + \cdots + M^{N}/N! + \cdots$, the unit matrix is denoted $\mathbf{1}$. The matrices E_{ij} must have real-valued entries so that the elements x of GL(n) are matrices with real-valued entries.

The Lie group SU(n) can be realized as the set of $n \times n$ unitary unimodular matrices with the usual matrix multiplication, the dot product, as the group operation. The group element x of SU(n) can be expressed as the matrix exponent of an element X of the su(n) Lie algebra. We find a group element x of SU(n),

$$x = \mathrm{e}^{iX} = \mathrm{e}^{i\sum_{j=1}^{n^2-1}\theta_j T_j}, \tag{5}$$

where the coefficients θ_j are real.

Note the presence of the factor i in (5) that is not in (4). With factor i, the matrix $X = \sum_j \theta_j T_j$ must be Hermitian for x to be unitary. Since the factor i is not universally adopted, the su(n) matrices calculated with the Mathematica notebook must be applied carefully.

The $n^2 - 1$ matrices T_j form a basis for the irrep of the su(n) Lie algebra. The matrices T_j must be traceless Hermition matrices so that the group elements x are unimodular and unitary. The presence of the factor i in the exponent of (5) lets us work with Hermitian rather than ant-Hermitian matrices in the su(n) Lie algebra.

The Gel'fand-Tsetlin (GT) formalism first finds the matrix generators E_{ij} of an irrep of gl(n) and then, with those E_{ij}, assembles Hermitian matrices by a linear transformation with complex coefficients. The Hermitian matrices form the basis for an irrep of the su(n) Lie algebra.

3. The Basis Vector Sequence Function

A matrix M acts on the vectors of a vector space V by matrix multiplication. The vector space has a basis, which is an ordered set of vectors. Some treatments use the label "states" instead of "vectors".

If a similarity transformation interchanges the first and third vectors, say, then the similarity transformation must be applied to the matrix M. That rearranges the entries in M. It follows that a part of the determination of matrices such as M is to determine the sequence of vectors in the basis of V.

To sort the basis vectors and assign each vector a place in sequence, we choose a well-known way to identify a vector with parameters. Then the parameters for a particular vector v determine the place assigned to v in the sequence. For the vector space of an irreducible representation (irrep) of gl(n), the sequence function $S(n,m)$ produces an integer S, which is the vector's place in the sequence, given the parameters m of the vector v.

Gel'fand-Tsetlin (GT) patterns parameterize the basis of the vector space for each irrep of gl(n). An order n GT pattern is a set of $n(n+1)/2$ integers, pos-

itive, negative, or zero, arranged in lines [1] [4]-[6] [7] [9].

$$\begin{array}{ccccc} m_{1n} & & m_{2n} & \cdots & m_{nn} \\ & m_{1,n-1} & \cdots & m_{n-1,n-1} & \\ & & \vdots & & \\ & & m_{11} & & \end{array} \tag{6}$$

The line k has k elements $m_{\tau k}$, $\tau \in \{1,\cdots,k\}$ and $k = 1,\cdots,n$. Successive elements in a line must obey the inequality,

$$m_{\tau+1k} \le m_{\tau k}; \tag{7}$$

they must not increase.

As an additional constraint, the integers $m_{\tau k}$ must be 'interlaced.' This is a 'betweenness' requirement that restricts the range of the elements of line k according to the elements of the next line up, line $k+1$. For the integer $m_{\tau k}$ in line k the requirement is

$$m_{\tau+1,k+1} \le m_{\tau k} \le m_{\tau,k+1}. \tag{8}$$

Thus, $m_{\tau k}$ must fit in the gap from $m_{\tau+1,k+1}$ to $m_{\tau,k+1}$ in the next higher line $k+1$. There are k gaps between $k+1$ elements.

A gl(n) irrep is determined by line n of its GT patterns. The basis vectors in an irrep of gl(n) are in a one-to-one correspondence with those GT patterns that all have a fixed, given set of integers $m_{\tau n}$, $\tau = 1,\cdots,n$. Line n is fixed while lines 1 to $n-1$ are interlaced by (8) and nonincreasing, but otherwise arbitrary. There is one GT pattern for each basis vector.

The dimension d of the irrep is the same as the number of vectors in the basis of the vector space. The dimension d_W is a well-known function, attributed to Weyl, [10] of the order n and the elements $m_{\tau n}$,

$$d_W(n,m) = \frac{\prod_{\tau=1}^{n-1}\prod_{f=\tau+1}^{n}\left(p_{\tau n} - p_{fn}\right)}{\prod_{\tau=1}^{n-1}\tau!}, \tag{9}$$

where the dependence on $m_{\tau n}$ is in the "partial hooks" $p_{\tau n}$. For the k th line of the GT pattern m, the partial hooks $p_{\tau k}$ are given by

$$p_{\tau k} = m_{\tau k} + k - \tau, \tag{10}$$

for $\tau = 1,\cdots,k$, a mixture of element $m_{\tau k}$, line number k and element number τ.

Therefore, sorting the basis of the vector space for a gl(n) irrep is equivalent to sorting the GT patterns associated with the gl(n) irrep. We deal with that problem next.

The GT patterns m for the irrep of gl(n) have a fixed line n. Consider line $n-1$. By the betweenness constraint (8), for a given τ, the element $m_{\tau,n-1}$ can have a total of $m_{\tau n} - m_{\tau+1,n} + 1$ values, the integers $m_{\tau,n-1} = m_{\tau+1,n},\cdots,m_{\tau n}$.

The set of GT patterns that have both lines n and $n-1$ fixed are a subset of the GT patterns of the gl(n) irrep with fixed line n. That subset forms the GT patterns of the gl($n-1$) irrep determined by the fixed line $n-1$. More thorough

investigation shows that the Lie algebra gl(n) is the direct sum of irreps of the Lie algebra gl($n-1$) [1] [2]. We write

$$gl(n) = \oplus_h gl_h(n-1), \tag{11}$$

where the index h ranges over the list of irreps of gl($n-1$) in the direct sum. For brevity, we often call $\mathrm{gl}_h(n-1)$ the "subalgebra h".

The rules for gl(n) apply to gl($n-1$). There is one list of elements $m^h_{\tau,n-1}$ for line $n-1$ in the irrep of $\mathrm{gl}_h(n-1)$. The basis vectors of the gl($n-1$) irrep are in one-to-one correspondence with the GT patterns of the irrep of gl(n) that has elements $m^h_{\tau,n-1}$ for line $n-1$. Thus both lines n and $n-1$ are fixed in the GT patterns for the subalgebra h.

We are led to sort the GT patterns iteratively. Let $S(n,m)$ be the place S of the GT pattern m in the sorted list of all GT patterns for a given irrep of gl(n). We have $1 \le S \le d_W(n,m)$, the dimension of the irrep.

First, the sequence function $S(2,m)$ is chosen for irreps of gl(2). Then, the sequence function $S(n,m)$ for the order n irreps of gl(n), $n \ge 3$, is defined in terms of order $n-1$ irreps in the direct sum (11). Since there are decisions such as taking larger before smaller or smaller before larger, the process can be done in many ways.

An irrep of gl(2) is determined by line 2 of its GT patterns, $m_{\tau 2} = \{m_{12}, m_{22}\}$, which is held fixed for the irrep. The first line $m_{\tau 1} = \{m_{11}\}$ must contain one of the integers $m_{11} = m_{22}, \cdots, m_{12}$, but which one is arbitrary.

Assumption 1. The sequence function $S(2,m)$ for the GT patterns m of an irrep of gl(2) is

$$S(2,m) = m_{12} - m_{11} + 1. \tag{12}$$

The largest value, $m_{11} = m_{12}$, is first, $S = 1$, and the smallest value, $m_{11} = m_{22}$, has the last place, $S = m_{12} - m_{22} + 1$.

To find the sequence function for irreps of gl(n) for $n \ge$ 3, we devise a way to define $S(n,m)$ in terms of $S(n-1,m)$.

Consider one of the GT patterns m in (6) of an irrep of gl(n). As stated above, the line n of the GT patterns is fixed. The list of integers $m_{\tau n} = \{m_{1n}, \cdots, m_{nn}\}$ is the same for all GT patterns of the irrep. Line $n-1$ of m is constrained to fit in the gaps of line n by (8).

The list of integers in line $n-1$ of m puts m in one of the subalgebra irreps, say the subalgebra h_m, *i.e.* $\mathrm{gl}_{h_m}(n-1)$. If we know the sequence function $S(n-1,m)$ of order $n-1$, then we know where m is placed in the list of GT patterns for $\mathrm{gl}_{h_m}(n-1)$.

