The BGG complex

1 Introduction

Let $\mathfrak{g}$ be a simple Lie algebra for example $\mathfrak{g}$ and let $\mathfrak{n}$ be the lower part (i.e $\mathfrak{g} = \mathfrak{u} \oplus \mathfrak{h} \oplus \mathfrak{n}$ and $\mathfrak{u}$ has positive weights). We write f_i for the generators of $U(\mathfrak{n})$, where $i \in I$ runs over the simple roots. If W is the Weyl group of $\mathfrak{g}$, the dot action of W on R (the root lattice) is defined by

$$s_i \cdot \lambda = s_i(\lambda + \rho) - \rho$$

2 Modules needed for the center

Let i, j, k with $i + j + k = 0$ and $0 \leq i, j \leq \dim \mathfrak{n}$ and k even ($\dim(\mathfrak{n})$ is the number of positive roots of $\mathfrak{g}$).

To construct the module we look first at $M_j := \wedge^j_{\mathrm{Sym}(\mathfrak{u})}(\mathrm{Sym}(\mathfrak{u}) \otimes \mathfrak{g} \oplus \mathrm{Sym}(\mathfrak{u}) \otimes \mathfrak{n})$, where the wedge is a wedge of Sym($\mathfrak{u}$)-algebra. For example for $j = 3$ we get

$$\mathrm{Sym}(\mathfrak{u}) \otimes \wedge^3 \mathfrak{g} \oplus \mathrm{Sym}(\mathfrak{u}) \otimes \wedge^2 \mathfrak{g} \otimes \mathfrak{n} \oplus \mathrm{Sym}(\mathfrak{u}) \otimes \mathfrak{g} \otimes \wedge^2 \mathfrak{n} \oplus \mathrm{Sym}(\mathfrak{u}) \otimes \wedge^3 \mathfrak{n}$$

The structure of module is as follows : $\mathfrak{g}, \mathfrak{n}$ get the adjoint action and $\mathfrak{u}$ get the coadjoint action (and then we extend like usual on tensor products/symmetric products/wedge products).

We now define a degree as follows : $\deg(\mathfrak{g}) = \deg(\mathfrak{b}) = 0$, $\deg(\mathfrak{n}) = -2$ and $\deg(\mathfrak{u}) = 2$, extending the natural way on tensor product. For example, the degree of $\mathrm{Sym}^4(\mathfrak{u}) \otimes \wedge^2 \mathfrak{g} \oplus \wedge^2 \mathfrak{n}$ is 4. Clearly the action of $\mathfrak{n}$ preserves the degree, so $M_j = \oplus_r M_j^r$ where each component has degree r (here $r \in \mathbb{Z}$).

For example, for $j = 3$ we wrote the module before, if say $r = -4$ the component $\mathrm{Sym}(\mathfrak{u}) \otimes \wedge^3\mathfrak{g}$ will not contribute because the degree is positive. Similarly, the degree of $\mathrm{Sym}(\mathfrak{u})\otimes\wedge^2\mathfrak{g}\otimes\mathfrak{n}$ is greater than -2. For the third component $\mathrm{Sym}(\mathfrak{u})\otimes\mathfrak{g}\otimes\wedge^2\mathfrak{n}$, we get $\mathfrak{g}\otimes\wedge^2\mathfrak{n}$. Finally the last component $\mathrm{Sym}(\mathfrak{u})\otimes\wedge^3\mathfrak{n}$ gives $\mathfrak{u}\otimes\wedge^3\mathfrak{n}$. So we get $M_3^{-4} = \mathfrak{g}\otimes\wedge^2\mathfrak{n}\oplus\mathfrak{u}\otimes\wedge^3\mathfrak{n}$.

Our modules will be $E_j^k := M_j^k/T$ where T are some relations that we describe now. We have a subspace $R \subset \mathfrak{g} \oplus \mathfrak{u} \otimes \mathfrak{n}$ given by the image of $\phi : \mathfrak{b} \to \mathfrak{g} \oplus \mathfrak{u} \otimes \mathfrak{n}$ defined before.

