A non-integral Kazhdan-Lusztig algorithm

Qixian Zhao

Yau Mathematical Sciences Center, Tsinghua University

Oct 14, 2023

Table of Contents

Table of Contents

Setup

- $\mathfrak{g}$ - semisimple Lie algebra over $\mathbb{C}$
- $\mathcal{C}$ - a nice category of representations of $\mathfrak{g}$.

Setup

- $\mathfrak{g}$ - semisimple Lie algebra over $\mathbb{C}$
- $\mathcal{C}$ - a nice category of representations of $\mathfrak{g}$.
  Examples:
  - $\mathcal{C} = \mathcal{O}'_\lambda = \text{Mod}_{fg}(\mathfrak{g}, N)_\lambda$ the Category $\mathcal{O}'$ with infinitesimal character χ_λ

Setup

- $\mathfrak{g}$ - semisimple Lie algebra over $\mathbb{C}$
- $\mathcal{C}$ - a nice category of representations of $\mathfrak{g}$.
  Examples:
  - $\mathcal{C} = \mathcal{O}'_\lambda = \mathrm{Mod}_{fg}(\mathfrak{g}, N)_\lambda$ the Category $\mathcal{O}'$ with infinitesimal character χ_λ
  - $\mathcal{C} = \mathrm{Mod}_{fg}(\mathfrak{g}, N, f)_\lambda$ the category of Whittaker modules

Setup

- $\mathfrak{g}$ - semisimple Lie algebra over $\mathbb{C}$
- $\mathcal{C}$ - a nice category of representations of $\mathfrak{g}$.
  Examples:
  - $\mathcal{C} = \mathcal{O}'_\lambda = \mathrm{Mod}_{fg}(\mathfrak{g}, N)_\lambda$ the Category $\mathcal{O}'$ with infinitesimal character χ_λ
  - $\mathcal{C} = \mathrm{Mod}_{fg}(\mathfrak{g}, N, f)_\lambda$ the category of Whittaker modules
  - $\mathcal{C} = \mathrm{Mod}_{fg}(\mathfrak{g}, K)_\lambda$ the category of $(\mathfrak{g}, K)$-modules (representations of real groups)

Setup

- $\mathfrak{g}$ - semisimple Lie algebra over $\mathbb{C}$
- $\mathcal{C}$ - a nice category of representations of $\mathfrak{g}$.
  We will focus on:
  - $\mathcal{C} = \mathcal{O}'_\lambda = \mathrm{Mod}_{fg}(\mathfrak{g}, N)_\lambda$ the Category $\mathcal{O}'$ with infinitesimal character λ

Setup - Properties of $\mathcal{C}$

The category $\mathcal{C}$ we are looking at has some nice properties:

Setup - Properties of $\mathcal{C}$

The category $\mathcal{C}$ we are looking at has some nice properties:

- Every object has finite length

Setup - Properties of $\mathcal{C}$

The category $\mathcal{C}$ we are looking at has some nice properties:

- Every object has finite length
- There are finitely many irreducible objects L_w, parameterized by a set $w \in \Xi$

Setup - Properties of $\mathcal{C}$

The category $\mathcal{C}$ we are looking at has some nice properties:

- Every object has finite length
- There are finitely many irreducible objects L_w, parameterized by a set $w \in \Xi$
- Each irreducible L_w is the unique irreducible submodule of a **standard object** I_w, which are much easier to understand.

Setup - An example

- $\mathfrak{g} = \mathfrak{sl}(3, \mathbb{C})$

Setup - An example

- $\mathfrak{g} = \mathfrak{sl}(3, \mathbb{C})$
- $\mathcal{C} = \mathrm{Mod}_{fg}(\mathfrak{g}, N)_\lambda$

Setup - An example

- $\mathfrak{g} = \mathfrak{sl}(3, \mathbb{C})$
- $\mathcal{C} = \text{Mod}_{fg}(\mathfrak{g}, N)_\lambda$
- $\Xi = W$, the Weyl group of $\mathfrak{g}$

Setup - An example

- $\mathfrak{g} = \mathfrak{sl}(3, \mathbb{C})$
- $\mathcal{C} = \mathrm{Mod}_{fg}(\mathfrak{g}, N)_\lambda$
- $\Xi = W$, the Weyl group of $\mathfrak{g}$

s_γ

$s_\alpha s_\beta$ $s_\beta s_\alpha$

s_α s_β

1

The Kazhdan-Lusztig problem

Describe the L_w's in terms of the I_w's, i.e.