We need to sequence the GT patterns of the other $h \ne h_m$ subalgebras. For simplicity, assume that all of the GT patterns in one subalgebra, say h_A, either precede or follow the patterns for a second subalgebra h_B, with $h_A \ne h_B$. We know that the subalgebras are determined by their fixed lines $n-1$, *i.e.* by $m^A_{i,n-1}$ and $m^B_{i,n-1}$ for $i = 1, \cdots, n-1$.

Since lines are nonincreasing, the smallest elements of their lines $n-1$ are $m^A_{n-1,n-1}$ and $m^B_{n-1,n-1}$. The two elements could be equal. Consider the smallest *distinct* elements $m^A_{k,n-1}$ and $m^B_{k,n-1}$ for some index k. Thus, when $k<n-1$, we have $m^A_{i,n-1}=m^B_{i,n-1}$ for $i=k+1,\cdots,n-1$ and, for any k, $m^A_{k,n-1}\neq m^B_{k,n-1}$. Clearly, $m^A_{k,n-1}$ must be either larger or smaller than $m^B_{k,n-1}$. We put larger before smaller.

Assumption 2. If $m^A_{k,n-1}>m^B_{k,n-1}$, then h_A precedes h_B and all of the GT patterns in subalgebra $\mathrm{gl}_{h_A}(n-1)$ precede the GT patterns in $\mathrm{gl}_{h_B}(n-1)$.

We define the sequence function $S(n,m)$ in terms of $S(n-1,m)$ in a way consistent with Assumption 2,

$$S(n,m)=S(n-1,m)+\sum_{t=1}^{n-1}\left(\sum_{\bar{m}_{1,n-1}=m_{2,n}}^{m_{1,n}}\sum_{\bar{m}_{2,n-1}=m_{3,n}}^{m_{2,n}}\cdots\sum_{\bar{m}_{t,n-1}=m_{t,n-1}+1}^{m_{t,n}} d_W(n-1,\bar{m})\Big|_{\substack{\bar{m}_{\tau,n-1}=m_{\tau,n-1},\\ \tau=t+1,\cdots,n-1}}\right). \quad (13)$$

The first term in (13) is the place $S(n-1,m)$ of the GT pattern m in the subalgebra $\mathrm{gl}_{h_m}(n-1)$. The rest of the expression sums the dimensions d_W of the subalgebra irreps $\mathrm{gl}_{h\neq h_m}(n-1)$ that precede the subalgebra $\mathrm{gl}_{h_m}(n-1)$ by Assumption 2.

With $t=n-1$ in the expression in parentheses in (13), all elements $\bar{m}_{\tau,n-1}$, $\tau=1,\cdots,n-1$, are summed over their allowed ranges except that $\bar{m}_{n-1,n-1}$ starts larger than $m_{n-1,n-1}$ at $\bar{m}_{n-1,n-1}=m_{n-1,n-1}+1$. With $t=n-2$, $\bar{m}_{n-1,n-1}$ is set to $m_{n-1,n-1}$ and all elements $\bar{m}_{\tau,n-1}$, $\tau=1,\cdots,n-2$, are summed over their allowed ranges except that $\bar{m}_{n-2,n-1}$ starts larger than $m_{n-2,n-1}$ at $m_{n-2,n-1}+1$. As t decreases, the index k decreases of the smallest *distinct* element $\bar{m}_{k,n-1}\geq m_{k,n-1}+1$, more elements are equal $\bar{m}_{i,n-1}=m_{i,n-1}$, $i=k+1,\cdots,n-1$, and fewer elements $\bar{m}_{\tau,n-1}$, $\tau\leq k-1$, are summed over all allowed values, in agreement with the discussion leading to Assumption 2.

Theorem 1. The sequence function $S(n,m)$ is defined for orders $n\geq 2$.

Proof. By (13), the sequence function $S(n,m)$ is a function of $S(n-1,m)$ and by (12), $S(2,m)$ is known. □

The sequence function $S(n,m)$ maps the GT patterns m for an irrep of gl(n) to the ordered set of integers, $S\in\{1,\cdots,d\}$, where $d=d_W(n,m)$ is the dimension (9) of the irrep. Inversely, for each integer S in the range 1 to d, there is a GT pattern m.

A matrix generator E_{ij} has entries E^{rc}_{ij} with a row index r and a column index c, each running over the integers in the range 1 to d. Therefore, the row r and the column c have associated GT patterns that we label mr and mc, respectively. The GT pattern mr gives r and mc gives c,

$$r=S(n,mr) \text{ and } c=S(n,mc). \quad (14)$$

where $S(n,m)$ is the sequence function in (12) and (13).

4. Matrix Formulas

The nonzero entry E^{rc}_{ij} in the matrix E_{ij} at the intersection of the row r and the column c can be found with the Louck-Biedenharn extension of the original

Gel'fand-Tsetlin (GT) formulas [4] [6] [7]. The formulas depend on the GT patterns mr and mc, introduced in Section 3, which determine r and c by (14).

A point of terminology. By "nonzero" entries we mean the entries E_{ij}^{rc} found with the formulas. This ignores the occasions when a formula yields the value zero. All entries other than nonzero entries vanish.

The matrices E_{ii}^{rc} are diagonal, so $r=c$ for their nonzero entries. Hence, the GT patterns for rows and columns are equal, $mr=mc$. The formulas for the nonzero entries E_{ii}^{cc} are

$$E_{11}^{cc}=m_{11};\quad E_{ii}^{cc}=\sum_{\tau=1}^{i}mc_{\tau i}-\sum_{\tau=1}^{i-1}mc_{\tau,i-1}\ \ (i\geq 2), \tag{15}$$

no sum over i or c.

There are also formulas for the upper triangulars E_{ij} which have $i<j$. Nonzero entries occur in multiple rows r_a in each column c. The label a indexes the rows with nonzero entries. Thus, in addition to a formula for the entry of the matrix $E_{ij}^{r_ac}$ in the row r_a in the column c, there are "row rules" to find the GT patterns mr_a given the GT pattern mc. Next, we consider the row rules [7].

The GT pattern mr_a differs from the GT pattern mc by one element per line for the lines k, $k=i,\cdots,j-1$. Each GT pattern mr_a differs from mc by adding one to just one element τ_k per line k. We have

$$\left(mr_a\right)_{\tau_k k}=mc_{\tau_k k}+1, \tag{16}$$

where exactly one element $\tau_k\in\{1,\cdots,k\}$ changes for each line k, with $k=i,\cdots,j-1$.

Thus, each GT pattern mr_a is associated with an ordered set of $j-i$ integers $\{\tau_i,\tau_{i+1},\cdots,\tau_{j-1}\}$, one for each line $k=i,\cdots,j-1$. We have

$$mr_a\leftrightarrow\{\tau_i,\tau_{i+1},\cdots,\tau_{j-1}\}, \tag{17}$$

where the integers τ_k range from 1 to k for $k=i,\cdots,j-1$.

For the first line $k=i$ there are i possible choices for the element τ_i, for the second line $k=i+1$ there are $i+1$ choices and so on up to the final line $k=j-1$ with $j-1$ choices for τ_{j-1}. It follows that the number N_r of distinct GT patterns mr_a is the product of the number of choices for each line,

$$N_r=i(i+1)\cdots(j-1)=(j-1)!/(i-1)!. \tag{18}$$

By (16), (17) and (18), the nonzero entries in column $c=S(n,mc)$ of E_{ij} occur at rows $r_a=S(n,mr_a)$, $a=1,\cdots,N_r$.

