F-polynomials and Newton polytopes

Gleb Koshevoy, Denis Mironov

Institute for Information Transmission Problems Russian Academy of Sciences
Moscow Center for Continuous Math Education

Polynomial Computer Algebra 2023
April 19, 2023

Notation

G simple, simply connected (simply-laced) algebraic group over $\mathbb{C}$

$$U \subset B \subset G, \text{unipotent radical in maximal Borel in } G$$

Standard example: $\begin{pmatrix} 1 & \star & \star & \star \\ 0 & \ddots & \star & \star \\ \vdots & 0 & \ddots & \star \\ 0 & \dots & 0 & 1 \end{pmatrix} \subset \begin{pmatrix} \star & \star & \star & \star \\ 0 & \ddots & \star & \star \\ \vdots & 0 & \ddots & \star \\ 0 & \dots & 0 & \star \end{pmatrix} \subset \mathrm{SL}_{n+1}(\mathbb{C})$

The ring $\mathbb{C}[U]$ of regular functions on U has the nice property that every irreducible representation of G embeds into it:

$$V(\lambda) \subset \mathbb{C}[U].$$

String parametrizations

- A 'nice' basis for $\mathbb{C}[U]$ provides 'nice' bases for all irreducible representations $V(\lambda) \subset \mathbb{C}[U]$ simultaneously.
- The dual canonical basis is such a 'nice' basis . Its combinatorics is governed by the string cones.
- String cones are rational polyhedral cones $\mathcal{S}_{\mathbf{i}} \subset \mathbb{R}^N$ ($N = \#$ positive roots of G) whose integer points parametrize the dual canonical bases. Each reduced expression $\mathbf{i}$ of the longest element w_0 of the Weyl group W of G yields such a cone.

An example of a string parametrizations

Let $G = \mathrm{SL}_3$, then $W \cong S_3$ and $s_1 s_2 s_1 = w_0$ is a reduced expressions $\mathbf{i} = (1,2,1)$ of w_0. In this example we have

$$\mathcal{S}_{\mathbf{i}} = \{(x_1, x_2, x_3) \in \mathbb{R}^3 \mid x_3 \geq 0,\ x_1 \geq 0,\ x_2 - x_3 \geq 0\}.$$

If we consider two reduced words $\mathbf{i_1}$ and $\mathbf{i_2}$, then there is a piecewise linear bijection

$$\Psi_{\mathbf{i_2}}^{\mathbf{i_1}} : \mathbb{R}^N \to \mathbb{R}^N$$

such that

$$\Psi_{\mathbf{i_2}}^{\mathbf{i_1}}(\mathcal{S}_{\mathbf{i}_1}) = \mathcal{S}_{\mathbf{i}_2}.$$

In our example:

$$\Psi_{2,1,2}^{1,2,1} : S_{(1,2,1)} \to S_{(2,1,2)}$$
$$(x_1, x_2, x_3) \mapsto (\max(x_3, x_2 - x_1), x_1 + x_3, \min(x_1, x_2 - x_3)).$$

Computation of inequalities

The piecewise linear bijection $\Psi^{\mathbf{i}_1}_{\mathbf{i}_2} : \mathcal{S}_{\mathbf{i}_1} \to \mathcal{S}_{\mathbf{i}_2}$ can be used to compute the inequalities of all string cones.

- In general many recursive steps are needed to compute the inequalities.
- The number of inequalities might be exponential with respect to the rank of G. (For example, for E_8 the number might be > 6899079264).

Changing perspective to the geometric crystals

Several varieties related to a reductive group G have geometric crystal structures. As shown by Berenstein and Kazhdan [1], the varieties $B^-_{w_0} = B^- \cap U\overline{w_0}U$ ($\overline{w_0}$ is is a representative of the longest element w_0 in the Weyl group W in $Norm_G(T)$) and $T \cdot B^-_{w_0}$ have geometric crystal structures.

The Berenstein-Kazhdan decoration function Φ_{BK} on $T \cdot B^-_{w_0}$ is defined as

$$\Phi_{BK} = \sum_{i \in I} \frac{\Delta_{w_0\Lambda_i, s_i\Lambda_i}}{\Delta_{w_0\Lambda_i,\Lambda_i}} + \sum_{i \in I} \frac{\Delta_{w_0 s_i\Lambda_i,\Lambda_i}}{\Delta_{w_0\Lambda_i,\Lambda_i}}$$

where Λ_i denotes the ith fundamental weight, I the set of simple roots, and $\Delta_{u\Lambda_i, v\Lambda_i}$ is a generalized minor due to Berenstein-Zelevinsky.

