On the minuscule representation of type B_n

William J. Cook[*][†] and Noah A. Hughes[*][‡]

April 23, 2014

Abstract

Using only $\mathfrak{sl}_2$ representation theory, we construct the set of weights of the minuscule representation of type B_n (also known as the spin representation). We then derive formulas for the simple reflections viewed as permutations of the weights. Using a computer aided calculuation we study the cycle structures of the permutations representing the Weyl group of type B_n as it acts on the set of weights of the minuscule representation. Then we are able to establish that, for certain ranks, the irreduciblity of the minuscule representation cannot be detected by cycle structures alone.[1]

1 Introduction

Minuscule representations of simple Lie algebras appear in many diverse applications. In fact, a whole book [G] devoted to their combinatorial structure was recently published. Each simple Lie algebra has infinitely many isomorphism classes of finite dimensional irreducible representations. However, not every simple Lie algebra possesses a minuscule representation. Those which do have only a handful.

Minuscule representations have the interesting property that all of their weights lie in a single Weyl group orbit. This then implies that all of the weight spaces are one dimensional. The irreducibility of such a module is guaranteed by the transitive action of the Weyl group. We set out to find when this transitivity (and thus irreducibility) can be seen from the cycle structures of the Weyl group elements (viewed as permutations) alone.

In this paper, we deal only with simple Lie algebras of type B_n. Such algebras have only one minuscule representation which is also known as the spin representation. Spin representations are commonly constructed using Clifford algebras. We are able to avoid such algebras as we use nothing more than $\mathfrak{sl}_2$ representation theory.

After some introductory material, we explicitly determine the set of weights of the minuscule representations of type B_n. This is done inductively by building higher rank modules from lower rank ones. We then derive formulas for the action of the simple reflections (which generate the Weyl group) on the set of weights of the minuscule representation.

First we will recall some terminology and establish notation related to simple Lie algebras. We refer the readers to [EW] and [H] for more details. Unless otherwise stated all vector spaces

[*] Department of Mathematical Sciences, Appalachian State University, 121 Bodenheimer Dr., Boone, NC 28608

[†] cookwj@appstate.edu   http://mathsci.appstate.edu/∼cookwj

[‡] hughesna@appstate.edu

[1] Subject classification: Primary = 17B10 [Representations, algebraic theory (weights)] and Secondary = 20F55 [Reflection and Coxeter groups] Keywords: Lie algebra, minuscule representation, Weyl group

will be finite dimensional and defined over the complex numbers, $\mathbb{C}$. We will let $\mathbb{Z}$ denote the ring of integers.

1.1 Simple Lie algebras

A *Lie algebra* is a vector space $\mathfrak{g}$ (over $\mathbb{C}$) equipped with a bilinear multiplication $[\cdot,\cdot] : \mathfrak{g}\times\mathfrak{g} \to \mathfrak{g}$, called the *Lie bracket*, which is alternating ($[x,x] = 0$ for all $x \in \mathfrak{g}$) and satisfies the Jacobi identity ($[[x,y],z] + [[y,z],x] + [[z,x],y] =$ for all $x,y,z \in \mathfrak{g}$). For each $g \in \mathfrak{g}$ we define $\mathrm{ad}(g) : \mathfrak{g} \to \mathfrak{g}$ to be left multiplication by g: $\mathrm{ad}(g)(x) = [g,x]$. A *subalgebra* of $\mathfrak{g}$ is a subspace of $\mathfrak{g}$ which is closed under the Lie bracket ($\mathfrak{h} \subseteq \mathfrak{g}$ such that for all $x,y \in \mathfrak{h}$ we have $[x,y] \in \mathfrak{h}$). An *ideal* of $\mathfrak{g}$ is a subspace of $\mathfrak{g}$ which absorbs multiplication by elements of $\mathfrak{g}$ ($\mathfrak{i} \subseteq \mathfrak{g}$ such that for all $x \in \mathfrak{i}$ and $g \in \mathfrak{g}$ we have $[g,x] \in \mathfrak{i}$). We call $\mathfrak{g}$ *abelian* if $[x,y] = 0$ for all $x,y \in \mathfrak{g}$. A non-abelian Lie algebra with no proper non-trivial ideals is called *simple* – that is – $\mathfrak{g}$ is simple if $[\mathfrak{g},\mathfrak{g}] \neq \mathbf{0}$ and if $\mathfrak{i}$ is an ideal of $\mathfrak{g}$, then $\mathfrak{i} = \mathbf{0}$ or $\mathfrak{g}$.

As an example, $\mathbb{R}^3$ equipped with the familiar cross product is a 3-dimensional simple Lie algebra (over the field of real numbers $\mathbb{R}$). If we let $\mathfrak{gl}_n$ denote the $n\times n$ complex matrices, then $\mathfrak{gl}_n$ becomes the *general linear* Lie algebra when given the *commutator bracket* $[A,B] = AB - BA$. The set of all trace zero $n \times n$ complex matrices is called the *special linear* Lie algebra $\mathfrak{sl}_n$. It is a subalgebra of $\mathfrak{gl}_n$ and turns out to be simple when $n \geq 2$.

Let $\varphi : \mathfrak{g}_1 \to \mathfrak{g}_2$ be a linear map between two Lie algebras. We call φ a *homomorphism* if $\varphi([x,y]) = [\varphi(x),\varphi(y)]$ for all $x,y \in \mathfrak{g}_1$. Of course, a bijective homomorphism is an *isomorphism.*

One of the early triumphs of Lie theory was Killing and Cartan's classification of all finite dimensional simple Lie algebras (over $\mathbb{C}$). Killing and Cartan were able to show that each finite dimensional simple Lie algebra was isomorphic to one of the algebras on their list:

$$A_n\ (n \geq 1), \quad B_n\ (n \geq 2), \quad C_n\ (n \geq 3), \quad D_n\ (n \geq 4), \quad E_6, E_7, E_8, \quad F_4, \quad \text{and} \quad G_2.$$

Algebras of types A through D are called *classical algebras.* Those of type E, F, and G are called *exceptional algebras.* We refer the reader to [EW] for an accessible discussion of this classification or to [H] or [C] for more complete discussions.

A *Cartan subalgebra* $\mathfrak{h}$ of a simple Lie algebra $\mathfrak{g}$ is a subalgebra which is nilpotent (this means that $\underbrace{[[\cdots[[\mathfrak{h},\mathfrak{h}],\mathfrak{h}],\dots],\mathfrak{h}]}_{k-\text{times}} = \mathbf{0}$ for some integer $k > 0$) and self-normalizing (if $x \in \mathfrak{g}$, $y \in \mathfrak{h}$, and $[x,y] \in \mathfrak{h}$ then $x \in \mathfrak{h}$). Equivalently, a Cartan subalgebra is a maximal toral subalgebra (a *toral* subalgebra is a subalgebra $\mathfrak{h}$ such that for all $h \in \mathfrak{h}$, the linear endomorphism $\mathrm{ad}(h) : \mathfrak{g} \to \mathfrak{g}$ is diagonalizable). Every Cartan subalgebra of a finite dimensional simple Lie algebra $\mathfrak{g}$ has the same dimension. This dimension is called the *rank* of the simple Lie algebra.

Since all toral subalgebras $\mathfrak{h}$ are abelian, we have that for all $x,y \in \mathfrak{h}$, $\mathrm{ad}(x)$ and $\mathrm{ad}(y)$ commute and so the space of endomorphisms $\mathrm{ad}(\mathfrak{h})$ can be simultaneously diagonalized. Thus $\mathfrak{g}$ decomposes into a collection of simultaneous eigenspaces for $\mathrm{ad}(\mathfrak{h})$ for any toral subalgebra $\mathfrak{h}$. By choosing $\mathfrak{h}$ to be maximal toral, our eigenspaces are in some sense maximally refined.

For what follows, let $\mathfrak{g}$ be a simple Lie algebra and let $\mathfrak{h}$ be a Cartan subalgebra of $\mathfrak{g}$. Let $n = \dim(\mathfrak{h})$ be the rank of $\mathfrak{g}$. Since $\mathrm{ad}(\mathfrak{h})$ is simultaneously diagonalizable, $\mathfrak{g} = \prod\limits_{\alpha\in\mathfrak{h}^*} \mathfrak{g}_\alpha$ where $\mathfrak{h}^* = \{f : \mathfrak{g} \to \mathbb{C} \mid f \text{ is linear}\}$ is the dual space of $\mathfrak{h}$ and $\mathfrak{g}_\alpha = \{g \in \mathfrak{g} \mid [h,g] = \alpha(h)g \text{ for all } h \in \mathfrak{h}\}$. When non-trivial, $\mathfrak{g}_\alpha$ is a simultaneous eigenspace corresponding to eigenvalue $\alpha(h)$ for each $h \in \mathfrak{h}$. Since $\mathfrak{h}$ is abelian and self-normalizing, $\mathfrak{g}_{\mathbf{0}} = \mathfrak{h}$. If $\mathbf{0} \neq \alpha \in \mathfrak{h}^*$ and $\mathfrak{g}_\alpha \neq \mathbf{0}$, we call α a *root* and $\mathfrak{g}_\alpha$ a *root space* of $\mathfrak{g}$. Let $\Delta \subset \mathfrak{h}^*$ be the set of roots of $\mathfrak{g}$.