Now we notice that there is a well defined map $\mathfrak{g} \otimes M_j \to M_{j+1}$ given by wedging with $\mathfrak{g}$ (or taking the tensor product for $\mathrm{Sym}(\mathfrak{u}) \otimes \wedge^j\mathfrak{n}$). More precisely, there is a map

$$\mathfrak{g} \otimes \mathrm{Sym}(\mathfrak{u}) \otimes \wedge^r\mathfrak{g} \otimes \wedge^{j-r}\mathfrak{n} \to \mathrm{Sym}(\mathfrak{u}) \otimes \wedge^{r+1}\mathfrak{g} \otimes \wedge^{j-r}\mathfrak{n}$$

if $r \geq 1$, and there is also a map $\mathfrak{g} \otimes \mathrm{Sym}(\mathfrak{u}) \otimes \wedge^j\mathfrak{n} \to \mathrm{Sym}(\mathfrak{u}) \otimes \mathfrak{g} \otimes \wedge^j\mathfrak{n}$. We take the direct sum of all these maps for $r = 0, 1, \ldots, j$ and we get a map $\mathfrak{g} \otimes M_j \to M_{j+1}$.

Similarly, there is a well defined map

$$(\mathfrak{u} \otimes \mathfrak{n}) \otimes \mathrm{Sym}(\mathfrak{u}) \otimes \wedge^r\mathfrak{g} \otimes \wedge^{j-r}\mathfrak{n} \to \mathrm{Sym}^{\bullet+1}(\mathfrak{u}) \otimes \wedge^r\mathfrak{g} \otimes \wedge^{j-r+1}\mathfrak{n}$$

When $r = j$ the map is $(\mathfrak{u} \otimes \mathfrak{n}) \otimes \mathrm{Sym}(\mathfrak{u}) \otimes \wedge^j\mathfrak{g} \to \mathrm{Sym}^{\bullet+1}(\mathfrak{u}) \otimes \wedge^j\mathfrak{g} \otimes \mathfrak{n}$.

The direct sum of these two maps gives a map $R \otimes M_j \to M_{j+1}$. We define T_{j+1}, be the image of this map, to be the relations in M_{j+1}. So for each i, j we get a module $E_j^k = M_j^k/T_j$.

3 Admissible weights, case $i = 0$

Now let us write $E := E_j^k$. Since E is a $\mathfrak{h}$-module in particular there is a basis of homogeneous elements.

Definition 3.1. *A weight μ is dot-regular if the dot-stabilizer is trivial. A weight μ is dominant if $\langle \mu, \alpha^\vee \rangle \geq 0$ for all simple roots α.*

Proposition 3.2. *If μ is dot-regular there is a unique element $w \in W$ so that $w \cdot \mu$ is dominant.*

Remark : I don't know a good algorithm to find such an element.
Now the algorithm to compute the cohomology is as follows :

- List all the weights of E, put in list1.
- Delete the dot-singular weights from list1.
- Create listdom and listNdom. For each μ in list1 transfer to listdom or listNdom if dominant or not.
- For each μ in listNdom find $w \in W$ so that $w \cdot \mu$ is dominant. Delete μ from listNdom if $\ell(w) > 1$, else keep it (and store w since we need it later).
- Create another list "cohomology".
- For each μ in listdom, test if there exists μ' in listNdom so that $w \cdot \mu' = \mu$
- If such μ' exists, runs BGG complex for E with weight μ and add H^0 to "cohomology".
- If no such μ' exists, add $E[\mu]$ to "cohomology".

4 Admissible weights, case $i > 0$

The algorithm is as follows :

- Compute all the weights of E, put it in list1.
- Delete the dot-singular weights from list1.
- Also delete the dominant weights from list1.
- For each μ in list1, compute w with $w \cdot \mu$ dominant. Delete if $\ell(w) \neq i$.
- Create a list "cohomology".
- For each remaining weight μ, compute the i-th cohomology of $\mathrm{BGG}(E, w \cdot \mu)$ and add it to "cohomology".

As before the list "cohomology" is the one we wanted to compute.