Next, we introduce the formulas for the upper triangular generators E_{ij} with $i<j$. For an entry $E_{ij}^{r_ac}$, the formula can be split into three factors,

$$E_{ij}^{r_ac}=f_1f_2f_3, \tag{19}$$

which we take from [7],

$$f_1(mc,mr_a)=\left[-\frac{\prod_{s=1}^{j}\left(p_{sj}-p_{\tau_{i-1},j-1}-1\right)}{\prod_{s=1,s\neq\tau_{j-1}}^{j-1}\left(p_{s,j-1}-p_{\tau_{j-1},j-1}-1\right)}\right]^{1/2}, \tag{20}$$

$$f_2(mc,mr_a)=\prod_{k=i+1}^{j-1}\left\{\operatorname{sign}(\tau_{k-1}-\tau_k+0.1)\right.$$
$$\left.\left[\prod_{s=1,s\neq\tau_{k-1}}^{k-1}\frac{(p_{s,k-1}-p_{\tau_k,k})}{(p_{s,k-1}-p_{\tau_{k-1},k-1}-1)}\prod_{s=1,s\neq\tau_k}^{k}\frac{(p_{s,k}-p_{\tau_{k-1},k-1}-1)}{(p_{s,k}-p_{\tau_k,k})}\right]^{1/2}\right\},\tag{21}$$

$$f_3(mc,mr_a)=\operatorname{sign}(i-\tau_i+0.1)\left[\frac{\prod_{s=1}^{i-1}(p_{s,i-1}-p_{\tau_i i})}{\prod_{s=1,s\neq\tau_i}^{i}(p_{s,i}-p_{\tau_i i})}\right]^{1/2},\tag{22}$$

where the 0.1 in the sign function makes the sign of 0 equal to +1, $p_{\tau k}$ depends on mc, $p_{\tau k}=mc_{\tau k}+k-\tau$, which is $m\to mc$ in (10) and, by (16) and (17), the various integers τ depend on the row rules for r_a.

Finally, the lower triangular generators E_{ij} that have $i>j$ are

$$E_{ij}^{rc}=E_{ji}^{cr},\ i>j,\tag{23}$$

which are the transposes of the upper triangular generators E_{ij} determined by (19) to (22).

This completes the set of formulas for the matrix generators E_{ij}, where $i,j=1,\cdots,n$. The matrix generators form a basis of an irreducible representation (irrep) of the gl(n) Lie algebra, which is verified for each irrep calculated in the Mathematica notebook in Section 5.

For the su(n) Lie algebra, a basis of matrix generators T_i, where $i=1,\cdots,n^2-1$, can be written in terms of the matrices E_{ij} found for gl(n). The matrices must be Hermitian, traceless, and satisfy the commutation relations of the su(n) Lie algebra [1]-[3].

For $i\neq j$, the matrices E_{ij} and E_{ji} are the transposes of each other, so the matrices $\lambda^{\pm}=E_{ij}\pm iE_{ji}$ are their own Hermitian conjugates, $\lambda^{\pm\dagger}=\lambda^{\pm}$, where $\dagger$ indicates the transpose of the matrix with its entries replaced by their complex conjugates. The diagonal generators E_{ii}, no sum over i, are real diagonal matrices with nonzero entries E_{ii}^{rr}, no sum over r, and, therefore, Hermitian. Then, following custom, the generators T_i are rescaled by a factor of 1/2 applied to the λ_i, *i.e.* $T_i=\lambda_i/2$.

The structure constants f_{ijk} for the CRs of su(n) are found by direct calculation from the fundamental representation. The calculated matrices for each irrep are verified by substituting them into the CRs of the su(n) Lie algebra.

5. Program

The Mathematica Notebook *Matrix Generators for Lie Algebras gl*(n) *and su*(n)*, the Mathematica Notebook* implements the sequence function of Section 3 and the matrix formulas of Section 4 to produce matrix generators E_{ij}, $i,j=1,\cdots,n$ for an irreducible representation (irrep) of the Lie algebra gl(n) and the matrix generators T_i for an irrep of Lie algebra su(n). The order n of the algebras is at most $n=6$, $n=2,3,4,5,6$.

The notebook can be run by copying it into a blank Mathematica notebook per

the following instructions:

- Open the Mathematica app and create a new blank Mathematica notebook.
- Copy the notebook from this section and paste it into the blank notebook.
- Under "File" use the "Save As..." option to save the notebook with "Save as type:" Wolfram Language Package, which gives the notebook file a ".wl" extension.
- At the beginning of the notebook, find the quantity "*irrep*". Make the list *irrep* equal to the list of elements m_{in}, $i=1,\cdots,n$, of line n of the GT patterns for the desired irrep.
- Run the cells in the notebook using the keyboard combination "Shift-Enter."

Note that the elements of the list *irrep* are the numbers of boxes per row in a semistandard Young tableau [1] [9].

The program calculates the matrices E_{ij} that form a basis for an irrep of the gl(n) Lie algebra and the matrices T_i that form a basis for an irrep of the su(n) Lie algebra. The resulting matrix entries are exact numerical expressions, not approximate decimal values. The program creates data files to store the matrices.

Description of the records in a gl(n) data file:

1. {n, *irrep* list, the dimension of the matrices, the total number of nonzero entries, the CPU time used in seconds }

2, 3... $\left\{i,j,r,c,E_{ij}^{rc}\right\}$ (nonzero entries only)

Description of the records in an su(n) data file:

1. {n, *irrep* list, the dimension of the matrices, the total number of nonzero entries, the CPU time used in seconds}

2, 3... $\left\{i,r,c,T_i^{rc}\right\}$ (nonzero entries only)

For example, running the notebook with the input $irrep=m_{\tau n}=\{2,0\}$ produces two data files, gl2irrep-2-0-gl2.dat for gl(2) and su2irrep-2-0-su2.dat for su(2). The 3×3 matrix generators (3) for $irrep=m_{\tau n}=\{2,0\}$, an irrep of gl(2), can be reconstructed from its data file gl2irrep-2-0-gl2.dat that we copy here:

$$\begin{aligned}&\{\{2,\{2,0\},3,10,0\},\\&\{1,2,2,3,\sqrt{2}\},\{2,1,3,2,\sqrt{2}\},\{1,2,1,2,\sqrt{2}\},\\&\{2,1,2,1,\sqrt{2}\},\{1,1,3,3,0\},\{1,1,2,2,1\},\\&\{1,1,1,1,2\},\{2,2,3,3,2\},\{2,2,2,2,1\},\{2,2,1,1,0\}\}.\end{aligned}$$

Transforming the 3×3 matrix generators (3) to a basis for su(2) produces three 3×3 generators T_i, $i=1,2,3$. The T_i can be reconstructed from its data file su2irrep-2-0-su2.dat. A copy of the data file is as follows,

$$\begin{aligned}&\{\{2,\{2,0\},3,10,0\},\\&\{1,1,2,1/\sqrt{2}\},\{1,2,1,1/\sqrt{2}\},\{1,2,3,1/\sqrt{2}\},\\&\{1,3,2,1/\sqrt{2}\},\{2,1,2,(-\mathrm{i})/\sqrt{2}\},\{2,2,1,\mathrm{i}/\sqrt{2}\},\\&\{2,2,3,(-\mathrm{i})/\sqrt{2}\},\{2,3,2,\mathrm{i}/\sqrt{2}\},\{3,1,1,1\},\{3,3,3,-1\}\}.\end{aligned}$$

The Mathematica notebook in plain ASCII format:

The Mathematica notebook in plain ASCII format:

(*Notebook Settings*)(*font:Consolas, Regular, 10 point*) SetOptions[EvaluationNotebook[], ShowCellLabel -> False]; SetOptions[EvaluationNotebook[], PageWidth -> 500]; (*Limit memory usage:*)

$HistoryLength = 0;

(* Matrix Generators for Lie Algebras gl(n) and su(n), \ the Mathematica Notebook

Richard Shurtleff

Wentworth Institute of Technology,Boston,USA Email:shurtleffr@wit.edu, momentum.matrix@gmail.com*)

(*This notebook calculates matrix bases for irreducible representation\ s (irreps) of the Lie algebras gl(n) and su(n) of order n=2,3,4,5,6.

The calculations in this notebook implement work by Louck and Biedenha\ rn that extends the formulas by Gel'fand-Tsetlin.[1,2,3,4]

*) (*Instructions:

Set the list "irrep" in the cell following Options to obtain the desir\ ed irrep.

Run the cells in order with Shift+Enter.

*)

(*USER:

Please supply 2 to 6 non-increasing integers.*) irrep = {2, 0};

(*irrep: the nth row m[i,n] of the GT pattern m[i,k].*) (*irrep: the number of boxes per row of a Young tableau.*)

(* irrep determines the matrix bases of the gl(n) and su(n) irreps.*)

(*Application:Mathematica 15.0[5] Platform:Microsoft Windows 11 (64-bit) File Size:170 kbytes

Max Memory Used In Session: 300 Mbytes Updated Date:September 29,2026

This file:matrixBasesForGLn-SUn4Article1.wl*) Print["Today's Date: ", Now]

nonIncreasing =

Union[Table[irrep[[i]] >= irrep[[i + 1]], {i, Length[irrep] - 1}]]; integers = Union[Table[IntegerQ[irrep[[i]]], {i, Length[irrep]}]]; Print["The supplied irrep list has integers: ", integers, ", that are \ nonincreasing: ", nonIncreasing, "."]