Given a set of roots Δ, there exists a subset $\Pi \subseteq \Delta$ such that each root can be expressed as a non-positive or non-negative integral linear combination of elements of Π. In this case we call the elements of Π *simple roots.* Every root system has many equivalent collections of simple roots. Each such set's cardnality is exactly the rank of $\mathfrak{g}$ (i.e. the dimension of $\mathfrak{h}$). Let us fix such a set of simple roots $\Pi = \{\alpha_1, \dots, \alpha_n\} \subseteq \Delta$. So for each $\alpha \in \Delta$ there exists $c_1, \dots, c_n \in \mathbb{Z}$ such that $\alpha = c_1\alpha_1 + \cdots + c_\ell \alpha_n$ with either all $c_i \geq 0$ (for a *positive root*) or all $c_i \leq 0$ (for a *negative root*).

1.2 The Weyl group and irreducible modules

The simple roots, $\Pi = \{\alpha_1, \dots, \alpha_n\}$, form a basis for $\mathfrak{h}^*$. The *fundamental weights* $\{\lambda_1, \lambda_2, \dots, \lambda_n\}$ form another important basis for $\mathfrak{h}^*$. The root and weight bases are related by the *Cartan matrix* of $\mathfrak{g}$. In particular, if $A = (a_{ij})_{1 \leq i,j \leq n}$ is the Cartan matrix, then $\alpha_i = a_{i1}\lambda_1 + a_{i2}\lambda_2 + \cdots + a_{in}\lambda_n$ for $1 \leq i \leq n$.

For each $1 \leq i \leq n$, we define $\sigma_i : \mathfrak{h}^* \to \mathfrak{h}^*$ by $\sigma_i(\lambda_j) = \lambda_j - \delta_{ij}\alpha_i$ and extend linearly (where δ_{ij} is the Kronecker delta – that is – $\delta_{ii} = 1$ and $\delta_{ij} = 0$ for $i \neq j$). The map σ_i is called the *simple reflection* associated with the simple root α_i. Let $\mathfrak{W}(\mathfrak{g}) = \langle \sigma_1, \sigma_2, \dots, \sigma_n \rangle$ be the group generated by the simple reflections (generated as a subgroup of, for example, $\mathrm{GL}(\mathfrak{h}^*)$). This is called the *Weyl group* of $\mathfrak{g}$.

A (finite dimensional) vector space M (over $\mathbb{C}$) equipped with an bilinear $\mathfrak{g}$-action $(g, \mathbf{v}) \mapsto g \cdot \mathbf{v}$ is a $\mathfrak{g}$*-module* if $[x, y] \cdot \mathbf{v} = x \cdot (y \cdot \mathbf{v}) - y \cdot (x \cdot \mathbf{v})$ for all $x, y \in \mathfrak{g}$ and $\mathbf{v} \in M$. A homomorphism $\varphi : \mathfrak{g} \to \mathfrak{gl}(M)$ (where $\mathfrak{gl}(M)$ is equipped with the commutator bracket) is called a *representation.* It is not hard to show that every module gives rise to a representation and vice-versa. Specifically, given a module action or representation, one can define the other structure as follows: $x \cdot \mathbf{v} = (\varphi(x))(\mathbf{v})$. For what follows, we will treat the words "module" and "representation" as synonyms.

Let $\varphi : M_1 \to M_2$ be a linear map between two $\mathfrak{g}$-modules. If $\varphi(g \cdot \mathbf{v}) = g \cdot \varphi(\mathbf{v})$ for all $g \in \mathfrak{g}$ and $\mathbf{v} \in M_1$, then φ is a $\mathfrak{g}$-module map. A bijective module map is called a ($\mathfrak{g}$-module) isomorphism.

A subspace closed under the action of $\mathfrak{g}$ is called a *submodule.* A non-trivial module ($M \neq \mathbf{0}$) which has no non-trivial proper submodules (if N is a submodule, then $N = \mathbf{0}$ or $N = M$) is called an *irreducible* module. Suppose M is a $\mathfrak{g}$-module and $\lambda \in \mathfrak{h}^*$, we define $M_\lambda = \{\mathbf{v} \in M \mid h \cdot \mathbf{v} = \lambda(h)\mathbf{v} \text{ for all } h \in \mathfrak{h}\}$. If $M_\lambda \neq \mathbf{0}$, we say that M_λ is a *weight space* (whose elements are *weight vectors*) with *weight* λ. Just as $\mathfrak{g}$ is a direct sum of root spaces, $\mathfrak{g}$-modules are direct sums of weight spaces: $M = \coprod\limits_{\lambda \in \mathfrak{h}^*} M_\lambda$.

Let M be an irreducible $\mathfrak{g}$-module. There exists a (unique) weight $\lambda \in \mathfrak{h}^*$ of M such that given any other weight $\mu \in \mathfrak{h}^*$ we have $\mu = \lambda - \sum\limits_{i=1}^{n} b_i\alpha_i$ where $b_i \in \mathbb{Z}$ and $b_i \geq 0$. So every other weight is obtained by subtracting certain collections of positive roots from this weight. Such a weight, λ, is unique and is called the *highest weight* of M. If $\lambda \in \mathfrak{h}^*$ and there exists $c_i \in \mathbb{Z}$, $c_i \geq 0$ such that $\lambda = \sum\limits_{i=1}^{n} c_i\lambda_i$ (the λ_i's are the fundamental weights), then λ is a *dominant integral weight.*

Highest weights of finite dimensional irreducible modules are dominant integral. Conversely, each dominant integral weight is the highest weight of some finite dimensional irreducible module. Two irreducible modules with the same highest weight are isomorphic, so we have a bijection between the set of dominant integral weights and the isomorphism classes of finite dimensional irreducible modules.

Let λ be a dominant integral weight for for some simple Lie algebra of type X_n. We denote the irreducible highest weight X_n-module with highest weight λ by $L(X_n, \lambda)$ or just $L(\lambda)$ when the algebra is understood.

1.3 Minuscule modules

There are many equivalent ways of defining minuscule weights. In fact, 6 equivalent conditions are given in [B] (see chapter VIII section 7.3). The following definition best fits our purposes:

Definition 1.1. *Let $L(\lambda)$ be an irreducible finite dimensional $\mathfrak{g}$-module with non-zero highest weight $\lambda \in \mathfrak{h}^*$. Then λ is a* minuscule weight *and $L(\lambda)$ is a* minuscule module *if the Weyl group $\mathfrak{W}(\mathfrak{g})$ acts transitively on the set of weights of $L(\lambda)$ (i.e. $\mathfrak{W}(\mathfrak{g}) \cdot \lambda$ is the set of all weights of $L(\lambda)$).*

Given an $\mathfrak{g}$-module M, we know M decomposes into weight spaces: M_λ for $\lambda \in \mathfrak{h}^*$. The dimension of a weight space M_λ is called the *multiplicity* of the weight λ.

If $\mu = w \cdot \lambda$ for $\mu, \lambda \in \mathfrak{h}^*$ and $w \in W$, then M_μ and M_λ have the same dimension. Therefore, weights lying in an orbit of the Weyl group all have the same multiplicity. Thus since the weights of a minuscule module all lie in a single Weyl group orbit, the weight spaces in a minuscule module must all have the same multiplicity as the highest weight. But the highest weight space for an irreducible module is always one dimensional. Therefore, all the weight spaces in a minuscule module are one dimensional and the dimension of a minuscule module is the same as the number of its weights.

Both [H] (section 13, page 72, exercise 13) and [B] (chapter VIII, section 7.3, page 132) give the following table of minuscule weights for finite dimensional simple Lie algebras:

Type:	A_n	B_n	C_n	D_n	E_6	E_7
Minuscule Weights:	$\lambda_1, \ldots, \lambda_n$	λ_n	λ_1	$\lambda_1, \lambda_{n-1}, \lambda_n$	λ_1, λ_6	λ_7

Let us note that algebras of type F_4, E_8, and G_2 have no minuscule representations. Also, be warned, we will be reversing the indices of the simple roots of our algebra of study, B_n, so that the minuscule representation will have highest weight λ_1 (instead of λ_n).

For further information about minuscule representations we direct the reader to either [B] Chapter VII Section 7.3 or the tract [G] by R. M. Green. Green's book is entirely devoted to the study of minuscule representations and contains a wealth of information about them.

1.4 $\mathfrak{sl}_2$-representation theory

It cannot be overstated how important the study of the smallest simple Lie algebra $\mathfrak{sl}_2$ (type A_1) is to the classification and understanding of the finite dimensional simple Lie algebras and their representation theory. Each simple Lie algebra is essentially built from copies of $\mathfrak{sl}_2$. Likewise in a similar way each representation of a simple Lie algebra is built from copies of irreducible $\mathfrak{sl}_2$-representations. Let us recall some facts about irreducible $\mathfrak{sl}_2$-modules.

First,

$$\mathfrak{sl}_2 = \{X \in \mathbb{C}^{2\times 2} \mid \mathrm{tr}(X) = 0\} = \mathrm{span}\left\{E = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}, F = \begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix}, H = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}\right\}$$

(2×2 traceless matrices) where $[X,Y] = XY - YX$. We have $[H,E] = 2E$, $[H,F] = -2F$, $[E,F] = H$. Note that $\mathfrak{h} = \text{span}(H)$ is a Cartan subalgebra for $\mathfrak{sl}_2$. Define $\alpha_1 \in \mathfrak{h}^*$ by $\alpha_1(H) = 2$. Then α_1 is our simple root and λ_1 where $\alpha_1 = 2\lambda_1$ is the corresponding fundamental weight.