The Kazhdan-Lusztig problem

Describe the L_w's in terms of the I_w's, i.e.
Find an expression

$$[L_w] = \sum_{v \in \Xi} c_{wv}[I_v]$$

in the Grothendieck group $K\mathcal{C}$.

Table of Contents

1st step: Localization

- $\mathcal{B}$ - flag variety of $\mathfrak{g}$, with a G-action

1st step: Localization

- $\mathcal{B}$ - flag variety of $\mathfrak{g}$, with a G-action
- $\mathcal{D}_\lambda$ - sheaf of differential operators on $\mathcal{B}$, with twist λ

1st step: Localization

- $\mathcal{B}$ - flag variety of $\mathfrak{g}$, with a G-action
- $\mathcal{D}_\lambda$ - sheaf of differential operators on $\mathcal{B}$, with twist λ
- $\text{Mod}_{coh}(\mathcal{D}_\lambda)$ - category of coherent $\mathcal{D}_\lambda$-modules (i.e. $\mathcal{O}_\mathcal{B}$ quasi-coherent and locally finitely generated over $\mathcal{D}_\lambda$)

1st step: Localization

- $\mathcal{B}$ - flag variety of $\mathfrak{g}$, with a G-action
- $\mathcal{D}_\lambda$ - sheaf of differential operators on $\mathcal{B}$, with twist λ
- $\mathrm{Mod}_{coh}(\mathcal{D}_\lambda)$ - category of coherent $\mathcal{D}_\lambda$-modules (i.e. $\mathcal{O}_\mathcal{B}$ quasi-coherent and locally finitely generated over $\mathcal{D}_\lambda$)

Theorem (Beilinson-Bernstein)

If λ is antidominant regular, then taking global sections is an equivalence of categories

$$\Gamma(X, -): \ \mathrm{Mod}_{coh}(\mathcal{D}_\lambda) \cong \mathrm{Mod}_{fg}(\mathfrak{g})_\lambda.$$

Its inverse $\mathcal{D}_\lambda \otimes_{\mathcal{U}(\mathfrak{g})_\lambda} -$ is called the ***localization functor****.*

1st step: Localization

- $\mathcal{B}$ - flag variety of $\mathfrak{g}$, with a G-action
- $\mathcal{D}_\lambda$ - sheaf of differential operators on $\mathcal{B}$, with twist λ
- $\mathrm{Mod}_{coh}(\mathcal{D}_\lambda)$ - category of coherent $\mathcal{D}_\lambda$-modules (i.e. $\mathcal{O}_\mathcal{B}$ quasi-coherent and locally finitely generated over $\mathcal{D}_\lambda$)

Theorem (Beilinson-Bernstein)

If λ is antidominant regular, then taking global sections is an equivalence of categories

$$\Gamma(X,-):\ \mathrm{Mod}_{coh}(\mathcal{D}_\lambda, N) \cong \mathrm{Mod}_{fg}(\mathfrak{g}, N)_\lambda.$$

Its inverse $\mathcal{D}_\lambda \otimes_{\mathcal{U}(\mathfrak{g})_\lambda} -$ is called the ***localization functor***.