(*1. GT pattern functions [1,2,3,4,6] and vector sequence functions*) (* Definitions

\[Delta]ij[i,j]: the Kronecker delta, \[Delta]ij = 1 when i = j and\

\[Delta]ij = 0 otherwise.

```
m[i,k]: the Subscript[m, ik] element of a GT pattern for gl(n) pik[m,i,k]: partial hook Subscript[p, ik] =
Subscript[m, ik]+k-i . *)

(*              betweenQ[n,k,m]: the betweenness test for GT patterns,

                e.g. m22<=m11<=m12, m33<=m22<=m23, m23<=m12<=m13.

          dimW[n,mm]: the Weyl dimension formula[7] gives the number of rows and columns in the matrices

             sSnm[n,m]: the sequence function that produces an integer from

              1 to dimW for each allowed GT pattern m of gl(n)*)

\[Delta]ij[i_, j_] := \[Delta]ij[i, j] = KroneckerDelta[i, j]; betweenQ[n_, k_, mm_] :=

              Union[Flatten[

  Table[mm[i + 1, k + 1] <= mm[i, k] <= mm[i, k + 1], {i, k}]]]; pik[mm_, i_, k_] := mm[i, k] + k - i

dimW[n_, mm_] :=

              Product[(pik[mm, \[Tau], n] - pik[mm, f, n]), {\[Tau], 1,

    n - 1}, {f, \[Tau] + 1, n}]/Product[\[Tau]!, {\[Tau], n - 1}]; sSnm[nn_, mm_] := (mm[1, 2] - mm[1, 1] +
1)/; nn == 2;

       sSnm[nn_, mm_] := Simplify[

                Sum[dimW[2, mBAR], {mBAR[1, 2], mm[2, 3], mm[1, 3]}, {mBAR[2, 2],

                  mm[2, 2] + 1, mm[2, 3]}]] +

              Simplify[

                Sum[dimW[2, mBAR] /. {mBAR[2, 2] -> mm[2, 2]}, {mBAR[1, 2],

      mm[1, 2] + 1, mm[1, 3]}]] + sSnm[2, mm] /; nn == 3; sSnm[nn_, mm_] :=

              Simplify[

                Sum[dimW[nn - 1, mBAR], {mBAR[1, 3], mm[2, 4],

                  mm[1, 4]}, {mBAR[2, 3], mm[3, 4], mm[2, 4]}, {mBAR[3, 3],

                  mm[3, 3] + 1, mm[3, 4]}]] +

              Simplify[

                Sum[dimW[nn - 1, mBAR] /. {mBAR[3, 3] -> mm[3, 3]}, {mBAR[1, 3],

                  mm[2, 4], mm[1, 4]}, {mBAR[2, 3], mm[2, 3] + 1, mm[2, 4]}]] +

              Simplify[

                Sum[dimW[nn - 1, mBAR] /. {mBAR[3, 3] -> mm[3, 3],

                  mBAR[2, 3] -> mm[2, 3]}, {mBAR[1, 3], mm[1, 3] + 1,

                  mm[1, 4]}]] + sSnm[nn - 1, mm] /; nn == 4;
```

```
sSnm[nn_, mm_] := Simplify[
    Sum[dimW[nn - 1, mBAR], {mBAR[1, 4], mm[2, 5],
      mm[1, 5]}, {mBAR[2, 4], mm[3, 5], mm[2, 5]}, {mBAR[3, 4],
      mm[4, 5], mm[3, 5]}, {mBAR[4, 4], mm[4, 4] + 1, mm[4, 5]}]] +
   Simplify[
    Sum[dimW[nn - 1, mBAR] /. {mBAR[4, 4] -> mm[4, 4]}, {mBAR[1, 4],
      mm[2, 5], mm[1, 5]}, {mBAR[2, 4], mm[3, 5],
      mm[2, 5]}, {mBAR[3, 4], mm[3, 4] + 1, mm[3, 5]}]] +
   Simplify[
                Sum[dimW[nn - 1, mBAR] /. {mBAR[4, 4] -> mm[4, 4],
                    mBAR[3, 4] -> mm[3, 4]}, {mBAR[1, 4], mm[2, 5],
               mm[1, 5]}, {mBAR[2, 4], mm[2, 4] + 1, mm[2, 5]}]] +
     Simplify[
          Sum[dimW[nn - 1, mBAR] /. {mBAR[4, 4] -> mm[4, 4],
              mBAR[3, 4] -> mm[3, 4], mBAR[2, 4] -> mm[2, 4]}, {mBAR[1, 4],
              mm[1, 4] + 1, mm[1, 5]}]] + sSnm[4, mm] /; nn == 5;
sSnm[nn_, mm_] := Simplify[
    Sum[dimW[nn - 1, mBAR], {mBAR[1, 5], mm[2, 6],
      mm[1, 6]}, {mBAR[2, 5], mm[3, 6], mm[2, 6]}, {mBAR[3, 5],
      mm[4, 6], mm[3, 6]}, {mBAR[4, 5], mm[5, 6],
      mm[4, 6]}, {mBAR[5, 5], mm[5, 5] + 1, mm[5, 6]}]] +
   Simplify[
    Sum[dimW[nn - 1, mBAR] /. {mBAR[5, 5] -> mm[5, 5]}, {mBAR[1, 5],
      mm[2, 6], mm[1, 6]}, {mBAR[2, 5], mm[3, 6],
      mm[2, 6]}, {mBAR[3, 5], mm[4, 6], mm[3, 6]}, {mBAR[4, 5],
      mm[4, 5] + 1, mm[4, 6]}]] +
   Simplify[
                Sum[dimW[nn - 1, mBAR] /. {mBAR[5, 5] -> mm[5, 5],
                    mBAR[4, 5] -> mm[4, 5]}, {mBAR[1, 5], mm[2, 6],
      mm[1, 6]}, {mBAR[2, 5], mm[3, 6], mm[2, 6]}, {mBAR[3, 5],
      mm[3, 5] + 1, mm[3, 6]}]] +
```

```
Simplify[
 Sum[dimW[nn - 1, mBAR] /. {mBAR[5, 5] -> mm[5, 5],
    mBAR[4, 5] -> mm[4, 5], mBAR[3, 5] -> mm[3, 5]}, {mBAR[1, 5],
   mm[2, 6], mm[1, 6]}, {mBAR[2, 5], mm[2, 5] + 1, mm[2, 6]}]] +
Simplify[
          Sum[dimW[nn - 1, mBAR] /. {mBAR[5, 5] -> mm[5, 5],
             mBAR[4, 5] -> mm[4, 5], mBAR[3, 5] -> mm[3, 5],
    mBAR[2, 5] -> mm[2, 5]}, {mBAR[1, 5], mm[1, 5] + 1,
   mm[1, 6]}]] + sSnm[5, mm] /; nn == 6;
```

(*2. GT patterns for row r and column c indices

The GT pattern mr for row r differs from the column pattern mc, \ the default,by at most one place \[Tau] in each GT pattern row k.*) (* Definitions

\[Tau]4mr: the place \[Tau], 1<=\[Tau]<=k, \

in row k that gets boosted from c to r, mik(r) = \ mik(c)+1. At most one place \[Tau] per GT row k gets

boosted from c to r.

Each Tuple in the table \[Tau]4mr designates a row ra in column c

*)

```
\[Tau]4mr[n_, i_, j_] := Tuples[Table[Range[k], {k, i, j - 1}]]; (*3. The su(6) structure constants fijk
```

[4,7]

The generators Ti for i=1,...,n^2-1 form a basis for su(n).