Suppose M is an irreducible $\mathfrak{sl}_2$-module. Then $M = L(A_1, k\lambda_1) = L(k\lambda_1)$ for some non-negative integer k. The weights of M are $k\lambda_1, k\lambda_1 - \alpha_1 = (k-2)\lambda_1, \ldots, k\lambda_1 - k\alpha_1 = (k-2k)\lambda_1 = -k\lambda$ and each of the corresponding weight spaces are 1-dimensional. In particular, if $M = L(\lambda_1)$, then the only weights are λ_1 and $\lambda_1 - \alpha_1 = \lambda_1 - 2\lambda_1 = -\lambda_1$.

By identifying the α_1 with "2" and λ_1 with "1" we get that the weights of $L(k)$ form the set of all even (if k is even) or all odd (if k is odd) integers between $-k$ and k.

Given a simple Lie algebra $\mathfrak{g}$ with root α, $\mathfrak{s}_\alpha = \mathfrak{g}_\alpha \oplus [\mathfrak{g}_\alpha, \mathfrak{g}_{-\alpha}] \oplus \mathfrak{g}_{-\alpha}$ is a subalgebra which is isomorphic with $\mathfrak{sl}_2$. So any $\mathfrak{g}$-module decomposes as an $\mathfrak{sl}_2$-module for each copy of $\mathfrak{sl}_2$ associated with root α. In particular, if $\mu = \sum_{i=1}^{n} c_i\lambda_i$ is a weight of a finite dimensional $\mathfrak{g}$-module M and $c_i > 0$, then treating M as an $\mathfrak{sl}_2 \cong \mathfrak{s}_{\alpha_i}$-module, we see that M contains a copy of the irreducible $\mathfrak{sl}_2$-module $L(A_1, c_i)$. Therefore, $\mu, \mu - \alpha_i, \mu - 2\alpha_i, \ldots, \mu - c_i\alpha_i$ are all weights of M. We call this an *α_i-weight string.*

1.5 Simple Lie algebras of type B_n

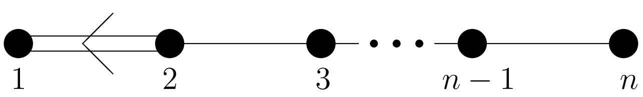
We now turn to the simple Lie algebras of type B_n. Algebras of type B_n can be realized as the *special orthogonal* Lie algebras $\mathfrak{so}_{2n+1}$. Specifically, letting I_n denote the $n \times n$ identity matrix, we have that the special orthogonal Lie algebra is the following set of $(2n+1) \times (2n+1)$ complex matrices:

$$\mathfrak{so}_{2n+1} = \left\{ X \in \mathfrak{gl}_{2n+1} \;\middle|\; X^T \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & I_n \\ 0 & -I_n & 0 \end{bmatrix} = - \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & I_n \\ 0 & -I_n & 0 \end{bmatrix} X \right\}.$$

This is a $2n^2 + n$ dimensional simple Lie algebra of rank n. Let us fix a collection of simple roots $\Pi = \{\alpha_1, \ldots, \alpha_n\}$ and corresponding fundamental weights $\Lambda = \{\lambda_1, \ldots, \lambda_n\}$ for this algebra. We have that the Cartan matrix (the change of basis matrix from Λ to Π) is

$$A = \begin{bmatrix} 2 & -1 & 0 & \cdots & 0 & 0 \\ -2 & 2 & -1 & \cdots & 0 & 0 \\ 0 & -1 & 2 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 2 & -1 \\ 0 & 0 & 0 & \cdots & -1 & 2 \end{bmatrix}$$

with corresponding Dynkin diagram



We would like to stress that our choice of indices is non-standard. Usually the node at the end of the diagram with the double bar and arrow comes last. We have chosen this non-standard labeling so that we have the natural embedding $B_n \subset B_{n+1}$. In particular, deleting the final node (or final row and column from our Cartan matrix) we get a type B algebra of with rank decreased by 1.

Explicitly we have the following relationships between our fundamental weights and simple roots: $\alpha_1 = 2\lambda_1 - \lambda_2$, $\alpha_2 = -2\lambda_1 + 2\lambda_2 - \lambda_3$, $\alpha_3 = -\lambda_2 + 2\lambda_3 - \lambda_4$, $\alpha_4 = -\lambda_3 + 2\lambda_4 - \lambda_5$, $\dots$, $\alpha_{n-1} = -\lambda_{n-2} + 2\lambda_n - \lambda_{n-1}$, $\alpha_n = -\lambda_{n-1} + 2\lambda_n$. For convenience we will write $\alpha_n = -\lambda_{n-1} + 2\lambda_n - \lambda_{n+1}$.

Notice that if we set $\lambda_{n+1} = 0$, then α_n is last simple root of B_n. Leaving it alone yields the next to last simple root of B_{n+1}. Also, notice that if we allow $n = 1$, we get that our Cartan matrix is $A = [2]$ so that $B_1 = A_1$ (indeed $\mathfrak{so}_3 \cong \mathfrak{sl}_2$). We will allow this case and note that λ_1 is still a minuscule weight in this $n = 1$ case.

2 The weights of the minuscule representation of type B_n

Recall that the irreducible representation of B_n with highest weight λ_1 (according to our non-standard indexing) is a minuscule representation. Fix the notation $M_n = L(B_n, \lambda_1)$. One can show that when we restrict the action of B_n to the copy of B_{n-1} obtained by deleting the final simple root, M_n decomposes into 2 copies of M_{n-1}. That is as B_{n-1}-modules $L(B_n, \lambda_1) \cong L(B_{n-1}, \lambda_1) \oplus L(B_{n-1}, \lambda_1)$. We will demonstrate this by explicitly determining the weights of M_n. This will be done using just a few basic facts about $\mathfrak{sl}_2$-representation theory. Let W_n be the weights of M_n.

Theorem 2.1. *Let $W_1 = \{\lambda_1, -\lambda_1 + \lambda_2\}$ and $W_n = W_{n-1} \cup (\lambda_{n+1} - W_{n-1})$ (disjoint union), then both $\pm W_n$ are the set of weights for $L(B_n, \lambda_1)$.*

Before beginning the proof, notice that our weights W_n reference "λ_{n+1}". Recall that $\lambda_{n+1} = 0$ when treating these as weights for the B_n-module M_n, so the λ_{n+1}'s are ghosts or echoes coming from the corresponding B_{n+1}-module M_{n+1}. This begs the question, "Why keep them around?" It turns out that these "ghosts" help streamline our proofs and allow for a very nice inductive argument.

Proof:

Note that $B_1 = A_1$, so M_1 is the irreducible $\mathfrak{sl}_2$-module with highest weight 1. Therefore, $W_1 = \{\lambda_1, -\lambda_1\}$.

Define $W_0 = \{\lambda_1\}$ (which would be $\{0\}$ if we could allow a B_0 case). Then $W_0 \cup (\lambda_2 - W_0) = \{\lambda_1, \lambda_2 - \lambda_1\}$. So $W_1 = W_0 \cup (\lambda_2 - W_0)$ (when we set $\lambda_2 = 0$) as required.

Alternatively, we could check that $W_1 = \{\lambda_1, -\lambda_1 + \lambda_2\}$ is closed under the action of the $\mathfrak{s}_1$ (the copy of $\mathfrak{sl}_2$ built from the $\pm\alpha_1$ root spaces). The coefficient of λ_1 is 1, so we need to go down by only 1 copy of α_1 to get our entire $\mathfrak{sl}_2$-string: $\lambda_1 - \alpha_1 = \lambda_1 - (2\lambda_1 - \lambda_2) = -\lambda_1 + \lambda_2 \in W_1$. Now because B_1 is generated by the root spaces for $\pm\alpha_1$, we must have that this is a complete set of weights for M_1.

Likewise, $(\lambda_1 - \lambda_2) - \alpha_1 = (\lambda_1 - \lambda_2) - (2\lambda_1 - \lambda_2) = -\lambda_1$ so $-W_1 = \{-\lambda_1, \lambda_1 - \lambda_2\}$ is complete set of weights for M_1. Notice that if we set $\lambda_2 = 0$, then $W_1 = -W_1 = \{\lambda_1, -\lambda_1\}$ is the set of weights for $M_1 = L(B_1, \lambda_1)$ (without "ghosts").

To get our inductive argument going we need to check the rank $n = 2$ and $n = 3$ cases as well (this is to get past the first few simple roots whose formulas do not fall into the pattern of the later roots).

Consider $W_2 = W_1 \cup (\lambda_3 - W_1)$. We know that $\pm W_1$ are closed with respect to α_1-weight strings and thus $\lambda_3 - W_1$ is as well (because λ_3 does not interact with the root $\alpha_1 = 2\lambda_1 - \lambda_2$ at all). Let us check for closure with respect to α_2-weight strings.

The only weight in W_1 in which λ_2 appears with a positive coefficient is $-\lambda_1 + \lambda_2$. The coefficient of λ_2 is 1 so we only need to go down by 1 copy of α_2 (again appealing to $\mathfrak{sl}_2$-representation theory):

$$(-\lambda_1 + \lambda_2) - \alpha_2 = (-\lambda_1 + \lambda_2) - (-2\lambda_1 + 2\lambda_2 - \lambda_3) = \lambda_3 - (-\lambda_1 + \lambda_2) \in \lambda_3 - W_1.$$

Now λ_2 does not appear in any of the weights in $\lambda_3 - W_1$ with a positive coefficient, so we have finished checking the close under α_2-weight strings. Thus since $W_2 = W_1 \cup (\lambda_3 - W_1)$ is closed under the action of the α_1 and α_2-weight strings and since the root spaces of $\pm\alpha_1$ and $\pm\alpha_2$ generate B_2, we have that W_2 must be the complete set of weights for M_2.