The variety $T \cdot B^{-}_{w_0}$ has a positive structure $\theta_{\mathbf{i}} : T \cdot (\mathbb{C}^*)^{l(w_0)} \to T \cdot B^{-}_{w_0}$ associated is with any reduced decomposition $\mathbf{i}$ of w_0. (Birational mapping with positive coefficients.)
Considering the tropicalization of the rational function Φ_{BK} with respect to such a positive structure, one obtains due to Berenstein and Kazhdan [1] a Kashiwara subcrystal

$$\{z \in X^*(T \cdot (\mathbb{C}^*)^{l(w_0)}) \,|\, Trop\Phi_{BK}(\theta_{\mathbf{i}}(z) \geq 0\},$$

X^* denotes the set of cocharacters, *Trop* the troopicalization functor, which is isomorphic to the disjoint union of all crystal bases $B(\lambda)$ of the finite dimensional irreducible representations of the quantum group $\mathcal{U}_q({}^L\mathfrak{g})$, with highest weights λ. Here, ${}^L\mathfrak{g}$ is the Langlands dual Lie algebra of $\mathfrak{g} = Lie(G)$.

Example

Let $T = (\mathbb{C}^*)^2$,$f(x_1, x_2) = \frac{x_1+x_2}{x_2}$. Then the tropicalization is the piecewise-linear map

$$[f]_{trop} : \mathbb{Z}^2 \to \mathbb{Z}, (x_1, x_2) \mapsto \min(x_1, x_2) - x_2.$$

Kankubo and Nakashima defined a half potential $\Phi^h_{BK} = \sum_I \Delta_{w_0\Lambda_i, s_i\Lambda_i}$ and a positive structure $\theta^-_{\mathbf{i}} : (\mathbb{C}^*)^{l(w_0)} \to B^-_{w_0}$, such that a subcrystal

$$\{z \in X^*((\mathbb{C}^*)^{l(w_0)}) \mid Trop\Phi^h_{BK}(\theta^-_{\mathbf{i}}(z) \geq 0\},$$

is isomorphic to the crystal base $B(\infty)$ of the negative part $\mathcal{U}^-_q({}^L\mathfrak{g})$ of $\mathcal{U}_q({}^L\mathfrak{g})$.

One can compute the generalized minors and hence the Nakashima-Zelevinsky polyhedral realization of string cones using **i**-trails due to Berenstein-Zelevinsky. However no combinatorial (and algorithmic) description of **i**-trails was known except the type A (Glietzer and Postnikov (2000)) and for special reduced decompositions (Littelmann (1996)).

Geometric crystals and the Berenstein-Kazhdan potential

For classical groups, Kanakubo-Koshevoy-Nakashima established [7] an algorithm for explicit computing the half of the Berenstein-Kazhdan potential Φ^h_{BK} for each reduced decomposition $\mathbf{i}$ of the longest element w_0.

The tropicalization of Φ^h_{BK} defines the string cone parametrization

$$\mathcal{C}_{\Sigma_{\mathbf{i}}}$$

Changing perspective to the cluster setup

The cluster spaces $\mathcal{A}$ and $\mathcal{X}$ are unions of open tori $\mathcal{A} = \cup_{\Sigma}\mathbb{T}_{\Sigma}$, $\mathcal{X} = \cup_{\Sigma}\mathbb{T}_{\Sigma}^{\vee}$, which are glued via certain birational transformations, called $\mathcal{A}$- and $\mathcal{X}$-cluster mutations, respectively. The elements Σ in the common index set of the two dual toric systems are called seeds. The families of charts, equip $\mathcal{A}$ and $\mathcal{X}$ with the structure of a positive variety admitting tropicalization.

U is a 'partial compactification of a cluster variety' $B^-_{w_0}$ $\Rightarrow$ we can apply the machinery of Gross-Hacking-Keel-Kontsevich to U (up to some technical conditions) giving

- a basis for $\mathbb{C}[U]$
- many parametrizations of this basis by rational polyhedral cones $\mathcal{C}_\Sigma$ (Σ a possibly infinite index set)

Theorem (Genz-Koshevoy-Schumann)

The string cones appear as a subset of the parametrizations $\mathcal{C}_\Sigma$, i.e. for any reduced expression $\mathbf{i}$ there exists a cluster seed $\Sigma_\mathbf{i}$ and a unimodular bijection

$$\mathcal{C}_{\Sigma_\mathbf{i}} \to \mathcal{S}_\mathbf{i}$$

(and the technical conditions are satisfied here.)