This implies that the set of structure constants fijk for n=6 has subsets for su(n) for n<6. The structure constants for n'=2 are

fijk with i,j,k=1,2,3. Likewise, the structure constants for n'<6 are fijk with i,j,k<=n'^2-1.Thus, the structure constants for n=6

gives us the fijk for su(n) with n=2,3,4,5,6*) (* Definitions

eij[i,j]: The matrix with a one in the ith row of the jth column and \

all other entries zero. The eij form a basis for the generators of the fundamental irreducible representation(irrep) of gl(n).

\[Lambda]0[i]: matrix generators for the fundamental rep of su(6), i = 1,...,35. The \[Lambda]0[\

i] can be used to get the structure constants fijk for n = 2,3,4,5,6.

mT0[i]: \[Lambda]0[i]/2, the conventional factor 1/2 applied to the\

```
        \[Lambda]i gives the Ti
fijk: structure constants [Ti,Tj] = \
\[ImaginaryI] \[CapitalSigma] fijk Tk, sum over k.
    *)

(*The fundamental irrep of gl(n):*) eij[n_, i_, j_] :=
        Table[\[Delta]ij[i, r] \[Delta]ij[j, c], {r, n}, {c, n}]

n = 6;(*Transform to the basis of su(n)*)
\[Lambda]0[1] = eij[n, 1, 2] + eij[n, 2, 1]; \[Lambda]0[2] =
        I (-eij[n, 1, 2] + eij[n, 2, 1]); \[Lambda]0[3] =
        eij[n, 1, 1] - eij[n, 2, 2];
\[Lambda]0[4] = eij[n, 1, 3] + eij[n, 3, 1]; \[Lambda]0[5] =
        I (-eij[n, 1, 3] + eij[n, 3, 1]);
\[Lambda]0[6] = eij[n, 2, 3] + eij[n, 3, 2]; \[Lambda]0[7] =
        I (-eij[n, 2, 3] + eij[n, 3, 2]);
\[Lambda]0[
          8] = (eij[n, 1, 1] + eij[n, 2, 2] - 2 eij[n, 3, 3]) (1/3)^(1/2);
\[Lambda]0[9] = eij[n, 1, 4] + eij[n, 4, 1]; \[Lambda]0[10] =
        I (-eij[n, 1, 4] + eij[n, 4, 1]);
\[Lambda]0[11] = eij[n, 2, 4] + eij[n, 4, 2]; \[Lambda]0[12] =
        I (-eij[n, 2, 4] + eij[n, 4, 2]);
\[Lambda]0[13] = eij[n, 3, 4] + eij[n, 4, 3]; \[Lambda]0[14] =
        I (-eij[n, 3, 4] + eij[n, 4, 3]);
\[Lambda]0[
          15] = (eij[n, 1, 1] + eij[n, 2, 2] + eij[n, 3, 3] -
            3 eij[n, 4, 4]) (1/6)^(1/2);


    \[Lambda]0[16] = eij[n, 1, 5] + eij[n, 5, 1]; \[Lambda]0[17] =
      I (-eij[n, 1, 5] + eij[n, 5, 1]);
    \[Lambda]0[18] = eij[n, 2, 5] + eij[n, 5, 2]; \[Lambda]0[19] =
      I (-eij[n, 2, 5] + eij[n, 5, 2]);
    \[Lambda]0[20] = eij[n, 3, 5] + eij[n, 5, 3]; \[Lambda]0[21] =
```

```
    I (-eij[n, 3,   5] + eij[n,   5,  3]);
\[Lambda]0[22] = eij[n, 4,   5] + eij[n,   5,  4]; \[Lambda]0[23] =
    I (-eij[n, 4,   5] + eij[n,   5,  4]);

\[Lambda]0[
    24] = (eij[n, 1, 1] + eij[n, 2, 2] + eij[n, 3, 3] + eij[n, 4, 4] -
      4 eij[n, 5, 5]) (1/10)^(1/2);

\[Lambda]0[25] = eij[n, 1,   6] + eij[n,   6,  1]; \[Lambda]0[26] =
    I (-eij[n, 1,   6] + eij[n,   6,  1]);
\[Lambda]0[27] = eij[n, 2,   6] + eij[n,   6,  2]; \[Lambda]0[28] =
    I (-eij[n, 2,   6] + eij[n,   6,  2]);
\[Lambda]0[29] = eij[n, 3,   6] + eij[n,   6,  3]; \[Lambda]0[30] =
    I (-eij[n, 3,   6] + eij[n,   6,  3]);
\[Lambda]0[31] = eij[n, 4,   6] + eij[n,   6,  4]; \[Lambda]0[32] =
    I (-eij[n, 4,   6] + eij[n,   6,  4]);

\[Lambda]0[33] = eij[n, 5, 6] + eij[n, 6, 5]; \[Lambda]0[34] =
    I (-eij[n, 5, 6] + eij[n, 6, 5]);
\[Lambda]0[
    35] = (eij[n, 1, 1] + eij[n, 2, 2] + eij[n, 3, 3] + eij[n, 4, 4] +
                eij[n, 5, 5] - 5 eij[n, 6, 6]) (1/15)^(1/2);
            Table[{i, \[Lambda]0[i] // MatrixForm}, {i, 35}];

Table[mT0[i] = \[Lambda]0[i]/2, {i, 35}]; fijk[i_, j_, k_] :=
    fijk[i, j, k] = -2 I Tr[(mT0[i] . mT0[j] - mT0[j] . mT0[i]) . mT0[k] ]

(*4. Formulas for the matrix generators Eij [8]*) (* Definitions
      sRange1a[n,i,j,ra], sRange2a, sRange2b, sRange3a: Restricted ranges of indices
    s in the functions Eijrc.
  factor1sqroot[n,i,j,ra,m,\[Tau]], factor2sqroot, factor3sqroot: factors in the formulas for Eijrc.
    Here m is mc, \
the GT pattern for column c. \
  Index ra and \[Tau] determine the GT pattern for rows ra with non-
    zero entries Eijrc.
Eijrc[n,i,j,ra,m,\[Tau]]: the formula for the matrix element E_ij^rc, \
```

```
                the row r column c component of the E_ij generator.

        *)

Clear[sRange1a, sRange2a, sRange2b, sRange3a]; sRange1a[n_, i_, j_, ra_] :=

            sRange1a[n, i, j, ra] =

Complement[Range[j - 1], {\[Tau]4mr[n, i, j][[ra, j - i]]}]; sRange2a[n_, i_, j_, k_, ra_] :=

            sRange2a[n, i, j, k, ra] =

             Complement[Range[k - 1], {\[Tau]4mr[n, i, j][[ra, k - i]]}];

      sRange2b[n_, i_, j_, k_, ra_] := sRange2b[n, i, j, k, ra] =

Complement[Range[k], {\[Tau]4mr[n, i, j][[ra, k - i + 1]]}]; sRange3a[n_, i_, j_, k_, ra_] :=

            sRange3a[n, i, j, k, ra] =

             Complement[Range[i], {\[Tau]4mr[n, i, j][[ra, 1]]}];

factor1sqroot[n_, i_, j_, ra_,

            m_, \[Tau]_] := ((-1) (Product[(pik[m, s, j] -

                  pik[m, \[Tau]4mr[n, i, j][[ra, j - i]] , j - 1] - 1), {s, 1, j}]/Product[(pik[m, s, j - 1] -

                pik[m, \[Tau]4mr[n, i, j][[ra, j - i]] , j - 1] - 1), {s, sRange1a[n, i, j, ra]}]))^(1/2);

      factor2sqroot[n_, i_, j_, ra_, m_, \[Tau]_] := Product[Sign[\[Tau][k - 1] - \[Tau][k]
        +

            0.1] (Product[(pik[m, s, k - 1] -

              pik[m, \[Tau]4mr[n, i, j][[ra, k - i + 1]] , k])/(pik[m, s, k - 1] -

              pik[m, \[Tau]4mr[n, i, j][[ra, k - i]] , k - 1] - 1), {s, sRange2a[n, i, j, k,

                ra]}] Product[(pik[m, s, k] -

                  pik[m, \[Tau]4mr[n, i, j][[ra, k - i]] , k - 1] - 1)/(pik[m, s, k] -

              pik[m, \[Tau]4mr[n, i, j][[ra, k - i + 1]]                    , k]), {s, sRange2b[n, i,
            j, k, ra]}])^(1/2)

          , {k, i + 1, j - 1}];

      factor3sqroot[n_, i_, j_, ra_, m_, \[Tau]_] := Product[(*+1*)(+1) Sign[

                              i - \[Tau]4mr[n, i, j][[ra, +1]] +

                               0.1] (Product[(pik[m, s, i - 1] -

            pik[m, \[Tau]4mr[n, i, j][[ra, +1]] , i]), {s, i - 1}]/ Product[(pik[m, s, i] -

            pik[m, \[Tau]4mr[n, i, j][[ra, +1]]                    , i]), {s, sRange3a[n, i, j,
          k, ra]}])^(1/2)
```

```
                , {k, i, i}];
(*The off-diagonal generators Eij, i != j:*) Eijrc[n_, i_, j_, ra_, m_, \[Tau]_] :=
        factor1sqroot[n, i, j, ra, m, \[Tau]]* factor2sqroot[n, i, j, ra, m,
          \[Tau]]* factor3sqroot[n, i, j, ra, m, \[Tau]];
(*The diagonal generators Eii:*) Eiicc[n_, i_, mc_] :=
          Sum[mc[jj, i], {jj, i}] - Sum[mc[jj, i - 1], {jj, i - 1}];
(*5. Calculate the gl(n) and su(n) irreps*) (*Definitions:
        glnpassQ, sunpassQ:
            These log the success or failure of the calculated Eij and Ti matrices to satisfy the
        CRs of the gl or su algebra.
```