If we consider $-W_2 = -W_1 \cup (-\lambda_3 + W_1)$, then the only change is that the single weight with a positive λ_2 coefficient is $-\lambda_3 - \lambda_1 + \lambda_2$. Again we need only go down by 1 copy of α_2:

$$(-\lambda_3 - \lambda_1 + \lambda_2) - \alpha_2 = (-\lambda_3 - \lambda_1 + \lambda_2) - (-2\lambda_1 + 2\lambda_2 - \lambda_3) = -(-\lambda_1 + \lambda_2) \in W_1.$$

As before, both $\pm W_2$ are the set of weights for M_2. If we set $\lambda_3 = 0$, we get

$$W_2 = -W_2 = \{\lambda_1, -\lambda_1 + \lambda_2, \lambda_1 - \lambda_2, -\lambda_1\}$$

is the set of weights for $M_2 = L(B_2, \lambda_1)$ (without "ghosts").

Let us take care of one last base case – that of W_3. First, since $\pm W_2$ are closed under α_1 and α_2-weight strings and thus so is $\lambda_4 - W_2$ (since λ_4 has no interactions with $\alpha_1 = 2\lambda_1 - \lambda_2$ or $\alpha_2 = -2\lambda_1 + 2\lambda_2 - \lambda_3$). So $W_3 = W_2 \cup (\lambda_4 - W_2)$ is closed with respect to α_1 and α_2-weight strings. Let's check the α_3-weight strings.

Notice that $W_3 = W_2 \cup (\lambda_4 - W_2) = W_1 \cup (\lambda_3 - W_1) \cup (\lambda_4 - W_1) \cup (\lambda_4 - \lambda_3 + W_1)$. So the only weights with positive λ_3 coefficients appear in $\lambda_3 - W_1 = \{\lambda_3 - \lambda_1, \lambda_3 + \lambda_1 - \lambda_2\}$. In both cases the coefficient of λ_3 is 1 so we only need to go down by 1 copy of α_3:

$$(\lambda_3 - \lambda_1) - \alpha_3 = (\lambda_3 - \lambda_1) - (-\lambda_2 + 2\lambda_3 - \lambda_4) = \lambda_4 - (\lambda_1 - \lambda_2 + \lambda_3) \in \lambda_4 - W_2 \subset W_3,$$

$$(\lambda_1 - \lambda_2 + \lambda_3) - \alpha_3 = (\lambda_1 - \lambda_2 + \lambda_3) - (-\lambda_2 + 2\lambda_3 - \lambda_4) = \lambda_4 - (-\lambda_1 + \lambda_3) \in \lambda_4 - W_2 \subset W_3.$$

Therefore,

$$W_3 = \{\lambda_1, -\lambda_1 + \lambda_2, \lambda_1 - \lambda_2 + \lambda_3, -\lambda_1 + \lambda_3, \lambda_1 - \lambda_3 + \lambda_4, -\lambda_1 + \lambda_2 - \lambda_3 + \lambda_4, \lambda_1 - \lambda_2 + \lambda_4, -\lambda_1 + \lambda_4\}$$

is closed under α_1, α_2, and α_3-weight strings and thus is the complete set of weights of M_3. Likewise, the same is true for $-W_3$.

Moreover, setting $\lambda_4 = 0$, we get that

$$W_3 = -W_3 = \{\lambda_1, -\lambda_1 + \lambda_2, \lambda_1 - \lambda_2 + \lambda_3, -\lambda_1 + \lambda_3, \lambda_1 - \lambda_3, -\lambda_1 + \lambda_2 - \lambda_3, \lambda_1 - \lambda_2, -\lambda_1\}$$

is the set of weights for $M_3 = L(B_3, \lambda_1)$ (without "ghosts").

Now consider some $n \geq 4$ (we have already proven cases $n = 1, 2, 3$). Suppose that for each $j < n$, $\pm W_j$ is closed under α_k-weight strings for $k = 1, \ldots, j$. In particular, $\pm W_{n-1}$ are closed under $\alpha_1, \ldots, \alpha_{n-1}$-weight strings. Therefore, so is $\lambda_{n+1} - W_{n-1}$ since λ_{n+1} does not interact with any $\alpha_1, \ldots, \alpha_{n-1}$ (they only involve λ_k where $k \leq n$). Therefore, $W_n = W_{n-1} \cup (\lambda_{n+1} - W_{n-1})$ is closed under $\alpha_1, \ldots, \alpha_{n-1}$-weight strings. This leaves us to check the α_n-weight string.

Next, notice that

$$W_n = W_{n-1} \cup (\lambda_{n+1} - W_{n-1}) = W_{n-2} \cup (\lambda_n - W_{n-2}) \cup (\lambda_{n+1} - W_{n-2}) \cup (\lambda_{n+1} - \lambda_n + W_{n-2}) .$$

Thus λ_n appears with a positive coefficient only in the subset $\lambda_n - W_{n-2}$. Here, as always, λ_n appears with coefficient 1 (W_{n-2} involves only $\lambda_1, \ldots, \lambda_{n-1}$ so it cannot contribute to λ_n's coefficient).

Consider $\lambda_n - \mu$ where $\mu \in W_{n-2}$. Then, because $n > 2$, $\alpha_n = -\lambda_{n-1} + 2\lambda_n - \lambda_{n+1}$ so

$$(\lambda_n - \mu) - \alpha_n = (\lambda_n - \mu) - (-\lambda_{n-1} + 2\lambda_n - \lambda_{n+1}) = \lambda_{n+1} - \lambda_n + (\lambda_{n-1} - \mu).$$

Now recall that $\mu \in W_{n-2} = W_{n-3} \cup (\lambda_{n-1} - W_{n-3})$, so either $\mu \in W_{n-3}$ or $\mu \in \lambda_{n-1} - W_{n-3}$.

Case 1: $\mu \in W_{n-3}$, so $\lambda_{n-1} - \mu \in \lambda_{n-1} - W_{n-3} \subset W_{n-2}$.

Case 2: $\mu \in \lambda_{n-1} - W_{n-3}$, so $\mu = \lambda_{n-1} - \mu'$ for some $\mu' \in W_{n-3}$. Therefore, $\lambda_{n-1} - \mu = \lambda_{n-1} - (\lambda_{n-1} - \mu') = \mu' \in W_{n-3} \subset W_{n-2}$.

In either case, $\lambda_{n-1} - \mu \in W_{n-2}$. Therefore, $\lambda_n - (\lambda_{n-1} - \mu) \in \lambda_n - W_{n-2} \subset W_{n-1}$ and so $(\lambda_n - \mu) - \alpha_n = \lambda_{n+1} - (\lambda_n - (\lambda_{n-1} - \mu)) \in \lambda_{n+1} - W_{n-1} \subset W_n$. Thus we have that W_n is closed under the α_n-weight string. This shows that W_n is closed under $\alpha_1, \ldots, \alpha_n$-weight strings.

Finally, consider $-W_n = -W_{n-1} \cup (-\lambda_{n+1} + W_{n-1})$

$$= -W_{n-2} \cup (-\lambda_n + W_{n-2}) \cup (-\lambda_{n+1} + W_{n-2}) \cup (-\lambda_{n+1} + \lambda_n - W_{n-2}) .$$

As with W_n, we have that $-W_n$ is closed under $\alpha_1, \ldots, \alpha_{n-1}$-weight strings, so we only need to check α_n's string. The only weights in which λ_n appears with a positive coefficient are in $-\lambda_{n+1} + \lambda_n - W_{n-2}$. As before, since only $\lambda_1, \ldots, \lambda_{n-1}$ are involved in the weights in W_{n-2}, the coefficient of λ_n is 1 and we only need to down by 1 copy of α_n. Let $\mu \in W_{n-2}$. Then,

$$(-\lambda_{n+1} + \lambda_n - \mu) - \alpha_n = (-\lambda_{n+1} + \lambda_n - \mu) - (-\lambda_{n-1} + 2\lambda_n - \lambda_{n+1}) = -\lambda_n + \lambda_{n-1} - \mu.$$

But we already know from the above argument that $\lambda_{n-1} - \mu \in W_{n-2}$. Therefore,

$$(-\lambda_{n+1} + \lambda_n - \mu) - \alpha_n = -\lambda_n + (\lambda_{n-1} - \mu) \in -\lambda_n + W_{n-2} \subset -W_n.$$

So $-W_n$ is closed under $\alpha_1, \ldots, \alpha_n$-weight strings.

Therefore, by induction, we have that $\pm W_n$ are the set of weights for M_n for $n \geq 1$. □

Corollary 2.2. *Setting* $\lambda_{n+1} = 0$*, we have* $W_n = -W_n$*. Also,* $\dim(L(B_n, \lambda_1)) = 2^n$.