1st step: Localization

$$\begin{array}{c} \mathrm{Mod}_{fg}(\mathfrak{g}, N)_\lambda \\ \downarrow \cong \\ \mathrm{Mod}_{coh}(\mathcal{D}_\lambda, N) \end{array} \qquad \begin{array}{ccc} L_w & \hookrightarrow & I_w \\ \downarrow & & \downarrow \\ \mathcal{L}(w,\lambda) & \hookrightarrow & \mathcal{I}(w,\lambda) \end{array}$$

1st step: Localization

$$
\begin{array}{ccccc}
\mathrm{Mod}_{fg}(\mathfrak{g}, N)_\lambda & \quad & L_w & \hookrightarrow & I_w \\
\downarrow \cong & & \downarrow & & \downarrow \\
\mathrm{Mod}_{coh}(\mathcal{D}_\lambda, N) & & \mathcal{L}(w,\lambda) & \hookrightarrow & \mathcal{I}(w,\lambda) \\
 & & & & \| \\
 & & & & i_{C(w) \hookrightarrow \mathcal{B}, *} \mathcal{O}_{C(w)}
\end{array}
$$

where the $C(w)$'s are N-orbits on $\mathcal{B}$, a.k.a. Schubert cells (parameterized also by W).

$\mathrm{Supp}\, \mathcal{L}(w,\lambda) = \mathrm{Supp}\, \mathcal{I}(w,\lambda) = \overline{C(w)}$.

1st step: Localization

$$
\begin{array}{ccccccc}
\mathrm{Mod}_{fg}(\mathfrak{g}, N)_\lambda & & L_w & \hookrightarrow & I_w & & \\
\downarrow \cong & & \downarrow & & \downarrow & & \\
\mathrm{Mod}_{coh}(\mathcal{D}_\lambda, N) & & \mathcal{L}(w,\lambda) & \hookrightarrow & \mathcal{I}(w,\lambda) & \twoheadrightarrow & \mathcal{K} \\
 & & & & \| & & \\
 & & & & i_{C(w)\hookrightarrow\mathcal{B},*}\mathcal{O}_{C(w)} & &
\end{array}
$$

the cokernel $\mathcal{K}$ is supported on the boundary of $C(w)$.

1st step: Localization

$$
\begin{array}{ccc}
\mathrm{Mod}_{fg}(\mathfrak{g}, N)_\lambda \\
\downarrow \cong \\
\mathrm{Mod}_{coh}(\mathcal{D}_\lambda, N)
\end{array}
\qquad
\begin{array}{ccccc}
L_w & \hookrightarrow & I_w & & \\
\downarrow & & \downarrow & & \\
\mathcal{L}(w,\lambda) & \hookrightarrow & \mathcal{I}(w,\lambda) & \twoheadrightarrow & \mathcal{K} \\
 & & \| & & \\
 & & i_{C(w)\hookrightarrow \mathcal{B},*}\mathcal{O}_{C(w)} & &
\end{array}
$$

the cokernel $\mathcal{K}$ is supported on the boundary of $C(w)$.

$\Longrightarrow$ for the closed orbit $C(1)$, $\mathcal{L}(1,\lambda) = \mathcal{I}(1,\lambda)$.

2nd step: Algorithm (λ integral)

Goal: find a way to obtain info about $\mathcal{L}(w, \lambda)$ from those $\mathcal{L}(v, \lambda)$'s with smaller support.

2nd step: Algorithm (λ integral)

Goal: find a way to obtain info about $\mathcal{L}(w, \lambda)$ from those $\mathcal{L}(v, \lambda)$'s with smaller support.
Suppose we already know $\mathcal{L}(v, \lambda)$.

2nd step: Algorithm (λ integral)

Goal: find a way to obtain info about $\mathcal{L}(w, \lambda)$ from those $\mathcal{L}(v, \lambda)$'s with smaller support.

Suppose we already know $\mathcal{L}(v, \lambda)$.

Find a partial flag variety with 1-dimensional fibers (given by a simple root α:

$$\begin{array}{ccc} \mathbb{P}^1 & \hookrightarrow & \mathcal{B} \\ \downarrow & & \downarrow p_\alpha \\ \{*\} & \hookrightarrow & \mathcal{P}_\alpha \end{array}$$

2nd step: Algorithm (λ integral)

Goal: find a way to obtain info about $\mathcal{L}(w, \lambda)$ from those $\mathcal{L}(v, \lambda)$'s with smaller support.
Suppose we already know $\mathcal{L}(v, \lambda)$.
Find a partial flag variety with 1-dimensional fibers (given by a simple root α:

$$\begin{array}{ccc} C(v) & \hookrightarrow & \mathcal{B} \\ & \searrow & \downarrow p_\alpha \\ & p_\alpha(C(v)) \hookrightarrow & \mathcal{P}_\alpha \end{array}$$

Want: $\dim p_\alpha(C(v)) = \dim C(v)$.