Parametrizations

In this setup Gross-Hacking-Keel-Kontsevich constructed a theta basis B_{can} of $\mathbb{C}[U]$ and a regular function $W\colon \mathcal{X} \to \mathbb{C}$ (called potential) such that B_{can} is parametrized by

$$\{x \in \mathbb{R}^N \mid [W|_{T_\Sigma^\vee}]_{trop}(x) \geq 0 \text{ for a } \Sigma \text{ } (\iff \text{ for any } \Sigma)\}.$$

Here $W|_{T_\Sigma^\vee} \in \mathbb{C}[T_\Sigma^\vee]$, hence $W|_{T_\Sigma^\vee} \in \mathbb{C}[x_k^{\pm 1} \mid 1 \leq k \leq N]$ is a Laurent polynomial.
$[W|_{T_\Sigma^\vee}]_{trop} : \mathbb{R}^N \to \mathbb{R}$ is the piecewise linear map we get when we replace multiplication by addition and addition by taking the minimum.

Computation W and Newton polytopes

Original definition of the potential W due to Gross-Hacking-Keel-Kontsevich uses division of Laurent polynomials. To get explicit formulas using the original definition is not an easy task for computers.

We get an algorithm for an explicit form of W using only summation.

Computation W and Newton polytopes

The frozen vertices and variables are labelled by the set $-I \cup I$. For a frozen vertex, a seed Σ is optimal if such a vertex is a source vertex after deleting the edges joining this vertex and other frozen. For a frozen $a \in I$, there exists an appropriate reduced word $\mathbf{i}'$, such that seed $\Sigma_{\mathbf{i}'}$ is optimal for a. For a frozen $-a \in -I$, an optimal seed is obtained along 'level line mutations' of $\Sigma_{\mathbf{i}'}$.

For the optimal seed Σ for a frozen $\pm a$, the $\pm a$th part of the GHKK-potential is equal to the value of the corresponding frozen cluster variable,

$$W_{\pm a} = Y_{\pm a}. \quad (1)$$

For a given reduced decomposition $\mathbf{i}$, one can compute the half

$$W^h_{GHKK} = \sum_{a \in I} W_a$$

of the GHKK-potential

$$W_{GHKK} = \sum_{\pm a \in -I \cup I} W_{\pm a}$$

using cluster mutations corresponding to 3-braid moves between the reduced decompositions of w_0 (for l and k, such that $a_{lk} = -1$, $s_k s_l s_k = s_l s_k s_l$). Namely, for computing W_a, we apply a sequence of cluster mutations corresponding to 3-braid moves which transform $\Sigma_{\mathbf{i}}$ into an optimal seeds for a, then W_a is the X-cluster variable at the frozen vertex labeled by a in the optimal seed computed in the variables of the seed $\Sigma_{\mathbf{i}}$. In variables of the seed $\Sigma_{\mathbf{i}}$, such an X-cluster variable is equal to the specification of the F-polynomial (see [4, 8]).
A half of W_{GHKK} is a polynomial in the X-cluster variables $\Sigma_{\mathbf{i}}$([9]).

Computation W and Newton polytopes

Thus W_a and takes the form

$$W_a = Y_1^{c_{1a}(t)} \cdots Y_N^{c_{Na}(t)} \prod_i F_i(t)(Y_1, \cdots, Y_N)^{b_{ia}(t)}.$$

In the above formula we take notations of [8], where t means the end vertex of the path in the mutation graph from the optimal seed for a to $\Sigma_{\mathbf{i}}$ and Y_j's are cluster variables of $\Sigma_{\mathbf{i}}$.
From [5] we can compute full GHKK-potential:

$$W_{GHKK} = W_{GHKK}^h + \sum_{i \in I} Y_{i_s}^{-1}(1 + Y_{i_{s-1}}^{-1}(1 + Y_{i_{s-2}}^{-1}(1 + Y_{i_{s-3}}^{-1}(\cdots))))),$$

where $i_1, i_2, \ldots i_s$ are indices of i in reduced decomposition $\mathbf{i}$.