Proof: We have already remarked that setting $\lambda_{n+1} = 0$ in $\pm W_n$ yields the weights "without ghosts" for $L(B_n, \lambda_1)$. But notice that

$$W_n|_{\lambda_{n+1}=0} = W_{n-1} \cup (0 - W_{n-1}) = -W_{n-1} \cup (-0 + W_{n-1}) = -W_n|_{\lambda_{n+1}=0} .$$

For the second statement, notice that $|W_1| = 2^1$. Assume $|W_{n-1}| = 2^{n-1}$ then $|W_n| = |W_{n-1}| + |\lambda_{n+1} - W_{n-1}|$ since the union of W_{n-1} and $\lambda_{n+1} - W_{n-1}$ is disjoint. Next, notice that $|W_{n-1}| = |\lambda_{n+1} - W_{n-1}|$. Therefore, $|W_n| = |W_{n-1}| + |W_{n-1}| = 2^{n-1} + 2^{n-1} = 2^n$. So by induction $L(B_n, \lambda_1)$ has 2^n weights. But this is a minuscule module so that is also its dimension. □

Corollary 2.3. *If* $\mu = \sum_{j=1}^{n+1} c_j\lambda_j \in W_n$ *then* $c_j \in \{-1, 0, 1\}$. *Thus this also holds for* $W_n\Big|_{\lambda_{n+1}=0}$ *(the weights of* $L\left(B_n,\, \lambda_1\right)$*).*

Proof: This is true for $W_1 = \{\lambda_1, -\lambda_1 + \lambda_2\}$. Notice that the elements of W_{n-1} do not involve λ_{n+1}, so if the statement is true about W_{n-1}, it must also follow for $W_n = W_{n-1} \cup (\lambda_{n+1} - W_{n-1})$. □

Corollary 2.4. $L\left(B_n,\, \lambda_1\right) \cong L\left(B_{n-1},\, \lambda_1\right) \oplus L\left(B_{n-1},\, \lambda_1\right)$ *as* B_{n-1}*-modules.*

Proof: We have that $W_n\Big|_{\lambda_{n+1}=0} = W_{n-1} \cup -W_{n-1}$ is the set of weights for the B_n-module $L(B_n, \lambda_1)$. Also, $-W_{n-1}\Big|_{\lambda_n=0} = W_{n-1}\Big|_{\lambda_n=0}$ is the set of weights for the B_{n-1}-module $L(B_{n-1}, \lambda_1)$. Thus there exists $\Lambda_1 \in W_{n-1}$ and $\Lambda_2 \in -W_{n-1}$ such that $\Lambda_1\Big|_{\lambda_n=0} = \Lambda_2\Big|_{\lambda_n=0} = \lambda_1$. Let $\mathbf{v}_i$ be a (non-zero) weight vector with weight Λ_i $(i = 1, 2)$. Since $\mathbf{v}_1$ and $\mathbf{v}_2$ have weight λ_1 (with respect to B_{n-1}), they each generate a copy of $L(B_{n-1}, \lambda_1)$, say U_1 and U_2. The weights of U_1 (generated by $\mathbf{v}_1$) exhaust W_{n-1} and those of U_2 (generated by $\mathbf{v}_2$) exhaust $-W_{n-1}$. This shows that $L(B_n, \lambda_1)$ contains a submodule $U_1 \oplus U_2$ where $W_i \cong L(B_{n-1}, \lambda_1)$ (the sum is direct because the sets of weights W_{n-1} and $-W_{n-1}$ are disjoint). Finally, $U_1 \oplus U_2 = L(B_n, \lambda_1)$ since all weight spaces are 1-dimensional and W_n, $-W_n$ exhaust all of the weights. □

3 Weyl Group Action as Permutations

In this we turn our attention to the action of the Weyl group $\mathfrak{W}(B_n)$ on the weights of the minuscule module $L(B_n, \lambda_1)$. Our goal is to get an explicit formula for the action of the group on these weights.

Observe that if the coefficient of λ_k in λ is 1, then $\sigma_k(\lambda) = \lambda - \alpha_k$. If the coefficent is -1, $\sigma_k(\lambda) = \lambda + \alpha_k$. Finally, if the coefficient is 0, $\sigma_k(\lambda) = \lambda$. Therefore, since the coefficients of the fundamental weights are $-1, 0, 1$ in the weights of the minuscule modules, applying σ_k is the same as traveling up or down the α_k-weight string. Therefore, the simple reflections permute W_n and $-W_n$ even with "ghosts" (i.e. even without setting $\lambda_{n+1} = 0$).

Assume $n \geq 2$ and let $2^n = 4\ell$ and $1 \leq j \leq 2\ell$. Define $\tau_n \in S_{2\ell}$ (permutations of 2ℓ) as follows: $\tau_n(j) = 2\ell - j + 1$. Notice that $\tau_n \circ \tau_n(j) = 2\ell - (2\ell - j + 1) + 1 = j$ so $\tau_n^{-1} = \tau_n$. Expressing τ_n as a product of disjoint cycles (in fact, disjoint transpositions) we have

$$\tau_n = (1, 2\ell)(2, 2\ell - 1) \ldots (\ell, \ell - 1).$$

We can (and do wish to) treat τ_n as a permutation in S_k for any $k \geq 2\ell$ by letting $\tau_n(x) = x$ for all $x > 2\ell$.

We know from Theorem 2.1 that

$$W_n = W_{n-1} \cup \left(\lambda_{n+1} - W_{n-1}\right).$$

Let $W_0 = \{\lambda_1\} = \{\mu_1\}$ and then define $\lambda_2 - W_0 = \{\lambda_2 - \lambda_1\} = \{\mu_2\}$. Continuing in this fashion, suppose $W_{n-1} = \{\mu_1, \ldots, \mu_{2\ell}\}$. Then let $\lambda_{n+1} - W_{n-1} = \{\mu_{2\ell+1}, \ldots, \mu_{4\ell}\}$ by setting

$$\mu_{2\ell+j} = \lambda_{n+1} - \mu_{4\ell-j+1} = \lambda_{n+1} - \mu_{\tau_{n+1}(j)} = \lambda_{n+1} - \mu_{2\ell+\tau_n(j)}.$$

So W_{n-1} is the first half of W_n. To get the second half of W_n, we negate W_{n-1}, add λ_{n+1}, and reverse its order. In particular,

$$\mu_1 = \lambda_1, \quad \mu_2 = \lambda_2 - \mu_1 = \lambda_2 - \lambda_1, \quad \mu_3 = \lambda_3 - \mu_2 = \lambda_3 - \lambda_2 + \lambda_1, \quad \mu_4 = \lambda_3 - \mu_1 = \lambda_3 - \lambda_1, \ldots,$$

$$\mu_{2\ell+1} = \lambda_{n+1} - \mu_{2\ell}, \quad \mu_{2\ell+2} = \lambda_{n+1} - \mu_{2\ell-1}, \quad \mu_{4\ell} = \lambda_{n+1} - \mu_1, \ldots.$$

From this point on we will freely move back and forth treating simple reflections σ_j as both permutations of weights and permutations of their indices – that is – we have $\sigma_j(\mu_a) = \mu_b$ if and only if $\sigma_j(a) = b$.

Recall that $\sigma_i(\lambda_j) = \lambda_j - \delta_{ij}\alpha_i$ and that because σ is linear $\sigma_i(-\lambda_j) = -\sigma(\lambda_j) = -\lambda_j + \delta_{ij}\alpha_i$. So, in W_1 we have,

$$\sigma_1(\mu_1) = \sigma_1(\lambda_1) = \lambda_1 - (2\lambda_1 - \lambda_2) = -\lambda_1 + \lambda_2 = \mu_2,$$
$$\sigma_1(\mu_2) = \sigma_1(-\lambda_1 + \lambda_2) = \sigma_1(-\lambda_1) + \sigma_1(\lambda_2) = -\lambda_1 + (2\lambda_1 - \lambda_2) + \lambda_2 = \lambda_1 = \mu_1.$$

Therefore σ_1 is a permutation transposing μ_1 and μ_2. By identifying μ_i with its index i, we get that σ_1 is precisely the transposition $\sigma_1 = (12)$ when acting on W_1.

Let us see what happens when we move to rank 2. In W_2 we have,

$$\sigma_1(\mu_1) = \sigma_1(\lambda_1) = \lambda_1 - (2\lambda_1 - \lambda_2) = -\lambda_1 + \lambda_2 = \mu_2,$$
$$\sigma_1(\mu_2) = \sigma_1(-\lambda_1 + \lambda_2) = \sigma_1(-\lambda_1) + \sigma_1(\lambda_2) = -\lambda_1 + (2\lambda_1 - \lambda_2) + \lambda_2 = \lambda_1 = \mu_1,$$
$$\sigma_1(\mu_3) = \sigma_1(\lambda_1 - \lambda_2 + \lambda_3) = \sigma_1(\lambda_1) + \sigma_1(-\lambda_2) + \sigma_1(\lambda_3) = \lambda_1 - (2\lambda_1 - \lambda_2) - \lambda_2 + \lambda_3 = -\lambda_1 + \lambda_3 = \mu_4,$$
$$\sigma_1(\mu_4) = \sigma_1(-\lambda_1 + \lambda_3) = \sigma_1(-\lambda_1) + \sigma_1(\lambda_3) = -\lambda_1 + (2\lambda_1 - \lambda_2) + \lambda_3 = \lambda_1 - \lambda_2 + \lambda_3 = \mu_3,$$

$$\sigma_2(\mu_1) = \sigma_2(\lambda_1) = \lambda_1 = \mu_1,$$
$$\sigma_2(\mu_2) = \sigma_2(-\lambda_1 + \lambda_2) = -\lambda_1 + \lambda_2 - (-2\lambda_1 + 2\lambda_2 - \lambda_3) = \lambda_1 - \lambda_2 + \lambda_3 = \mu_3,$$
$$\sigma_2(\mu_3) = \sigma_2(\lambda_1 - \lambda_2 + \lambda_3) = \lambda_1 - \lambda_2 + (-2\lambda_1 + 2\lambda_2 - \lambda_3) + \lambda_3 = -\lambda_1 + \lambda_2 = \mu_2,$$
$$\sigma_2(\mu_4) = \sigma_2(-\lambda_1 + \lambda_3) = -\lambda_1 + \lambda_3 = \mu_4.$$

Thus, suppressing one cycles we see that $\sigma_1 = (12)(34)$ and $\sigma_2 = (23)$ in W_2.

