Laurent Phenomenon Algebras Talk

Oliver Daisey

Structure of the talk

1. Introduction to cluster algebras.
2. Introduction to LP algebras.
3. Linear LP algebras.
4. LP algebra SageMath package.

1 Cluster algebras

First start by introducing cluster algebras. Keep in mind this is a "simplified" account (there are many, many developments on the subject).

1.1 Background

Cluster algebras are commutative rings with distinguished generators (cluster variables) having a remarkable combinatorial structure. Useful in:

- Poisson geometry (we can give a Poisson variety a mutually compatible structure of a cluster algebra),
- triangulations of surfaces and Teichmüller theory (triangulations of surfaces form a cluster algebra),
- tropical geomety (ties into the positivity and mathematical physics aspects of cluster algebras), and in general:
- Mathematical physics: wall-crossing phenomena, quiver gauge theories, scattering amplitudes, soliton solutions to the KP equation.
- ...and many more!

They were originally defined by Andrei Zelevinsky and Sergey Fomin in 2000 to study things like total positivity and (dual) canonical bases of semisimple Lie algebras. In this talk we will approach them combinatorially.

1.2 Pentagon recurrence

To get the basic idea, consider the sequence $f_1, f_2, f_3, \ldots$ defined recursively by $f_1 = x$, $f_2 = y$ and

$$f_{n+1} = \frac{f_n + 1}{f_{n-1}}. \tag{1}$$

The first five entries are

$$x, y, \frac{y+1}{x}, \frac{x+y+1}{xy}, \frac{x+1}{y}. \tag{2}$$

Two surprising/important properties to note:

1. The sixth and seventh entries are x and y respectively, so this sequence is periodic with period five.
2. Each entry of the sequence is a Laurent polynomial (even with nonnegative integer coefficients) in the original x and y. We cannot expect this *a priori*.

You could view this recurrence as the evolution of a "moving window" consisting of two consecutive terms f_i and f_{i+1}:

$$\begin{bmatrix} f_1 \\ f_2 \end{bmatrix} \xrightarrow{\mu_1} \begin{bmatrix} f_3 \\ f_2 \end{bmatrix} \xrightarrow{\mu_2} \begin{bmatrix} f_3 \\ f_4 \end{bmatrix} \xrightarrow{\mu_1} \begin{bmatrix} f_5 \\ f_4 \end{bmatrix} \xrightarrow{\mu_2} \cdots$$

where

$$\mu_1 : \begin{bmatrix} f \\ g \end{bmatrix} \to \begin{bmatrix} \frac{g+1}{f} \\ g \end{bmatrix}, \qquad \mu_2 : \begin{bmatrix} f \\ g \end{bmatrix} \to \begin{bmatrix} f \\ \frac{f+1}{g} \end{bmatrix}.$$

Both μ_1, μ_2 are involutions, so $\mu_1^2 = \mu_2^2 = 1$. Let us refer to these maps as "mutations" of the *cluster* $\begin{bmatrix} f \\ g \end{bmatrix}$. The 5-periodicity of the recurrence translates into the identity $(\mu_2\mu_1)^5 = 1$. Thus the group generated by μ_1, μ_2 is a dihedral group with 10 elements.

Take the algebra generated by the terms of sequence (1) over $\mathbb{C}$ (different base rings are permissible; we'll get to that later). Since (1) is periodic, this algebra is finitely generated.

$$\mathcal{A} = \mathbb{C}\left[x, y, \frac{y+1}{x}, \frac{x+y+1}{xy}, \frac{x+1}{y}\right]$$

Thus it is the coordinate ring of an affine algebraic set. Relations exist among generators. Embeds in $\mathbb{A}^3$ as

$$\mathbb{C}[x,y,z]/(xyz - x - y - 1).$$

Our strategy with cluster algebras is to find a framework for taking a set of n independent variables like this, and then proceed to "mutate" them to obtain new generators for the cluster algebra. We should then be able to generalise a lot of recurrences similar to this one. Properties we want:

1. Laurent phenomenon should be "built in" to our framework: Every new generator of the cluster algebra should be a Laurent polynomial in the original variables.

2. Total positivity.

1.3 Quivers

A *quiver* is a finite directed graph. Multiple edges are allowed, but no loops or 2-cycles.