Computation W and Newton polytopes

Based on such an algorithm we study the Newton polytope of W and the half W^h of W and make conjecture that the latter polytope is void, that is it does not contains interior integer lattice points, and the former polytope contains a unique lattice interior point. This conjecture supports by the mirror symmetry construction of W as a Landau-Ginzburg potential to an affine Calabi-Yau manifold (compatifications of big Bruhat cells). However, we do not have a rigorous mirror symmetry proof of this conjecture.

Lattice properties of W and Newton polytopes

Theorem 1

For simply-laced G, and a given reduced decomposition $\mathbf{i}$*, the Newton polytopes* $\Phi_{BK}|_{\Sigma_\mathbf{i}}$ *and* $W|_{\Sigma_\mathbf{i}}$ *are isomorphic under a unimodular transformation.*

Corollary 2

The Newton polytopes Φ^h_{BK} *is void if and only if the Newton polytopes* W^h *is void.*

We state the following

Conjecture 1. For a simply-laced group G, and any reduced decomposition $\mathbf{i}$ of w_0, the Newton polytope of W^h_{GHKK} is void.

Lattice properties of W and Newton polytopes

Conjecture 2. For a simply-laced group G, and any reduced decomposition $\mathbf{i}$ of w_0, the Newton polytope of W_{GHKK} contains a unique interior lattice point.

For type A, the conjecture 1 holds true (Theorem).
For the numeric verification of Conjectures we compute the Newton polytope Φ^h_{BK} using the Kanakubo-Koshevoy-Nakashima algorithm and Polymake. We made computer verification of Conjecture 1 for the following cases D_n, $n = 4, 5, 6, 7$, E_6, E_7, and of Conjecture 2 for the following cases A_n, $n = 3, 4, 5, 6$, D_4, D_5.

Algorithm description

We consider the Berenstein-Zelevinsky positive structure $\theta_{-\mathbf{i}} : \mathbf{T}' \to B^-_{w_0} = B^- \cap Bw_0B$, on the geometric crystal $B^-_{w_0} = B^- \cap Bw_0B$. A tuple $(t_1, \ldots, t_N)$ denotes an element of $\mathbf{T}'$. The algorithm for computation the half of the Berenstein-Kazhdan decoration function Φ^h_{BK} is based on Theorem 4.4 in [7]. For the input data consisting of a group G and reduced word $w_0 = \mathbf{i}$.

We compute Φ_{BK} as sum:

$$\Phi^h_{BK} = \sum_{j \in I} \Delta_{w_0\Lambda_j, s_j\Lambda_j}(\theta_{-\mathbf{i}}(t_1, \ldots, t_N)),$$

where $\Delta_{w_0\Lambda_j, s_j\Lambda_j}$ is generalized minor function (see Defenition 2.2 in [7]).

By Theorem 4.4. [7] it is possible to compute all monomials in $\Delta_{w_0\Lambda_i,s_i\Lambda_i}$ by consequently applying multiplication by monomials

$$A_k^{-1} = \frac{\prod_{k<l<k^+} t_l^{-a_{i_l i_k}}}{t_k t_{k^+}},$$

(see 3.7 in [7]) starting from predefined source monomial (A= $(a_{i_l i_k})$ is the Cartan matrix). The algorithm performs graph enumeration for graph with vertices being monomials of $\Delta_{w_0\Lambda_j,s_j\Lambda_j}$ and edges are relations of monomials being different by multiplication by A_k^{-1}.

Algorithm description: additional notation

Operations $k+$ and $k-$ are lookups in reduced decomposition w:

Defenition

$$k+ := \min\{l \in [1, \mathrm{len}(w)] | w_l = w_k, l > k\} \cup \{\infty\}$$
$$k- := \max\{l \in [1, \mathrm{len}(w)] | w_l = w_k, l < k\} \cup \{0\}$$

Example:

$11- = 3$

$$w = (5,\ 3,\ 4,\ 2,\ 3,\ 5,\ 1,\ 2,\ 3,\ 2,\ 4,\ 3,\ 1,\ 2,\ 1,\ 3,\ 5,\ 3,\ 2,\ 1)$$

$2+ = 5$

Algorithm description: additional notation

In the algorithm, we will associate integer vector $b = (b_1, b_2, \cdots, b_N)$ to each monomial.