If we continue this sort of calculation for σ_1, σ_2, and σ_3 acting on W_3 we would find that

$$\sigma_1 = (12)(34)(56)(78), \qquad \sigma_2 = (23)(67), \qquad \sigma_3 = (35)(46).$$

Now that a pattern is becoming clear let us investigate this action in general. To continue, we will need to make use of the following lemma:

Lemma 3.1. *Acting on W_n, we have $\tau_n \sigma_j \tau_n = \sigma_j$ for $j = 1, 2, \ldots, n-1$.*

Proof: First, recall that $W_n = W_{n-2} \cup (\lambda_n - W_{n-2}) \cup (\lambda_{n+1} - W_{n-1})$.

Let $\mu \in \lambda_{n+1} - W_{n-1}$. Since σ_j $(j < n)$ permutes $-W_{n-1}$, it permutes $\lambda_{n+1} - W_{n-1}$ (σ_j fixes λ_{n+1} and σ_j is linear). Thus both μ and $\sigma_j(\mu)$ are equal to μ_k's with $k > 2\ell$. Because of this τ_n has no effect on either μ or $\sigma_j(\mu)$. Thus $\tau_n(\sigma_j(\tau_n(\mu))) = \sigma_j(\mu)$.

Next, let $\mu \in W_{n-2}$, so $\mu = \mu_k$ for some $1 \leq k \leq \ell$. Then $\tau_n(\mu_k) = \mu_{2\ell-k+1} = \lambda_n - \mu_k$ (by the definition of $\mu_{2\ell-k+1}$). Likewise, if $\mu \in \lambda_n - W_{n-2}$, then $\mu = \lambda_n - \mu_k$ for some $1 \leq k \leq \ell$ and so $\mu = \mu_{2\ell-k+1}$. Therefore,

$$\tau_n(\mu) = \tau_n(\mu_{2\ell-k+1}) = \mu_{2\ell-(2\ell-k+1)+1} = \mu_k = \lambda_n - (\lambda_n - \mu_k) = \lambda_n - \mu.$$

So for all $1 \leq k \leq 2\ell$, $\tau_n(\mu_k) = \lambda_n - \mu_k$ and $\tau_n(\lambda_n - \mu_k) = \mu_k$ since $\tau_n^{-1} = \tau_n$. Therefore, $\tau_n(\sigma_j(\tau_n(\mu_k))) = \tau_n(\sigma_j(\lambda_n - \mu_k)) = \tau_n(\lambda_n - \sigma_j(\mu_k)) = \sigma_j(\mu_k)$.

So for all μ, we have $\tau_n(\sigma_j(\tau_n(\mu))) = \sigma_j(\mu)$. □

As an aside, let us draw attention to the useful fact:

$$\mu_{2\ell-k+1} = \mu_{\tau_n(k)} = \lambda_n - \mu_k$$

for $1 \leq k \leq 2\ell$.

We can now begin the task of determining the action of each simple reflection. First, consider $1 \leq k < n$. Suppose we have already determined σ_k on $W_{n-1} = \{\mu_1, \ldots, \mu_{2\ell}\}$. We need to determine how σ_k acts on $\lambda_{n+1} - W_{n-1}$. Since $k \neq n+1$, $\sigma_k(\lambda_{n+1}) = \lambda_{n+1}$. Thus for $1 \leq j \leq 2\ell$,

$$\begin{aligned} \sigma_k(\mu_{2\ell+j}) &= \sigma_k(\lambda_{n+1} - \mu_{4\ell-(2\ell+j)+1}) = \lambda_{n+1} - \sigma_k(\mu_{2\ell-j+1}) = \lambda_{n+1} - \sigma_k(\mu_{\tau_n(j)}) \\ &= \lambda_{n+1} - \mu_{\sigma_k(\tau_n(j))} = \mu_{4\ell-\sigma_k(\tau_n(j))+1} = \mu_{2\ell+\tau_n(\sigma_k(\tau_n(j)))}. \end{aligned}$$

Therefore by Lemma 3.1, $\sigma_k(\mu_{2\ell+j}) = \mu_{2\ell+\sigma_k(j)}$. Summarizing this discussion we present the following lemma.

Lemma 3.2. *Given the action of σ_k, $k = 1, \ldots, n-1$ on W_{n-1}, we have for any $1 \leq j \leq 2^{n-1}$,*

$$\sigma_k(2^{n-1} + j) = 2^{n-1} + \sigma_k(j).$$

So if σ_k contains the transposition (a, b) when acting on W_k, then σ_k must contain the transpositions $(p2^k + a, p2^k + b)$ for $p = 0, \ldots, 2^{n-k} - 1$ when acting on W_n.

This leaves us to determine σ_n's action on W_n. We have already determined σ_1 acting on W_1 and σ_2 acting on W_2. Therefore, let us assume $n \geq 3$.

$$\begin{aligned} W_n &= W_{n-1} \cup (\lambda_{n+1} - W_{n-1}) \\ &= W_{n-2} \cup (\lambda_n - W_{n-2}) \cup (\lambda_{n+1} - \lambda_n + W_{n-2}) \cup (\lambda_{n+1} - W_{n-2}) \\ &= \{\mu_1, \ldots, \mu_\ell\} \cup \{\mu_{\ell+1}, \ldots, \mu_{2\ell}\} \cup \{\mu_{2\ell+1}, \ldots, \mu_{3\ell}\} \cup \{\mu_{3\ell+1}, \ldots, \mu_{4\ell}\}. \end{aligned}$$

We consider each of these 4 parts of W_n separately. Note that $\sigma_n(\lambda_n) = \lambda_n - \alpha_n = \lambda_n - (-\lambda_{n-1} + 2\lambda_n - \lambda_{n+1}) = \lambda_{n-1} - \lambda_n + \lambda_{n+1}$ (we are assuming $n \geq 3$ so $\alpha_n = -\lambda_{n-1} + 2\lambda_n - \lambda_{n+1}$).

Case 1: Let $1 \leq j \leq \ell$, then $\mu_j \in W_{n-2}$. We know that W_{n-2} does not involve λ_n so $\sigma_n(\mu_j) = \mu_j$.

Case 2: Let $\ell + 1 \leq j \leq 2\ell$, then $\mu_j = \lambda_n - \mu_{\tau_n(j)}$ where $\mu_{\tau_n(j)} \in W_{n-2}$. Therefore, $\sigma_n(\mu_j) = \sigma_n(\lambda_n) - \sigma_n(\mu_{\tau_n(j)}) = (\lambda_{n-1} - \lambda_n + \lambda_{n+1}) - \mu_{\tau_n(j)} = \lambda_{n+1} - (\lambda_n - (\lambda_{n-1} - \mu_{\tau_n(j)})) = \lambda_{n+1} - (\lambda_n - \mu_{\tau_{n-1}(\tau_n(j))}) = \lambda_{n+1} - \mu_{\tau_n(\tau_{n-1}(\tau_n(j)))} = \mu_{\tau_{n+1}(\tau_n(\tau_{n-1}(\tau_n(j))))}$.

$\tau_{n+1}(\tau_n(\tau_{n-1}(\tau_n(j)))) = \tau_{n+1}(\tau_n(\tau_{n-1}(2\ell - j + 1))) = \tau_{n+1}(\tau_n(\ell - (2\ell - j + 1) + 1)) = \tau_{n+1}(\tau_n(j - \ell)) = \tau_{n+1}(2\ell - (j - \ell) + 1) = \tau_{n+1}(3\ell - j + 1) = 4\ell - (3\ell - j + 1) + 1 = j + \ell$.

Therefore, $\sigma_n(\mu_j) = \mu_{j+\ell}$.

Case 3: Since σ_n is its own inverse, we can conclude from Case 2 that $\sigma_n(\mu_j) = \mu_{j-\ell}$ for $2\ell+1 \leq j \leq 3\ell$. Alternatively, we could just do the direct computation. Let $2\ell+1 \leq j \leq 3\ell$, then $\mu_j = \lambda_{n+1} - \mu_{\tau_{n+1}(j)} = \lambda_{n+1} - (\lambda_n - \mu_{\tau_n(\tau_{n+1}(j))})$. Therefore, $\sigma_n(\mu_j) = \sigma_n(\lambda_{n+1}) - \sigma_n(\lambda_n) + \sigma_n(\mu_{\tau_n(\tau_{n+1}(j))}) = \lambda_{n+1} - (\lambda_{n-1} - \lambda_n + \lambda_{n+1}) + \mu_{\tau_n(\tau_{n+1}(j))} = \lambda_n - (\lambda_{n-1} - \mu_{\tau_n(\tau_{n+1}(j))}) = \lambda_n - \mu_{\tau_{n-1}(\tau_n(\tau_{n+1}(j)))} = \mu_{\tau_n(\tau_{n-1}(\tau_n(\tau_{n+1}(j))))}$.

$\tau_n(\tau_{n-1}(\tau_n(\tau_{n+1}(j)))) = \tau_n(\tau_{n-1}(\tau_n(4\ell-j+1))) = \tau_n(\tau_{n-1}(2\ell-(4\ell-j+1)+1)) = \tau_n(\tau_{n-1}(j-2\ell)) = \tau_n(\ell-(j-2\ell)+1) = \tau_n(3\ell-j+1) = 2\ell-(3\ell-j+1)+1 = j-\ell$.

Therefore, $\sigma_n(\mu_j) = \mu_{j-\ell}$.