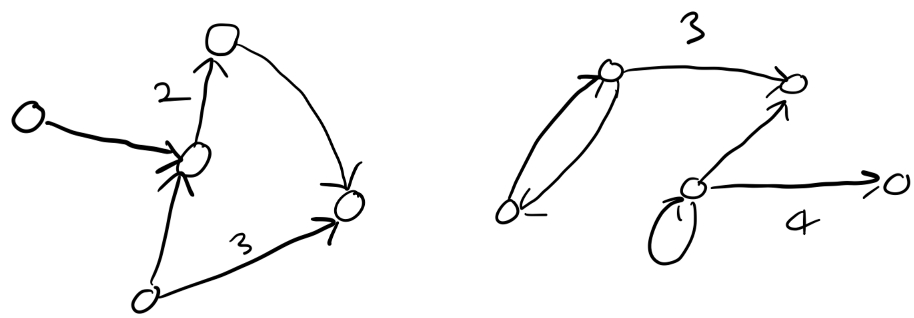


Figure 1: Left: Example of quiver. Right: Non-example.

We can *mutate* a quiver at a vertex k to get a new quiver on the same number of vertices using the following procedure:

1. Reverse all arrows incident to vertex k.

2. For every path $i \to k \to j$, add a "shortcut" $i \to j$.

3. Cancel out all 2-cycles.

Here is an example.

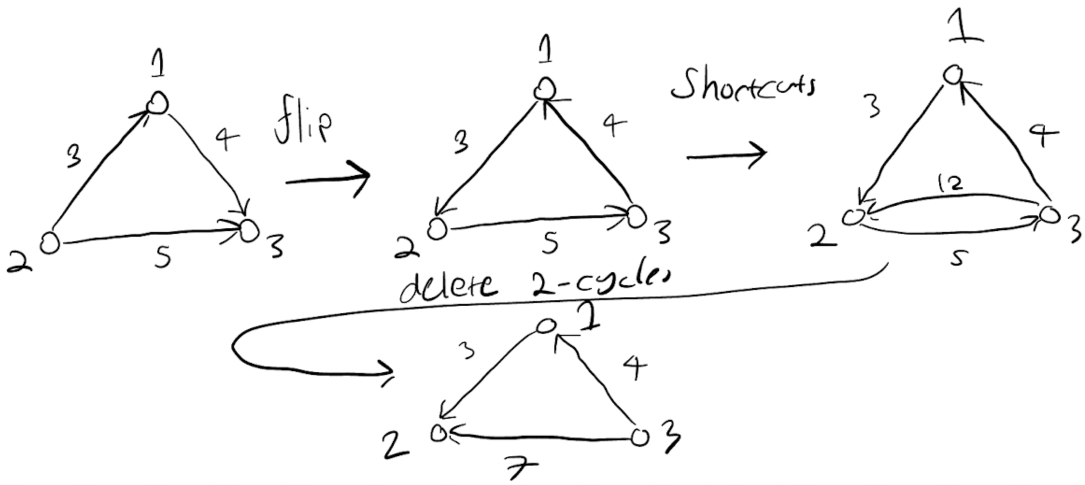


Figure 2: Example of mutating a quiver at vertex 1.

Mutation is an involution (not too obvious immediately, but should be evident from examples). Quivers are:

- **mutation equivalent** if there exists a sequence of mutations taking one quiver to the other. Check this is an equivalence relation. We can then talk about the *mutation class* of a quiver.
- **mutation finite** if the mutation class of the quiver is finite. Easy examples and nonexamples.

Thanks to result of Felikson, Tumarkin and Shapiro in 2008, we know which quivers are mutation finite (quivers on two vertices, quivers that come from triangulated surfaces, or one of 11 exceptional quivers).

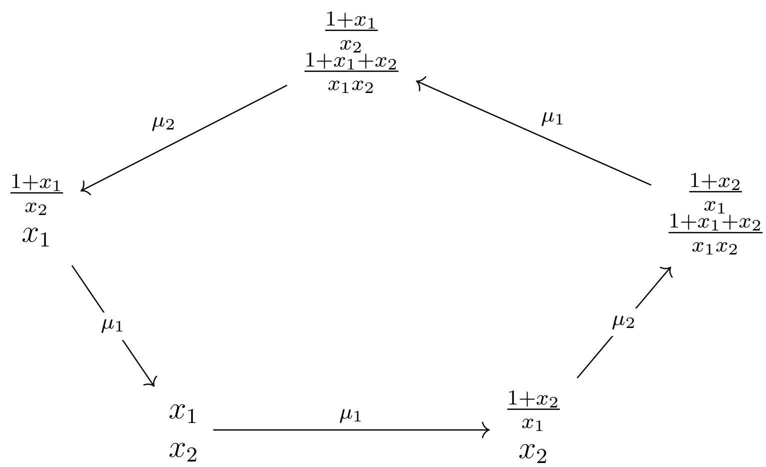
1.4 Cluster algebras

Attach algebraically independent variables $x_1, \ldots, x_n$ to the vertices of a quiver to get a *seed*. Each vertex corresponds to an *exchange relation*

$$x_k x_k' = \prod_{i \to k} x_i + \prod_{k \to j} x_j$$

The set of variables attached to the quiver is called a *cluster*. Then when we mutate at vertex k, we replace x_k with x_k' as defined by its exchange relation. Cluster algebra is defined to be the algebra generated by all cluster variables obtainable by mutations.