Definition

Let $M = \prod_{l=1}^{N} t_l^{d_l}$ be a Laurent monomial. We inductively define integers $\{b_l\}_{l=N,N-1,\cdots,1}$ as

$$b_N = d_N + s_i \Lambda_i(h_{i_N}),$$

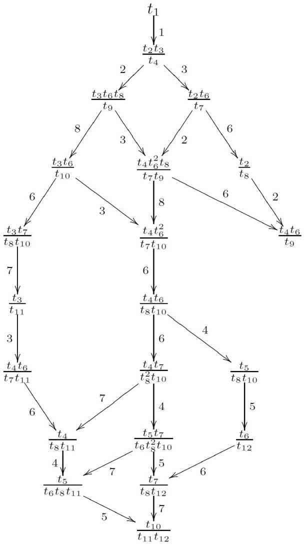
$$b_t = d_t + s_i \Lambda_i(h_{i_t}) - \sum_{l=t}^{N-1} b_{l+1} a_{i_t, i_{l+1}} \quad (t = N, N-1, \cdots, 1).$$

We need to compute b-vector only for source monomial.
By lemma 5.2 [7] each A_k^{-1} multiplication changes b-vector only at two indices k and $k+$.

Mutation procedure

```
def mutate_node(M)
        //M = ∏_{m=1}^N t_m^{d_m}
        compute b_j for node if it was not computed previously
        set initial result as empty list
        for all simple roots α_l
            for all k in set of indices where w_k == l
                if k+ < ∞
                    if d_k<2 and b_{k+}>0
                        new_monomial = M * A_k^{-1}
                        compute new_monomial b_j:
                            Lemma 5.2 [7]
                            new_monomial b_j = old_monomial b_j
                            new_monomial b_k +=1
                            new_monomial b_{k+} -=1
                        cache new_monomial b_j
                        add new_monomial and graph edge to result
                    if d_k == d_{k+}
                        set lookup depth h=2
                        while k^{+h} < ∞ and d_{k+h} = 0 and b_{k^{h+}} = 0
                            h++
                        if d_{k+h} = −1 and b_{k+h} = 1
                            new_monomial = M * A_k^{-1}
                            compute new_monomial b_i:
                                [Lemma 3.4]
                                new_monomial b_j = old_monomial b_j
                                new_monomial b_k +=1
                                new_monomial b_{k+} -=1
                            cache new_monomial b_j
                            add new_monomial and graph edge to result
        return result
```

Φ_{BK} example



$G = D_4,\ w = (2,1,3,2,4,2,3,2,1,2,3,4)$ and $i = 2$

Algorithm description

To compute potential W we use [6, 4, 8] as follows. Corresponding X-cluster optimal seed Σ_w produces decomposition of half of W_{GHKK} using cluster mutations corresponding to 3-braid moves. In variables of seed Σ_w X-cluster variables are equal to X to F-polynomials.

$$W^h_{GHKK} = \sum_{a \in I} Y_1^{c_{1a}(t)} \cdots Y_N^{c_{Na}(t)} \prod_i F_i(t)(Y_1, \cdots, Y_N)^{b_{ia}(t)}.$$

To compute F-polynomial by [2, 3] we can determine that $t_m = \frac{X_m}{X_{m^-}}$ and A_j becomes equal to Y_j in X-cluster variables. Using this property we can produce algorithm to compute W_{GHKK} from data obtained in Φ_{BK} computation:

$$\sum_i \Delta_{w_0 \Lambda_i, s_i \Lambda_i} \circ \theta^-_{\mathbf{i}}(t_1, \cdots, t_N) \|_{t_m \to Y_m}$$

W_{GHKK} computation procedure

```
compute GHKK support
        b_start = get b_i vector of source monomial in Δ_{w_0Λ_i,s_iΛ_i}
        b_stop = get b_i vector of stop monomial in Δ_{w_0Λ_i,s_iΛ_i}
        ee = basis spanned by e_k − e_{k+}
        GHKK_support = coordinates of b_stop-b_start in ee


compute Y_frozen
        Y_frozen=Y[number last occurrence of j in i]


compute W_j
        compute start monomial
                Y_start = Y_frozen * ∏_{0<k<=len(i)} Y_k
        W = dictionary with keys in monomials in Δ_{w_0Λ_i,s_iΛ_i}
        W[start monomial of Δ_{w_0Λ_i,s_iΛ_i}] = Y_start
        enumerate edges in graph Gs starting from source monomial:
                v_b = start of the edge
                v_e = end of the edge
                k = mark on the edge
                W[v_e]=W[e_b]*Y[k]

        return set(values(W))
```