Case 4: Let $3\ell+1 \leq j \leq 4\ell$, then $\mu_j = \lambda_{n+1} - \mu_{\tau_{n+1}(j)}$ where $\mu_{\tau_{n+1}(j)} \in W_{n-2}$. Therefore, $\sigma_n(\mu_{\tau_{n+1}(j)}) = \mu_{\tau_{n+1}(j)}$ so $\sigma_n(\mu_j) = \sigma_n(\lambda_{n+1} - \mu_{\tau_{n+1}(j)}) = \sigma(\lambda_{n+1}) - \sigma_n(\mu_{\tau_{n+1}(j)}) = \lambda_{n+1} - \mu_{\tau_{n+1}(j)} = \mu_{\tau_{n+1}(\tau_{n+1}(j))} = \mu_j$. So just as in Case 1, $\sigma_n(\mu_j) = \mu_j$.

We have shown the following for $1 \leq k \leq \ell = 2^{n-2}$, $\sigma_n(2^{n-2}+k) = \sigma_n(\ell+k) = 2\ell+k = 2^{n-1}+k$.

Lemma 3.3. *For $n \geq 3$, as a permutation on W_n we have $\sigma_n = \prod_{k=1}^{2^{n-2}} (2^{n-2}+k, 2^{n-1}+k)$.*

Therefore, putting together our base case calculations, lemma 3.2, and lemma 3.3 we get the following theorem.

Theorem 3.4. *The simple reflections of the Weyl group $\mathfrak{W}(B_n)$ act as follows on W_n (the weights of the minuscule representation $L(B_n, \lambda_1)$):*

$$\sigma_1 = \prod_{k=1}^{2^{n-1}} (2k-1, 2k) \qquad \text{and} \qquad \sigma_j = \prod_{p=0}^{2^{(n-j)}-1} \prod_{k=1}^{2^{j-2}} (p2^j + 2^{j-2} + k, p2^j + 2^{j-1} + k), \quad 2 \leq j \leq n.$$

4 Seeing irreducibility from cycle structures

The original motivation for this project was to extend a result found in [CMS]. In that paper the authors present a constructive method for solving the inverse problem in differential Galois theory. As a part of their construction the authors required an irreducible representation for each finite dimensional simple Lie algebra. But to make everything work, they also required that the irreducibility of this representation be visible from examining the cycle structures of the Weyl group elements acting as permutations on the weights of this representation.

This last requirement is quite harsh. To be able to conclude a representation is irreducible from cycle structures, we would first need to know that all of the weight spaces were one dimensional and we would at the very least need all of the weights to lie in a single orbit. Therefore, the only representations that could possibly work are the minuscule representations.

In [CMS] the authors were able to show that each of algebra of type A_n ($n \geq 1$), C_n ($n \geq 3$), D_n ($n \geq 4$), E_6, and E_7 had such a minuscule representation. Since E_8, F_4, and G_2 have no minuscule representations at all, they must be discarded. This left type B_n as the final case to be considered. Using calculations performed in Maple (a computer algebra system), the authors were able to show that B_2, B_3, B_5, and B_7 have a conforming minuscule representation. They also showed that B_4's irreducibility cannot be seen from cycle structures alone. The fate of the other type B_n cases was left open.

Using theorem 3.4 and GAP ("Groups, Algorithms, and Programming" – mathematical software) [GAP], for $n \leq 12$, we were able to find complete lists of cycle structures for the elements in $\mathfrak{W}(B_n)$ viewed as permutations of weights of the minuscule module. These lists allowed us to conclude that the cycle structures for types B_n ($n = 1, 2, 3, 5$, and 7) only allow for invariant subspaces of dimension 0 and 2^n (thus the corresponding representation must be irreducible). The same could not be concluded for other values of n. Below we elaborate on our method for determining irreducibility from cycle structures by examining the cycle structures of B_n ($n = 1, 2, 3, 4$, and 5).

Note that, viewed as permutations, $\mathfrak{W}(B_1) = \{(1), (12)\}$. For our purposes we describe the cycle structures in this group by $1 + 1$ for the identity (two 1-cycles) and 2 for the transposition (12) (a single 2-cycle). This identification allows us to read off the possible dimensions of invariant subspaces allowed by each cycle structure. If we can find a cycle structure (or a collection of cycle structures) that only allows for dimensions of 0 and 2^n we know we can conclude irreducibility from the cycle structures alone. In this case, the 2-cycle structure guarantees the irreducibility of our minuscule representation. We will understand why after the following examples.

When $n = 2$, we have $\mathfrak{W}(B_2) = \big\langle\ (12)(34),\ (23)\ \big\rangle$ with cycle structures

$$1+1+1+1 \quad = \quad 1+1+2 \quad = \quad 2+2 \quad = \quad 4.$$

So every element in $\mathfrak{W}(B_2)$ viewed as a permutation is of the form: four 1-cycles, two 1-cycles and a 2-cycle, two 2-cycles or a 4-cycle. Any partial sum of a type of cycle structure is a possible dimension for an invariant subspace of our minuscule representation allowed by that cycle structure. So the cycle structure $1+1+2$ allows for possible dimensions of $0, 1, 2, 3 = 1+2$, and $4 = 1+1+2$. However, the pair of cycles: $2+2$ only allows dimensions $0, 2$, and $4 = 2+2$. Critically, we also have that the cycle structure 4 (a 4-cycle) allows for dimensions of only 0 and 4. Hence, we conclude that any invariant subspace of our minuscule representation must be of dimension 0 or 4. So irreducibility of our minuscule representation is visible from examining cycle structures alone.

Next $\mathfrak{W}(B_3) = \big\langle\ (12)(34)(56)(78),\quad (23)(67),\quad (35)(46)\ \big\rangle$ and has cycle structures

$$\begin{aligned} 1+1+\cdots+1 \quad &= \quad 1+1+1+1+2+2 \quad = \quad 1+1+3+3 \\ = \quad 2+2+2+2 \quad &= \qquad 2+6 \qquad = \qquad 4+4. \end{aligned}$$

In this case there is no structure of the form $2^3 = 8$ to guarantee irreducibility. Instead we must consider the structures $2+6$ and $4+4$ simultaneously: $2+6$ allows for the possible dimensions $0, 2, 6$, and 8 while $4+4$ allows for $0, 4$, and 8. These lists of possible dimensions of invariant subspaces intersect at just 0 and 8. Hence, irreducibility follows from cycle structures.

The first case in which this method fails is that of $n = 4$.

$$\begin{aligned} \mathfrak{W}(B_4) = \big\langle\ &(12)(34)\cdots(15,16),\quad (23)(67)(10,11)(14,15), \\ &(35)(46)(11,13)(12,14),\quad (59)(6,10)(7,11)(8,12)\ \big\rangle \end{aligned}$$

In this realization of $\mathfrak{W}(B_4)$ we find the following cycle structures:

$$\begin{array}{rcccccc} & 1+1+\cdots+1 & = & 1+1+\cdots+1+2+2+2+2 & & \\ = & 1+1+2+4+4+4 & = & 1+1+1+1+3+3+3+3 & & \\ = & 2+2+\cdots+2 & = & 1+1+1+1+2+2+\cdots+2 & & \\ = & 2+2+6+6 & = & 4+4+4+4 & = & 8+8. \end{array}$$

Each of these cycle structures allows for an invariant subspace of dimension 8. So even though B_4's minuscule module is irreducible, cycle structures alone will not reveal this to us.

For B_5, we have that $\mathfrak{W}(B_5)$ has cycles structures of the form $8+8+8+8$ and $2+10+10+10$. $8+8+8+8$ only allows for submodules of dimensions 0, 8, 16, 24, and 32 whereas $2+10+10+10$ only allows for submodules of dimensions 0, 2, 10, 12, 20, 22, 30, and 32. Thus, only 0 and 32 are allowed, so irreducibility follows.

Below is a table summing up the results for ranks $6 \leq n \leq 12$. We see that the cycle structures for B_7 imply the irreducibility of its minuscule representation.

Rank	Invariant subspace dimensions allowed by cycle structures
6	0, 24, 40, 64
7	0, 128
8	0, 16, 32, 112, 128, 144, 224, 240, 256
9	0, 144, 224, 288, 368, 512
10	0, 64, 144, 224, 240, 320, 400, 464, 480, 544, 560, 624, 704, 784, 800, 880, 960, 1024
11	0, 288, 464, 528, 640, 704, 1344, 1408, 1520, 1584, 1760, 2048
12	0, 48, 112, 176, 224, 288, 352, 400, 464, 528, 576, 640, 704, 752, 816, 880, 928, 992, 1056, 1104, 1168, 1232, 1280, 1344, 1408, 1456, 1520, 1584, 1632, 1696, 1760, 1808, 1872, 1936, 1984, 2048, 2112, 2160, 2224, 2288, 2336, 2400, 2464, 2512, 2576, 2640, 2688, 2752, 2816, 2864, 2928, 2992, 3040, 3104, 3168, 3216, 3280, 3344, 3392, 3456, 3520, 3568, 3632, 3696, 3744, 3808, 3872, 3920, 3984, 4048, 4096

We were not able to get GAP to complete calculations for any higher rank cases. The problem is that Weyl groups grow very fast as rank is increased. In fact $\mathfrak{W}(B_n)$ is isomorphic to a semi-direct product of S_n and $(\mathbb{Z}_2)^n$ so that $|\mathfrak{W}(B_n)| = 2^n \cdot n!$. Even at rank 12 we have a group of order $2^{12} \cdot 12!$ acting on a set of $2^{12} = 4096$ weights! However, by randomly sampling $\mathfrak{W}(B_n)$ for ranks of up to $n = 23$, we obtained strong evidence that the number of allowed invariant subspace dimensions blows up as rank is increased. We conjecture that the irreducibility of the minuscule representation cannot be seen from cycle structures alone after rank 7. We found this quite surprising given the nature of the minuscule representations for the other types of algebras.