irrep: the n integers that characterize the gl, su irreps. It makes the nth row of the GT pattern mc for the column index c irrepsubs[n,m]: \

substitute irrep into the nth row of the a GT pattern m*)

```
        (*
```

mc[i,k]: The GT pattern for the column index c, a GT pattern

```
            with 8 (>6) rows. The nth row has the n integers that characterize the irrep.
        Lower rows k<n are variables, higher rows are made compatible with Do loop-
        ing. *)
        (* dimREP: dimension of the irrep, the number of columns c and rows r \ in an Eij or Ti matrix
        not0Eij: the nonzero components calculated for the n^2 matrix generators Eij.
```

xMmatrix1: a dummy matrix in the calculations of the Eij *)

```
        (*    r,c: row r and column c indices, 1<= r,c <=dimREP mEij0[i,j,ra]: the ra^th
        matrix, \
```

a partial result that contributes to Eij

mEij: the matrix mEij = Subscript[\[CapitalSigma], \ ra]mEij0 is a generator in the basis of

```
            the gl(n) irrep
            EijrcA0[i,j,r,c]: an intermediate table that assists converting the downloaded data to EijA
        matrices
            EijA[i,j] the downloaded generator Eij *)
tnow1 = AbsoluteTime[]; t1 = TimeUsed[];(*times*) n = Length[irrep];
t0 = TimeUsed[];
```

```
irrepsubs[n_, m_] := Table[m[i, n] -> irrep[[i]], {i, n}]; dimREP = dimW[n, mm] /. ir-
repsubs[n, mm];

Table[mEij[i, j] = Table[0, {r, dimREP}, {c, dimREP}], {i, 6}, {j, 6}]; (*Calculate Eijrc = Subscript[E, ij]^rc for i<j
and j<i*)

(*mc[i,k]. Build a GT pattern with 8 rows, 2 more than the max n = 6. \

        *)
(*mc[i,k] has extra rows to accommodate some lines of code.*) Clear[mc];

Table[mc[i, n] = irrep[[i]], {i, n}];

Table[Table[mc[i + 1, nn] = mc[i, nn - 1], {i, nn}], {nn, n + 1, 8}];

Table[mc[1, nn] = mc[1, nn - 1], {nn, n + 1, 8}];

(*Table[Print[Table[mc[i,k],{i,k}]],{k,8}];*)

(*Calculate off-diagonal Subscript[E, ij]^rc with i!=j.*) not0Eij = {};

    (*gl*)Do[(*i*)Do[(*j*) Do[(*ra*)

        Table[\[Tau][nk] = -999, {nk, n}];

      Table[\[Tau][i + nk - 1] = \[Tau]4mr[n, i, j][[ra, nk]], {nk, j - i}];

        xMmatrix1 = Table[0, {i, dimREP}, {j, dimREP}]; xMmatrixT1 = Table[0, {i,
      dimREP}, {j, dimREP}]; Catch[Do[(*mc[6,6]*)Do[(*mc[5,6]*)

          Do[(*mc[4,6]*)Do[(*mc[3,6]*)Do[(*mc[2,6]*)

            Do[(*mc[1,6]*)

             Do[(*mc[5,5]*)

              Do[(*mc[4,5]*)

               Do[(*mc[3,5]*)

                Do[(*mc[2,5]*)

                 Do[(*mc[1,5]*)

                  Do[(*mc[4,4]*)

                   Do[(*mc[3,4]*)

                    Do[(*mc[2,4]*)

                     Do[(*mc[1,4]*)

                      Do[(*mc[3,3]*)

                       Do[(*mc[2,3]*)          Do[(*mc[1,3]*)Do[(*mc[2,2]*)Do[(*mc[1,2]*)
                  Do[(*mc[1,1]*) c = sSnm[n, mc];

                      Table[If[\[Tau][k] > 0, Table[mr[i, k] =
```

```
        mc[i, k] + \[Delta]ij[i, \[Tau][k]], {i, k}],
        Table[mr[i, k] = mc[i, k], {i, k}]], {k, n}]; r = sSnm[n, mr];
        If[Union[ Flatten[Table[
        betweenQ[n, k, mr], {k, n - 1}]]] == {True}, "that is good", Con-
      tinue[]];
        xMmatrix1[[r, c]] =
        Eijrc[n, i, j, ra, mc, \[Tau]];(*Transpose:*) xMmatrixT1[[c, r]] =
        Eijrc[n, i, j, ra, mc, \[Tau]];
        AppendTo[
        not0Eij, {i, j, r, c,
        Eijrc[n, i, j, ra, mc, \[Tau]]}]; AppendTo[
        not0Eij, {j, i, c, r,
        Eijrc[n, i, j, ra, mc, \[Tau]]}], {mc[1, 1], mc[2, 2], mc[1, 2]}];
        If[n < 3, Throw["Done"]], {mc[1, 2], mc[2, 3],
        mc[1, 3]}], {mc[2, 2], mc[3, 3], mc[2, 3]}];
        If[n < 4, Throw["Done"]], {mc[1, 3], mc[2, 4],
        mc[1, 4]}], {mc[2, 3], mc[3, 4], mc[2, 4]}], {mc[
        3, 3], mc[4, 4], mc[3, 4]}];
        If[n < 5, Throw["Done"]], {mc[1, 4], mc[2, 5],
        mc[1, 5]}], {mc[2, 4], mc[3, 5], mc[2, 5]}], {mc[
        3, 4], mc[4, 5], mc[3, 5]}], {mc[4, 4], mc[5, 5],
      mc[4, 5]}];
    If[n < 6, Throw["Done"]], {mc[1, 5], mc[2, 6],
     mc[1, 6]}], {mc[2, 5], mc[3, 6], mc[2, 6]}], {mc[3,
    5], mc[4, 6], mc[3, 6]}], {mc[4, 5], mc[5, 6],
  mc[4, 6]}], {mc[5, 5], mc[6, 6], mc[5, 6]}];
If[n < 7,
 Throw["Done"]], {mc[1, 6], mc[2, 7], mc[1, 7]}], {mc[2, 6],
 mc[3, 7], mc[2, 7]}], {mc[3, 6], mc[4, 7],
mc[3, 7]}], {mc[4, 6], mc[5, 7], mc[4, 7]}], {mc[5, 6],
```

```
                                        mc[6, 7], mc[5, 7]}], {mc[6, 6], mc[7, 7], mc[6, 7]}]];
                                    mEij0[i, j, ra] = xMmatrix1; mEijT0[i, j, ra] = xMmatrixT1,
                {ra, Length[\[Tau]4mr[n, i, j]]}];
              mEij[i, j] = Sum[mEij0[i, j, ra], {ra, Length[\[Tau]4mr[n, i, j]]}]; mEij[j, i] =
               Sum[mEijT0[i, j, ra], {ra, Length[\[Tau]4mr[n, i, j]]}], {j, i + 1, n}], {i, n - 1}];
(*Calculate diagonal Subscript[E, ij]^rc for i=j.*) Table[mc[i, n] = irrep[[i]], {i,
n}];(*Print["{n,mc[i,n]}= ",{n,Table[mc[i,n],{i,n}]}];*) Do[(*i*)
          xMmatrix1 = Table[0, {i, dimREP}, {j, dimREP}];
    Catch[Do[(*mc[6,6]*)Do[(*mc[5,6]*)Do[(*mc[4,6]*)Do[(*mc[3,6]*)
        Do[(*mc[2,6]*) Do[(*mc[1,6]*)Do[(*mc[5,5]*)Do[(*mc[4,5]*)Do[(*mc[3,5]*)
              Do[(*mc[2,5]*)
               Do[(*mc[1,5]*)
                Do[(*mc[4,4]*)
              Do[(*mc[3,4]*) Do[(*mc[2,4]*)Do[(*mc[1,4]*)Do[(*mc[3,3]*)Do[(*mc[2,3]*)
                  Do[(*mc[1,3]*)
             Do[(*mc[2,2]*)Do[(*mc[1,2]*)Do[(*mc[1,1]*)   c = sSnm[n,
             mc]; r = sSnm[n, mc];
                  xMmatrix1[[r, c]] = Eiicc[n, i, mc]; AppendTo[
             not0Eij, {i, i, r, c, Eiicc[n, i, mc]}], {mc[1, 1], mc[2, 2], mc[1, 2]}];
                  If[n < 3, Throw["Done"]], {mc[1, 2], mc[2, 3],
                  mc[1, 3]}], {mc[2, 2], mc[3, 3], mc[2, 3]}];
                  If[n < 4, Throw["Done"]], {mc[1, 3], mc[2, 4],
                  mc[1, 4]}], {mc[2, 3], mc[3, 4], mc[2, 4]}], {mc[
                  3, 3], mc[4, 4], mc[3, 4]}];
               If[n < 5, Throw["Done"]], {mc[1, 4], mc[2, 5],
                mc[1, 5]}], {mc[2, 4], mc[3, 5], mc[2, 5]}], {mc[3,
                4], mc[4, 5], mc[3, 5]}], {mc[4, 4], mc[5, 5],
            mc[4, 5]}];
          If[n < 6, Throw["Done"]], {mc[1, 5], mc[2, 6],
           mc[1, 6]}], {mc[2, 5], mc[3, 6], mc[2, 6]}], {mc[3, 5],
                          mc[4, 6], mc[3, 6]}], {mc[4, 5], mc[5, 6],
```

```
                                        mc[4, 6]}], {mc[5, 5], mc[6, 6], mc[5, 6]}];

                    If[n < 7, Throw["Done"]], {mc[1, 6], mc[2, 7],

                     mc[1, 7]}], {mc[2, 6], mc[3, 7], mc[2, 7]}], {mc[3, 6],

                    mc[4, 7], mc[3, 7]}], {mc[4, 6], mc[5, 7], mc[4, 7]}], {mc[5,

                   6], mc[6, 7], mc[5, 7]}], {mc[6, 6], mc[7, 7], mc[6, 7]}]];

            mEij[i, i] = xMmatrix1, {i, n}];

  Print["{i,j,Eij}= ",

  Table[{i, j, mEij[i, j] // MatrixForm}, {i, n}, {j, n}]]; glnPASSQ = {n, irrep, {0} == N[Union[Simplify[Un-
ion[Flatten[Chop[

                    Table[N[mEij[i, j]] . N[mEij[k, l]] -

                      N[mEij[k, l]] .

                     N[mEij[i, j]] - (N[\[Delta]ij[j, k] mEij[i, l]] - N[\[Delta]ij[i, l] mEij[k, j]]),

                     {i, n}, {j, n}, {k, n}, {l, k, n}]]]]]]};

  (*su(n)*)

      \[Lambda][1] = mEij[1, 2] + mEij[2, 1]; \[Lambda][2] = I (-mEij[1, 2] + mEij[2,
        1]);

  \[Lambda][3] = mEij[1, 1] - mEij[2, 2];

      \[Lambda][4] = mEij[1, 3] + mEij[3, 1]; \[Lambda][5] = I (-mEij[1, 3] + mEij[3,
        1]);

      \[Lambda][6] = mEij[2, 3] + mEij[3, 2]; \[Lambda][7] = I (-mEij[2, 3] + mEij[3,
        2]);

  \[Lambda][8] = (mEij[1, 1] + mEij[2, 2] - 2 mEij[3, 3]) (1/3)^(1/2);

      \[Lambda][9] = mEij[1, 4] + mEij[4, 1]; \[Lambda][10] = I (-mEij[1, 4] + mEij[4,
        1]);

      \[Lambda][11] = mEij[2, 4] + mEij[4, 2]; \[Lambda][12] = I (-mEij[2, 4] + mEij[4,
        2]);

  \[Lambda][13] = mEij[3, 4] + mEij[4, 3]; \[Lambda][14] =

          I (-mEij[3, 4] + mEij[4, 3]);

  \[Lambda][