2nd step: Algorithm (λ integral)

Goal: find a way to obtain info about $\mathcal{L}(w, \lambda)$ from those $\mathcal{L}(v, \lambda)$'s with smaller support.
Suppose we already know $\mathcal{L}(v, \lambda)$.
Find a partial flag variety with 1-dimensional fibers (given by a simple root α:

$$\begin{array}{ccc} C(v) & \hookrightarrow & \mathcal{B} \\ & \searrow & \downarrow p_\alpha \\ & p_\alpha(C(v)) \hookrightarrow & \mathcal{P}_\alpha \end{array}$$

Want: $\dim p_\alpha(C(v)) = \dim C(v)$.
(Such α always exists if $C(v)$ is not already the largest orbit in $\mathcal{B}$).

2nd step: Algorithm (λ integral)

Goal: find a way to obtain info about $\mathcal{L}(w,\lambda)$ from those $\mathcal{L}(v,\lambda)$'s with smaller support.

Suppose we already know $\mathcal{L}(v,\lambda)$.

Find a partial flag variety with 1-dimensional fibers (given by a simple root α:

$$\begin{array}{ccccc} & & C(v) \cup C(vs_\alpha) & & \\ & & \| & & \\ C(v) & \hookrightarrow & p_\alpha^{-1}(p_\alpha(C(v))) & \hookrightarrow & \mathcal{B} \\ & \searrow & \downarrow & & \downarrow p_\alpha \\ & & p_\alpha(C(v)) & \hookrightarrow & \mathcal{P}_\alpha \end{array}$$

We say α is **transversal** to $C(v)$.

2nd step: Algorithm (λ integral)

$$
\begin{array}{ccccc}
 & & C(v) \cup C(vs_\alpha) & & \\
 & & \| & & \\
C(v) & \hookrightarrow & p_\alpha^{-1}(p_\alpha(C(v))) & \hookrightarrow & \mathcal{B} \\
 & \searrow & \downarrow & & \downarrow p_\alpha \\
 & & p_\alpha(C(v)) & \hookrightarrow & \mathcal{P}_\alpha
\end{array}
$$

2nd step: Algorithm (λ integral)

$$
\begin{array}{ccccc}
 & & C(v) \cup C(vs_\alpha) & & \\
 & & \| & & \\
C(v) & \hookrightarrow & p_\alpha^{-1}(p_\alpha(C(v))) & \hookrightarrow & \mathcal{B} \\
 & \searrow & \downarrow & & \downarrow p_\alpha \\
 & & p_\alpha(C(v)) & \hookrightarrow & \mathcal{P}_\alpha
\end{array}
$$

Decomposition Theorem [Beilinson-Bernstein-Deligne-Gabber]

$\Longrightarrow$ $p_{\alpha *}\mathcal{L}(v, \lambda)$ is a $\oplus$ of irreducibles

2nd step: Algorithm (λ integral)

$$
\begin{array}{ccccc}
 & & C(v) \cup C(vs_\alpha) & & \\
 & & \| & & \\
C(v) & \hookrightarrow & p_\alpha^{-1}(p_\alpha(C(v))) & \hookrightarrow & \mathcal{B} \\
 & \searrow & \downarrow & & \downarrow p_\alpha \\
 & & p_\alpha(C(v)) & \hookrightarrow & \mathcal{P}_\alpha
\end{array}
$$

Decomposition Theorem [Beilinson-Bernstein-Deligne-Gabber]

$\Longrightarrow$ $p_{\alpha *}\mathcal{L}(v, \lambda)$ is a $\oplus$ of irreducibles

$\rightsquigarrow$ $p_\alpha^* p_{\alpha *}\mathcal{L}(v, \lambda)$ is a $\oplus$ of irreducibles