5 Appendix: GAP Code

This code was run in GAP version 4.4.12 [GAP] on a Mac (OS X Version 10.7.4) with 2.5 GHz Intel Core i5 processor and 4 GB of RAM. The generators created by “BminGenerators” are exactly those found in theorem 3.4.

```
#
# This function returns a list of n permutations which represent the simple
# reflections of the Weyl group of type B rank n acting on the weights of
# its minuscule representation. These generate the permutation representation
# of the Weyl group of type B rank n.
#
# s[i] corresponds to the simple reflection accross the hyperplane determined
# by the simple root alpha[i]. Since reflections are involutions (order 2),
# each permutation is the product of disjoint transpositions.
#
BminGenerators := function(n)
local s,i,tmp,j,k;

s := ListWithIdenticalEntries(n,[]);;
for i in [1..n-1] do
   tmp := ListWithIdenticalEntries(2^n,0);
   for j in [1..2^n] do
      tmp[j] := j;
   od;
   for j in [1..2^(n-1-i)] do
      for k in [1..2^(i-1)] do
         tmp[2^(i-1)+1+(j-1)*2^(i+1)+(k-1)] := 2^(i-1)+1+(j-1)*2^(i+1)+(k-1)+2^(i-1);
         tmp[2^(i-1)+1+(j-1)*2^(i+1)+(k-1)+2^(i-1)] := 2^(i-1)+1+(j-1)*2^(i+1)+(k-1);
      od;
   od;
   s[n-i] := PermList(tmp);
od;;

tmp := ListWithIdenticalEntries(2^n,0);;
for j in [1..2^(n-1)] do
   tmp[2*j-1] := 2*j;
   tmp[2*j] := 2*j-1;
od;;
s[n] := PermList(tmp);;

return s;

end;;



#
# Given a permutation s and rank n, this function determines how many of each
# type of cycle appears in s. Since we want to keep track of 1-cycles (which
# are normally suppressed), we need the rank to find out how many integers in
# the list 1..2^n are unmoved (i.e. the number of 1-cycles).
#
# This function returns a list of pairs of the form "[k,m]" which indicates
# that the permutation has m k-cycles.
#
# For example: s=(1,2,3)(4,5,6) and n=4 means s=(1,2,3)(4,5,6)(7)(8)...(16)
# so the function returns [[1,16],[3,2]] (16 1-cycles and 2 3-cycles).
#
CycleType := function(s,n)
local tmp,lst,i,z;

# CycleStructurePerm returns a list of the number of cycles of each type
# starting with transpositions.
tmp := CycleStructurePerm(s);

# lst = [0,tmp]
# The "0" will be replaced by the number of 1-cycles.
lst := [0];;
Append(lst,tmp);;

# This replaces empty spots in lst with 0’s.
z := Zero([1..Length(lst)]);
lst := lst+z;

tmp := 0;;
for i in [1..Length(lst)] do
   # i*lst[i] is the number of integers moved by the i-cycles.
   tmp := tmp+i*lst[i];
od;
# tmp is the total number of integers in 1..2^n moved by non-trivial cycles,
# so 2^n-tmp is the number of 1-cycles (trivial cycles).
lst[1] := 2^n-tmp;;

# This converts our list of numbers of k-cycles to a more convenient format.
# If list[k]=m > 0 then we add "[k,m]" to our list signifying that there
# are a total of m k-cycles. So [3,5,0,7] turns into [[1,3],[2,5],[4,7]].
tmp := [];
for i in [1..Length(lst)] do
   if not lst[i] = 0 then
      Append(tmp,[[i,lst[i]]]);
   fi;
od;;
lst := tmp;

return lst;
```

```
end;;


#
# This function returns the distinct cycle types that appear in the minuscule
# permutation representation of the Weyl group of type B rank n.
#
# For example: When n=2, we get [[[1,2],[2,1]], [[1,4]], [[2,2]], [[4,1]]].
# This means that the permutation representation contains permutations of the
# form... (A) 2 1-cycles and a transposition, (B) 4 1-cycles (the identity),
# (C) 2 tranpositions, and (D) 1 4-cycle.
#
BminCycleTypes := function(n)
local ccl,csl,cycTypes,k;

# This the a complete list of the conjugacy classes of our perm. rep.
ccl := ConjugacyClasses(Group(BminGenerators(n)));;

# csl is a list of representatives -- one from each conjugacy class.
csl := List(ccl, c -> Representative(c));;

# We compute the cycle type of each representative in csl and add it to our
# list of cycle types: cycTypes.
cycTypes := [];;
for k in [1..Length(csl)] do
   Append(cycTypes,[CycleType(csl[k],n)]);
od;

# Elements of two distinct conjugacy classes can share the same cycle type.
# Thus we apply SSortedList to remove redundancies in our list.
return SSortedList(cycTypes);

end;;



#
# We know that the Weyl group acts transitively on the set of weights of a
# minuscule representation. So there are no non-empty proper subsets of
# weights left invariant under the group's action. In some cases, this is
# visible from the cycle structures (of the Weyl group elements realized
# as permutations) alone.
#
# This function returns a list of sizes of invariant subsets of weights
# allowed by the cycle structures of the perm. rep. of the Weyl group of
# type B rank n acting on the weights of its minuscule representation.
#
```

```
BminInvSubspDim := function(n)
local cycTypes,subsp,m,elt,myList,i,indicesOfInterest,j,k,tmp,subspTMP;

# Get the cycle types for the perm. rep.
cycTypes := BminCycleTypes(n);

# no elements (yet) ==> all subset sizes are allowed.
subsp := [0..2^n];;

for m in [1..Length(cycTypes)] do

   # grab a cycle type.
   elt := cycTypes[m];

   # myList is a list of 2^n+1 copies of "false". myList[i+1] corresponds
   # to an allowed invariant subset of size i.
   myList := ListWithIdenticalEntries(2^n+1,false);;

   # the empty set is always allowed.
   myList[1] := true;;

   for i in [1..Length(elt)] do   # i-th type of cycle in elt

      # look through myList and grab only the indices y for which myList[y] is true.
      indicesOfInterest := Filtered([1..Length(myList)], y -> myList[y]);

      for j in indicesOfInterest do   # all j's where myList[j]=true
         # If elt[i]=[x,y], then elt has y x-cycles, so k goes from 1 to y which
         # happens to be the number of x-cycles.
         for k in [1..elt[i][2]] do
            # Suppose elt[i]=[x,y]. We know myList[j]=true (an invariant
            # subset of size j is allowed by elt). If we let in anything from
            # an x-cycle, we must allow all x elements from that cycle. So
            # if j is allowed, then so is j+x (but nothing between j and j+x).
            # Looping through all y x-cycles, we get j,j+x,j+2x,...,j+yx are all
            # allowed.
            myList[j+k*elt[i][1]] := true;
         od;
      od;
   od;

   # tmp is a list of indices corresponding to invariant subset sizes allowed
   # by the permutation elt.
   tmp := Filtered([1..Length(myList)], y -> myList[y]);;

   # since the y-th element corresponded to a set of size y-1 we need to decrease
   # everything in tmp by 1.
   tmp := List(tmp, p -> p-1);

   # subspTMP is the list of common subset sizes allowed by previous elements.
   subspTMP := subsp;

   subsp := [];
   for i in tmp do   # size "i" is allowed by elt (it appears in tmp)

     # If size "i" was allowed by all previous elements, we should add it
     # to our list of allowed sizes.
     if i in subspTMP then
        Add(subsp,i);
     fi;
   od;
od;;

return subsp;  # The listed sizes were allowed by all of the cycle types in cycTypes.

end;;


#
# This returns in the order of the Weyl group of type B rank n.
#
BWeylSize := function(n)
   return(Size(Group(BminGenerators(n))));
end;;

#
# This returns a permutation representing the Coxeter element of the Weyl group
# of type B rank n. This is just the product of all of the simple reflections:
# s[1]s[2]...s[n].
#
BminCoxeter := function(n)
local s,coxeter,i;

s := BminGenerators(n);

coxeter := (1);;
for i in [1..n] do
   coxeter := coxeter*s[i];
od;;

return coxeter;
```

```
end;;

#
# Let's see what dimensions are allowed for the first 12 ranks...
#
for n in [1..12] do
  Print("B",n,": ",BminInvSubspDim(n),"\n");
od;
```

References

[B] N. Bourbaki, *Lie Groups and Lie Algebras* (Springer, 2005), Chaps. 79.

[C] Roger Carter, *Lie algebras of finite and affine type*, Cambridge Studies in Advanced Mathematics, vol. 96, Cambridge University Press, Cambridge, 2005.

[CMS] Cook, William J.; Mitschi, Claude; Singer, Michael F. On the constructive inverse problem in differential Galois theory. *Comm. Algebra* **33** (2005) 3639–3665.

[EW] Erdmann, K., Wildon, M. J. (2006). *Introduction to Lie algebras*, Springer Undergraduate Mathematics Series, Springer-Verlag London Ltd., London.

[GAP] The GAP Group, GAP – Groups, Algorithms, and Programming, Version 4.7.4; 2014. (http://www.gap-system.org).

[G] R. M. Green, *Combinatorics of minuscule representations*, Cambridge Tracts in Mathematics, vol. 199, Cambridge University Press, Cambridge, 2013.

[H] J. Humphreys, *Introduction to Lie Algebras and Representation Theory* (Springer-Verlag, 1972).