            15] = (mEij[1, 1] + mEij[2, 2] + mEij[3, 3] -

             3 mEij[4, 4]) (1/6)^(1/2);

      \[Lambda][16] = mEij[1, 5] + mEij[5, 1]; \[Lambda][17] = I (-mEij[1, 5] + mEij[5,
        1]);
```

```
\[Lambda][18] = mEij[2, 5] + mEij[5, 2]; \[Lambda][19] = I (-mEij[2, 5] + mEij[5,
  2]);
\[Lambda][20] = mEij[3, 5] + mEij[5, 3]; \[Lambda][21] = I (-mEij[3, 5] + mEij[5,
  3]);
\[Lambda][22] = mEij[4, 5] + mEij[5, 4]; \[Lambda][23] = I (-mEij[4, 5] + mEij[5,
  4]);
\[Lambda][
    24] = (mEij[1, 1] + mEij[2, 2] + mEij[3, 3] + mEij[4, 4] -
      4 mEij[5, 5]) (1/10)^(1/2);
\[Lambda][25] = mEij[1, 6] + mEij[6, 1]; \[Lambda][26] = I (-mEij[1, 6] + mEij[6,
  1]);
\[Lambda][27] = mEij[2, 6] + mEij[6, 2]; \[Lambda][28] = I (-mEij[2, 6] + mEij[6,
  2]);
\[Lambda][29] = mEij[3, 6] + mEij[6, 3]; \[Lambda][30] = I (-mEij[3, 6] + mEij[6,
  3]);
\[Lambda][31] = mEij[4, 6] + mEij[6, 4]; \[Lambda][32] = I (-mEij[4, 6] + mEij[6,
  4]);
\[Lambda][33] = mEij[5, 6] + mEij[6, 5]; \[Lambda][34] = I (-mEij[5, 6] + mEij[6,
  5]);
\[Lambda][
    35] = (mEij[1, 1] + mEij[2, 2] + mEij[3, 3] + mEij[4, 4] +
                mEij[5, 5] - 5 mEij[6, 6]) (1/15)^(1/2);
            Table[mTi[i] = \[Lambda][i]/2, {i, n^2 - 1}];
sunPASSQ = {n, irrep, {0} == Union[Chop[N[Union[Flatten[Table[
              mTi[i] . mTi[j] - mTi[j] . mTi[i] -
               I Sum[fijk[i, j, k] mTi[k], {k, n^2 - 1}],
              {i, n^2 - 2}, {j, i + 1, n^2 - 1}]]]]]]};
(*Print["{i,Ti}=     ",Table[{i,mTi[i]//MatrixForm},{i,n^2-1}]];*) (*Make a table of nonzero
Ti components*)
not0Ti = {}; Table[Table[
    If[(0.001 < Abs[mTi[i][[r, c]]]),
     AppendTo[not0Ti, {i, r, c, mTi[i][[r, c]]}]], {r, dimREP}, {c, dimREP}], {i, n^2 - 1}];
PrependTo[
```

```
not0Eij, {n, irrep, dimREP, Length[not0Eij], TimeUsed[] - t0}]; SetDirectory[NotebookDirectory[]];
 (*Create the filename and save the data.*) fileName =
        StringJoin["gl", ToString[n], "irrep", Apply[StringJoin,
            Flatten[{"-", Table[{ToString[irrep[[i]]], "-"}, {i, n}]}]], "gl", ToString[n], ".dat"];
 Put[not0Eij, fileName];
 PrependTo[not0Ti, {n, irrep, dimREP, Length[not0Ti], TimeUsed[] - t0}]; (*Create the filename and save the data.*)
 fileNameB =
        StringJoin["su", ToString[n], "irrep", Apply[StringJoin,
            Flatten[{"-", Table[{ToString[irrep[[i]]], "-"}, {i, n}]}]], "su", ToString[n], ".dat"];
 SetDirectory[NotebookDirectory[]]; Put[not0Ti, fileNameB]

 (*6. Check the matrices; Other post-calculation accounting*) Print["The Eij matrices satisfied the CRs
of the gl(", n,
          ") Lie algebra: ", glnPASSQ[[3]]];
      Print["The Ti matrices satisfied the CRs of the su(", n, ") Lie algebra: ",
        sunPASSQ[[3]]];
      Print["CPU time for the calculation of the irrep = ", TimeUsed[] - t0, "s."];
 tnow2 = AbsoluteTime[];
      Print["Lab time for the calculation of the irrep = ", tnow2 - tnow1, "s."];
 Print[""]
 Print[{{"n |", "irrep |", "matrix dimensions |",
          "# nonzero entries Eij |", "# nonzero entries Ti|"}, {n, irrep, dimREP, Length[not0Eij] - 1,
        Length[not0Ti] - 1}} // Grid]

 (*7. Download a gl(n) data file*) (*Definitions
        glnirrep-#-#-#-gln.dat : the Eij matrices for gl(n) are saved in a file.
 {gl5irrep-5-4-3-2-1-gl5.dat has the matrices Eij for irrep = \ 5,4,3,2,1.)
Record 1. {n, irrep m[\[Tau],n], dimension d of the dxd matrices, \ total number of nonzero entries, \
 CPU time to generate the gl,su matrices}
         Records 2,3,.... { i, j, r, c, the value of Eij^rc }*)
      (*   sunirrep-#-#-#-sun.dat : the Ti matrices for su(n) are saved in a \ file.
 {su5irrep-5-4-3-2-1-su5.dat has the matrices Eij for irrep = \ 5,4,3,2,1.)
Record 1. {n, irrep m[\[Tau],n], dimension d of the dxd matrices, \ total number of nonzero entries, \
```

```
CPU time to generate the gl,su matrices}
                Records 2,3,.... { i, r, c, the value of Ti^rc }*)
(*By default, irrepA = irrep gives the irrep calculated above. *) irrepA =
irrep(*Change irrepA if you have saved another set of generators. *); nA = Length[irrepA];
fileNameA =
          StringJoin["gl", ToString[nA], "irrep", Apply[StringJoin,
            Flatten[{"-", Table[{ToString[irrepA[[i]]], "-"}, {i, nA}]}]], "gl", ToString[nA], ".dat"];
not0EijA = Get[fileNameA]; dimREPA = not0EijA[[1, 3]];
datA = Table[not0EijA[[i]], {i, 2, Length[not0EijA]}]; Table[EijrcA0[i, j, r, c] = 0, {i, nA}, {j, nA}, {r, dimREPA}, {c,
                dimREPA}];
Table[EijrcA0[ datA[[ii, 1]], datA[[ii, 2]] , datA[[ii, 3]],
  datA[[ii, 4]] ] = datA[[ii, 5]], {ii, Length[datA]}]; mEijA[i_, j_] :=
Table[EijrcA0[ i, j , r, c ], {r, dimREPA}, {c, dimREPA}]; glnPASSQA = {nA,
          irrepA, {0.} == N[Union[Simplify[
            Union[Flatten[ Chop[Table[
              N[mEijA[i, j]] . N[mEijA[k, l]] - N[mEijA[k, l]] .
                N[mEijA[i, j]] - (N[\[Delta]ij[j, k] mEijA[i, l]] -
                N[\[Delta]ij[i, l] mEijA[k, j]]), {i, nA}, {j, nA}, {k, nA}, {l, k, nA}]]]]]]};
(*Display the Eij matrices.*)
(*Print["{i,j,Eij}:    ",Table[{i,j,mEijA[i,j]//MatrixForm},{i,nA},{j,nA}]];*)
Print["The EijA matrices retrieved from file ", fileNameA, " have "] Print[{{"nA |", "irrepA |", "matrix dimensions (A) |",
            "# nonzero entries Eij(A) |"}, {nA, irrepA, dimREPA, Length[not0EijA] - 1}}
          // Grid]
        Print["The EijA matrices satisfy the CRs of the gl(", nA, ") Lie algebra: ",
          glnPASSQA[[3]]];
(*8. Download an su(n) data file*)
        (*Change irrepB to the nth row of the GT pattern for the irrep you want *)
        irrepB = irrepA;(*'irrepB = \
```

```
irrepA' gives the su(nA) generators for the generators EijA of irrepA o\ f gl(nA) above.*)
nB = Length[irrepB]; fileNameB =
        StringJoin["su", ToString[nB], "irrep", Apply[StringJoin,
          Flatten[{"-", Table[{ToString[irrepB[[i]]], "-"}, {i, nB}]}]], "su", ToString[nB], ".dat"];
not0TiB = Get[fileNameB]; dimREPB = not0TiB[[1, 3]];
datB = Table[not0TiB[[i]], {i, 2, Length[not0TiB]}];
Table[mTircB0[i, r, c] = 0, {i, nB^2 - 1}, {r, dimREPB}, {c, dimREPB}];
    Table[mTircB0[ datB[[ii, 1]], datB[[ii, 2]] , datB[[ii, 3]] ] = datB[[ii, 4]], {ii, Length[datB]}];
mTiB[i_] := Table[mTircB0[i, r, c], {r, dimREPB}, {c, dimREPB}]; sunPASSQB = {nB,
          irrepB, {0} == Union[Chop[
              N[Union[Flatten[
                Table[mTiB[i] . mTiB[j] - mTiB[j] . mTiB[i] -
                  I Sum[fijk[i, j, k] mTiB[k], {k, nB^2 - 1}], {i,
                  nB^2 - 2}, {j, i + 1, nB^2 - 1}]]]]]]};
(*Print["{i,TiB}: ",Table[{i,mTiB[i]//MatrixForm},{i,n^2-1}]];*)
Print["The TiB matrices retrieved from file ", fileNameB, " have "] Print[{{"nB |", "irrepB |", "matrix dimensions (B) |",
                "# nonzero entries (B) |"}, {nB, irrepB, dimREPB, Length[not0TiB] - 1}} //
            Grid]
      Print["The TiB matrices satisfy the CRs of the su(", nB, ") Lie algebra: ",
        sunPASSQB[[3]]];
(*References*)
(*1. Gel'fand,I.M.and Zetlin,M.L.\
(1950) Finite Dimensional Representations of the Group of Unimodular M\ atrices,Dokl.Akad.Nauk SSSR,71,825-828.*)
(*2. Louck,J.D. and Biedenharn, I.C.(1970) Canonical Unit Adjoint Tensor Operators in U(n),{\textit{J.Math.Phys.}}
{\textbf{11}},2368\[Dash]2414. https://doi.org/10.1063/1.1665404*) (*3. Louck,J.D.\
(1970) Recent Progress Toward a Theory of Tensor Operators in the Unit\ ary Group,Am.J.Phys.38,3-42.*)
(*4. Gould,M.D.(1981) On the Matrix Elements of the U(n) Generators,\
{\textit{J.Math.Phys.}} {\textbf{22}}(1),15-22.*)
```

```
(*5. Wolfram Research,Inc.,Mathematica,Version[15.0],Champaign,\ IL (2026)*)

(*6. Site Maintained by Per Alexandersson,Gelfand-Tsetlin patterns, The Symmetric Functions Catalog
Updated 2026-08-17 (2026)*)

(*URL: https://www.symmetricfunctions.com/gtpatterns.htm *) (*7. Weyl,H.\

(1931) The Theory of Groups and Quantum Mechanics (Dover Publications,\ Inc.,New York)*)

    Print["The max memory in use = ", MaxMemoryUsed[], " bytes."]; Print["end-of-file"];
```