2nd step: Algorithm (λ integral)

$$
\begin{array}{ccccc}
 & & C(v) \cup C(vs_\alpha) & & \\
 & & \| & & \\
C(v) & \hookrightarrow & p_\alpha^{-1}(p_\alpha(C(v))) & \hookrightarrow & \mathcal{B} \\
 & \searrow & \downarrow & & \downarrow p_\alpha \\
 & & p_\alpha(C(v)) & \hookrightarrow & \mathcal{P}_\alpha
\end{array}
$$

Decomposition Theorem [Beilinson-Bernstein-Deligne-Gabber]

$\Longrightarrow$ $p_{\alpha *}\mathcal{L}(v, \lambda)$ is a $\oplus$ of irreducibles

$\leadsto$ $p_\alpha^* p_{\alpha *}\mathcal{L}(v, \lambda)$ is a $\oplus$ of irreducibles

Note: $\operatorname{Supp} p_\alpha^* p_{\alpha *}\mathcal{L}(v, \lambda) = \overline{C(v) \cup C(vs_\alpha)}$

2nd step: Algorithm (λ integral)

$$\begin{array}{ccccc} & & C(v) \cup C(vs_\alpha) & & \\ & & \| & & \\ C(v) & \hookrightarrow & p_\alpha^{-1}(p_\alpha(C(v))) & \hookrightarrow & \mathcal{B} \\ & \searrow & \downarrow & & \downarrow p_\alpha \\ & & p_\alpha(C(v)) & \hookrightarrow & \mathcal{P}_\alpha \end{array}$$

Decomposition Theorem [Beilinson-Bernstein-Deligne-Gabber]

$\implies$ $p_{\alpha *}\mathcal{L}(v,\lambda)$ is a $\oplus$ of irreducibles

$\rightsquigarrow$ $p_\alpha^* p_{\alpha *}\mathcal{L}(v,\lambda)$ is a $\oplus$ of irreducibles

Note: $\operatorname{Supp} p_\alpha^* p_{\alpha *}\mathcal{L}(v,\lambda) = \overline{C(v) \cup C(vs_\alpha)}$

$\implies$ $\mathcal{L}(vs_\alpha,\lambda) \underset{\oplus}{\subseteq} p_\alpha^* p_{\alpha *}\mathcal{L}(v,\lambda)$, with multiplicity 1.

2nd step: Algorithm (λ integral)

Start with

$$\mathcal{L}(v,\lambda) = \sum_u c_{vu}\mathcal{I}(u,\lambda)$$

(in the *mixed/graded* Grothendieck group $K^m \operatorname{Mod}_{coh}(\mathcal{D}_\lambda, N)$)

2nd step: Algorithm (λ integral)

Start with

$$\mathcal{L}(v,\lambda) = \sum_u c_{vu}\mathcal{I}(u,\lambda)$$

(in the *mixed/graded* Grothendieck group $K^m \operatorname{Mod}_{coh}(\mathcal{D}_\lambda, N)$)

Apply $U_\alpha := p_\alpha^* p_{\alpha *}$:

$$\mathcal{L}(vs_\alpha,\lambda) \oplus (\cdots) = U_\alpha \mathcal{L}(v,\lambda) = \sum_u c_{vu} U_\alpha \mathcal{I}(u,\lambda)$$

2nd step: Algorithm (λ integral)

Start with

$$\mathcal{L}(v,\lambda) = \sum_u c_{vu}\mathcal{I}(u,\lambda)$$

(in the *mixed/graded* Grothendieck group $K^m\,\mathrm{Mod}_{coh}(\mathcal{D}_\lambda, N)$)

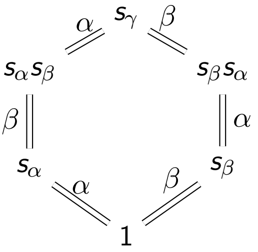
Apply $U_\alpha := p_\alpha^* p_{\alpha *}$:

$$\mathcal{L}(vs_\alpha,\lambda) \oplus (\cdots) = U_\alpha\mathcal{L}(v,\lambda) = \sum_u c_{vu} U_\alpha \mathcal{I}(u,\lambda)$$

Solve for $\mathcal{L}(vs_\alpha,\lambda)$.