To compute full W_{GHKK} we use straightforward formula

$$W_{GHKK} = W^h_{GHKK} + \sum_{i \in I} Y_{i_s}^{-1}(1 + Y_{i_{s-1}}^{-1}(1 + Y_{i_{s-2}}^{-1}(1 + Y_{i_{s-3}}^{-1}(\cdots))))),$$

Algorithm complexity

All computational operations in the algorithm can be reduced only to vector summation (on each step we only multiply Laurent monomials).
If r is rank of G, then complexity can be expressed in the terms r and number of monomials in Φ_{BK}. Precisely, graph enumeration in is making no more than $len(w_0) \sim r^2$ tries for each monomial in Φ_{BK}. Each try does no more than $O(r^2)$ additions and no more than $O(r)$ lookup operations in reduced decomposition and b-vectors.
Using prefix tree search for string representations of monomials it is possible to reduce search of already computed monomials to linear time in length of string representation of monomials of $\Phi_{BK} \sim O(r^2)$. Also W_{GHKK} is linear in Φ_{BK} string length.
Therefore total complexity is

$$O(r^4 K) \sim O(r^2 * \textbf{length of string representation}),$$

which is proportional to time needed to print the answer.

Real computational speed and experiments

We computed Φ_{BK} and W_{GHKK} for several millions cases for A_n, $n = 3, 4, 5, 6, 7$, D_n, $n = 4, 5, 6, 7$, E_6, E_7. Average time of $\Delta_{w_0\Lambda_j, s_j\Lambda_j}$ and W_j calcutaion with SageMath for single simple root of D_6 is 70ms on PC with dual 3.8 Ghz Intel®Xeon® Gold 5222 CPU running Ubuntu Linux.

To make verification of lattics properties conjectures we used Polymake `N_INTERIOR_LATTICE_POINTS` method. Unfortunately it is exponentially bound in terms of number of inequalities.

Example

For single reduced word for D_6 average time of verification of Conjecture 1 is 12 seconds, for E_7 - almost 90 seconds. Some cases can take much more time depending on actual Newton polytope of Φ_{BK} and W_{GHKK}. Checking Conjecture 2 for full W_{GHKK} polytope is much slower and can take from several hours to days even for D_5.

Real computational speed and experiments

Checking conjecture and other properties of Φ_{BK} and W_{GHKK} relies on enumeration of all reduced decompositions of longest element in Weyl group w_0. If two reduced words can be transformed one into another using only 2-braid moves, then the results of computation differ only by variable exchange.

Number of classes of reduced decompositions is much smaller than number of all reduced decompositions. For example, for D_4 it's 182 and 2316, and for D_5 - 13198 and 12985968 (see A180607 OEIS sequence).

Auxiliary problem

Find algorithm of minimal complexity and constant memory consumption enumerating set classes of reduced decompositions of longest element of Weyl group and produce one reduced decomposition from each class with respect of equivalence by 2-braid moves.

[1] A.Berenstein, D.Kazhdan, Geometric and unipotent crystals. II. From unipotent bicrystals to crystal bases, Quantum groups, 13–88, Contemp. Math., 433, Amer. Math. Soc., Providence, RI, (2007).

[2] Berenstein, A., Fomin,S., Zelevinsky,A., Cluster algebras.III. Upper bounds and double Bruhat cells. Duke Math. J. 126(1), 1–52 (2005)

[3] A. Berenstein, A. Zelevinsky, Tensor product multiplicities, canonical bases and totally positive varieties, Invent. Math. 143(1), 77–128 (2001) arXiv:math/9912012.

[4] Fomin, S., Zelevinsky, A. Cluster algebras.IV. Coefficients. Compos. Math.143,112–164(2007)

[5] V. Genz, G. Koshevoy, and B. Schumann. "Polyhedral parametrizations of canonical bases & cluster duality". Advances in Mathematics 369 (2020), p. 107178

[6] Gross, M., Hacking, P., Keel, S., Kontsevich, M., Canonical bases for cluster algebras. J. Am. Math. Soc. 31, 497–608 (2018)

[7] Y.Kanakubo, G.Koshevoy and T.Nakashima, An algorithm for Berenstein-Kazhdan decoration functions and trails for classical groups, arXiv:2207.08065

[8] Keller B., Cluster algebras and derived categories, arXiv:1202.4161

[9] G.Koshevoy and B.Schumann, Redundancy in string cone inequalities and multiplicities in potential functions on cluster varieties, J. of Algebraic Combinatorics (2022)

[10] https://github.com/mironovd/crystal