Acknowledgements

The author thanks an anonymous reviewer of a previous article for their helpful suggestions. Special thanks to the members of JAMP for their professional assistance.

Conflicts of Interest

The author declares no conflicts of interest regarding the publication of this paper.

References

[1] Barut, A.O. and Rack, R. (1986) Theory of Group Representations and Applications. 2nd Revised Edition, World Scientific. https://doi.org/10.1142/0352

[2] Hall, B.C. (2015) Lie Groups, Lie Algebras, and Representations. 2nd Edition, Springer.

[3] Pfeifer, W. (2003) The Lie Algebras su(N). 1st Edition, Birkhäuser Verlag.

[4] Gel'fand, I.M. and Zetlin, M.L. (1950) Finite Dimensional Representations of the Group of Unimodular Matrices. *Doklady Akademii nauk SSSR*, **71**, 825-828.

[5] Gould, M.D. (1981) On the Matrix Elements of the U(n) Generators. *Journal of Mathematical Physics*, **22**, 15-22. https://doi.org/10.1063/1.524749

[6] Louck, J.D. and Biedenharn, L.C. (1970) Canonical Unit Adjoint Tensor Operators in U(n). *Journal of Mathematical Physics*, **11**, 2368-2414. https://doi.org/10.1063/1.1665404

[7] Louck, J.D. (1970) Recent Progress toward a Theory of Tensor Operators in the Unitary Groups. *American Journal of Physics*, **38**, 3-42. https://doi.org/10.1119/1.1976225

[8] Wolfram Research Inc (2026) Mathematica, Version [15.0]. Champaign, IL. Wolfram Research Inc.

[9] Per Alexandersson (2026) Gelfand-Tsetlin Patterns. The Symmetric Functions Catalog. https://www.symmetricfunctions.com/gtpatterns.htm

[10] Weyl, H. (1931) The Theory of Groups and Quantum Mechanics. Dover Publications Inc.