Remark: the U_α's define an action $\mathcal{H}(W) \circlearrowright K^m\,\mathrm{Mod}_{coh}(\mathcal{D}_\lambda, N)$.

Example: $\mathfrak{g} = \mathfrak{sl}(3, \mathbb{C})$



dim = 3

dim = 2

dim = 1

dim = 0

Table of Contents

What's wrong with λ non-integral?

What goes wrong when λ is non-integral?

(λ non-integral $\iff \nexists$ finite dim rep of $\mathfrak{g}$ with inf char χ_λ)

What's wrong with λ non-integral?

What goes wrong when λ is non-integral?

(λ non-integral $\iff$ $\nexists$ finite dim rep of $\mathfrak{g}$ with inf char χ_λ)

$$
\begin{array}{ccccc}
 & & C(v) \cup C(vs_\alpha) & & \\
 & & \| & & \\
C(v) & \hookrightarrow & p_\alpha^{-1}(p_\alpha(C(v))) & \hookrightarrow & \mathcal{B} \\
 & \searrow & \downarrow & & \downarrow{\scriptstyle p_\alpha} \\
 & & p_\alpha(C(v)) & \hookrightarrow & \mathcal{P}_\alpha
\end{array}
$$

What's wrong with λ non-integral?

What goes wrong when λ is non-integral?
(λ non-integral $\iff$ $\nexists$ finite dim rep of $\mathfrak{g}$ with inf char χ_λ)

To define $p_{\alpha *}$ and p_α^* we need a sheaf $\mathcal{D}_{\lambda,\mathcal{P}_\alpha}$ on $\mathcal{P}_\alpha$.

What's wrong with λ non-integral?

What goes wrong when λ is non-integral?
(λ non-integral $\iff \nexists$ finite dim rep of $\mathfrak{g}$ with inf char χ_λ)

To define $p_{\alpha *}$ and p_α^* we need a sheaf $\mathcal{D}_{\lambda,\mathcal{P}_\alpha}$ on $\mathcal{P}_\alpha$.
If α is non-integral to λ, $\mathcal{D}_{\lambda,\mathcal{P}_\alpha}$ does not exist.

Strategy (λ non-integral)

Solution: if α is non-integral to λ,
replace U_α by the **intertwining functor**

Strategy (λ non-integral)

Solution: if α is non-integral to λ,
replace U_α by the **intertwining functor**

$$\begin{array}{ccccc} \mathcal{B} \times \mathcal{B} & \longleftarrow & \mathcal{B} \times_{\mathcal{P}_\alpha} \mathcal{B} & \longrightarrow & \mathcal{B} \\ & & \downarrow & & \downarrow \\ & & \mathcal{B} & \longrightarrow & \mathcal{P}_\alpha \end{array}$$

Strategy (λ non-integral)

Solution: if α is non-integral to λ,
replace U_α by the **intertwining functor**

$$\begin{array}{ccccc} \mathcal{B} \times \mathcal{B} & \longleftarrow & \mathcal{B} \times_{\mathcal{P}_\alpha} \mathcal{B} & \longrightarrow & \mathcal{B} \\ & & \downarrow & & \downarrow \\ & & \mathcal{B} & \longrightarrow & \mathcal{P}_\alpha \end{array}$$

$Z_\alpha := \mathcal{B} \times_{\mathcal{P}_\alpha} \mathcal{B} - \Delta\mathcal{B}$, a single G-orbit in $\mathcal{B} \times \mathcal{B}$.

Strategy (λ non-integral)

Solution: if α is non-integral to λ,
replace U_α by the **intertwining functor**

$$\begin{array}{ccc} Z_\alpha & \xrightarrow{p_1} & \mathcal{B} \\ {\scriptstyle p_2}\downarrow & & \\ \mathcal{B} & & \end{array}$$

Strategy (λ non-integral)

Solution: if α is non-integral to λ,
replace U_α by the **intertwining functor**

$$\begin{array}{ccc} Z_\alpha & \xrightarrow{p_1} & \mathcal{B} \\ {\scriptstyle p_2}\downarrow & & \\ \mathcal{B} & & \end{array}$$

$$I_\alpha := p_{2*}\left(\mathcal{L}_\alpha \underset{\mathcal{O}_{Z_\alpha}}{\otimes} p_1^*(-)\right)$$

Strategy (λ non-integral)

Solution: if α is non-integral to λ,
replace U_α by the **intertwining functor**

$$\begin{array}{ccc} Z_\alpha & \xrightarrow{p_1} & \mathcal{B} \\ {\scriptstyle p_2}\downarrow & & \\ \mathcal{B} & & \end{array}$$

$$I_\alpha := p_{2*}\left(\mathcal{L}_\alpha \underset{\mathcal{O}_{Z_\alpha}}{\otimes} p_1^*(-)\right)$$

α transversal to $C(v) \implies \dim p_2(p_1^{-1}(C(v))) = \dim C(v) + 1$
(I_α does the same job as U_α in the algorithm).

Strategy (λ non-integral)

Theorem (Beilinson-Bernstein)

If α is non-integral to λ, then I_α is an equivalence of categories

$$I_\alpha : \mathrm{Mod}_{coh}(\mathcal{D}_\lambda) \cong \mathrm{Mod}_{coh}(\mathcal{D}_{s_\alpha \lambda})$$

whose inverse is I_α. Moreover,

$$I_\alpha \mathcal{L}(v, \lambda) = \mathcal{L}(vs_\alpha, s_\alpha \lambda),$$

$$I_\alpha \mathcal{I}(v, \lambda) = \mathcal{I}(vs_\alpha, s_\alpha \lambda).$$

Strategy (λ non-integral)

Theorem (Beilinson-Bernstein)

If α is non-integral to λ, then I_α is an equivalence of categories

$$I_\alpha : \mathrm{Mod}_{coh}(\mathcal{D}_\lambda) \cong \mathrm{Mod}_{coh}(\mathcal{D}_{s_\alpha \lambda})$$

whose inverse is I_α. Moreover,

$$I_\alpha \mathcal{L}(v, \lambda) = \mathcal{L}(vs_\alpha, s_\alpha \lambda),$$

$$I_\alpha \mathcal{I}(v, \lambda) = \mathcal{I}(vs_\alpha, s_\alpha \lambda).$$

Price: need to work with different λ's.

Algorithm (λ non-integral)

Suppose we know $\mathcal{L}(v, \mu)$ for any μ. Let α be transversal to $C(v)$.
Want to find $\mathcal{L}(vs_\alpha, \lambda)$

Algorithm (λ non-integral)

Suppose we know $\mathcal{L}(v, \mu)$ for any μ. Let α be transversal to $C(v)$.
Want to find $\mathcal{L}(vs_\alpha, \lambda)$
If α is integral to λ,

$$U_\alpha \mathcal{L}(v, \lambda) \rightsquigarrow \mathcal{L}(vs_\alpha, \lambda).$$

Algorithm (λ non-integral)

Suppose we know $\mathcal{L}(v, \mu)$ for any μ. Let α be transversal to $C(v)$.
Want to find $\mathcal{L}(vs_\alpha, \lambda)$
If α is integral to λ,

$$U_\alpha \mathcal{L}(v, \lambda) \rightsquigarrow \mathcal{L}(vs_\alpha, \lambda).$$

If α is non-integral to λ,

$$I_\alpha \mathcal{L}(v, s_\alpha \lambda) = \mathcal{L}(vs_\alpha, \lambda).$$

This gives an algorithm for finding all irreducibles for all λ.
Remark: ... and an action $\mathcal{H}(W_\lambda) \circlearrowright \mathrm{Mod}_{coh}(\mathcal{D}_\lambda, N)$.

Example: $\mathfrak{g} = \mathfrak{sl}(3, \mathbb{C})$, $\lambda = \frac{1}{2}$(highest root)

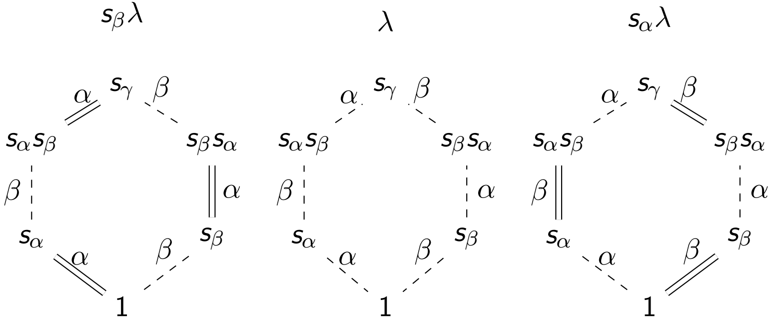


Table of Contents

Existing methods of category $\mathcal{O}'$

Beilinson-Bernstein-Lusztig (1984):

$$\mathrm{Mod}(\mathfrak{g}, N)_\lambda \overset{deform}{\rightsquigarrow} \mathrm{Mod}(\mathfrak{g}, N)_{rat} \rightsquigarrow \mathrm{Mod}(\mathcal{D}_{L*}, N)$$

$$\rightsquigarrow \mathrm{Perv}_N(L^*) \rightsquigarrow \text{positive char},$$

where $L \to \mathcal{B}$ is the total space of a line bundle determined by the rational twist, and $L^* = L-$ zero sections.

Existing methods of category $\mathcal{O}'$

Beilinson-Bernstein-Lusztig (1984):

$$\mathrm{Mod}(\mathfrak{g}, N)_\lambda \overset{deform}{\rightsquigarrow} \mathrm{Mod}(\mathfrak{g}, N)_{rat} \rightsquigarrow \mathrm{Mod}(\mathcal{D}_{L*}, N)$$

$$\rightsquigarrow \mathrm{Perv}_N(L^*) \rightsquigarrow \text{positive char},$$

where $L \to \mathcal{B}$ is the total space of a line bundle determined by the rational twist, and $L^* = L-$ zero sections.

Soergel (1990):

$$\mathrm{Mod}(\mathfrak{g}, N) \rightsquigarrow \mathrm{Mod}(\mathcal{D}_\mathcal{B}, N) \rightsquigarrow \mathrm{Perv}_\mathrm{N}(\mathcal{B}) \rightsquigarrow \text{Soergel bimodules}$$

Existing methods of category $\mathcal{O}'$

Beilinson-Bernstein-Lusztig (1984):

$$\mathrm{Mod}(\mathfrak{g}, N)_\lambda \overset{deform}{\rightsquigarrow} \mathrm{Mod}(\mathfrak{g}, N)_{rat} \rightsquigarrow \mathrm{Mod}(\mathcal{D}_{L*}, N)$$

$$\rightsquigarrow \mathrm{Perv}_N(L^*) \rightsquigarrow \text{positive char},$$

where $L \to \mathcal{B}$ is the total space of a line bundle determined by the rational twist, and $L^* = L-$ zero sections.

Soergel (1990):

$$\mathrm{Mod}(\mathfrak{g}, N) \rightsquigarrow \mathrm{Mod}(\mathcal{D}_\mathcal{B}, N) \rightsquigarrow \mathrm{Perv}_\mathrm{N}(\mathcal{B}) \rightsquigarrow \text{Soergel bimodules}$$

Both methods require going to perverse sheaves. (with Mochizuki's Decomposition Theorem for holonomic D-modules, BBL's method doesn't need perverse sheaf anymore...)

Advantage of our method

Compared with Beilinson-Bernstein-Lusztig's approach, we don't need to distinguish rational twists from arbitrary twists.

Advantage of our method

Compared with Beilinson-Bernstein-Lusztig's approach, we don't need to distinguish rational twists from arbitrary twists.

Compared with Soergel's approach, we don't need to go into perverse sheaves.

Advantage of our method

Compared with Beilinson-Bernstein-Lusztig's approach, we don't need to distinguish rational twists from arbitrary twists.

Compared with Soergel's approach, we don't need to go into perverse sheaves.

Example: Whittaker modules $\mathrm{Mod}_{coh}(\mathcal{D}_\lambda, N, f)$ do not correspond to perverse sheaves (because these D-modules are NOT regular holonomic).

Thank you!