Quiver with two vertices and a single edge corresponds to the pentagon recurrence. We can form the *exchange graph* where the vertices are seeds and the edges are mutations.



Note:

1. Notions of mutation equivalence and finite type carry over. Equality of seeds is pretty much saying there is a permutation of the cluster variables of one quiver that induces a quiver isomorphism on the underlying quivers.

2. Seed is finite type implies underlying quiver is finite mutation type, but the converse is not true.

3. Classification of finite type cluster algebras is exactly parallel to Dynkin diagram classification of semisimple Lie algebras over algebraically closed field.

Note that finite type cluster algebras and quivers are therefore completely understood.

Indeed, this definition of cluster algebras (and quiver mutation) gives us the properties we want, although this is *not* easy to prove:

- Laurent phenomenon
- Positivity

2 LP algebras

We love the Laurent phenomenon so much, that we will try to define a new type of cluster-style algebra with clusters that satisfy this property, *and* allow a strictly greater range of exchange relations than just binomials. We will also hope to get positivity as a consequence. Maybe then we can also identify when algebras like this arise in the wild.

Motivation: Generalisation as a problem-solving strategy; LP algebras from marked surfaces; LP algebras from homogeneous varieties (joint work of me and Tom Ducat); etc.

Introduced by Thomas Lam and Pavlo Pylyavskyy in 2012.

2.1 Seeds

We start by letting A be a UFD over $\mathbb{Z}$. We view A as a *coefficient ring*. Denote by $\mathcal{F}$ the rational function field in n independent variables over the field of fractions $\mathrm{Frac}(A)$.

A *seed* S is a tuple $(\mathbf{x}, \mathbf{f})$, where $\mathbf{x} = \{x_1, \ldots, x_n\}$ is a transcendence basis for $\mathcal{F}$ over $\mathrm{Frac}(A)$, and $\mathbf{f} = \{f^1, \ldots, f^n\}$ is a collection of polynomials in $A[x_1, \ldots, x_n]$ satisfying the following conditions:

1. (LP1) Each f^ℓ is (irreducible) and not divisible by any variable x_t,
2. (LP2) f^ℓ does not involve the variable x_ℓ.

We call $\mathbf{x}$ the *cluster* of S. Each element of $\mathbf{x}$ is called a *cluster variable*. Each element of $\mathbf{f}$ is called an *exchange polynomial*. Note that our exchange polynomials have much more flexibility than cluster algebra exchange polynomials.

Note the cluster variables and the exchange polynomials are unordered, but the exchange polynomial f^ℓ corresponds to the cluster variable x_ℓ. Since we have n cluster variables, we say that S is of *rank* n..

Let $S = (\mathbf{x}, \mathbf{f})$ be a seed. We define the collection $\hat{\mathbf{f}} = \{\hat{f}^1, \ldots, \hat{f}^n\}$ of *exchange Laurent polynomials* by the following conditions:

1. $\hat{f}^\ell = \frac{f^\ell}{M}$, where $M = x_1^{c_1} \ldots x_n^{c_n}$ is a monomial in the cluster variables x_t, $t \neq \ell$,
2. For each $t \neq \ell$, we have that

$$\hat{f}^\ell|_{x_t \leftarrow f^t/x} \in A[x_1^{\pm 1}, \ldots, x_{t-1}^{\pm 1}, x^{\pm 1}, x_{t+1}^{\pm 1}, \ldots, x_n^{\pm 1}],$$

   and $\hat{f}^\ell|_{x_t \leftarrow f^t/x}$ is not divisible by f^t as an element of this ring.

Note that the collections $\{f^1, \ldots, f^n\}$ and $\{\hat{f}^1, \ldots, \hat{f}^n\}$ determine each other uniquely. The power of x_t in the monomial M is equal to the largest power of f^t that divides f^ℓ upon the substitution $x_t \leftarrow f^\ell/x$.

Mutation works similarly for cluster variables. Suppose we are mutating at index k. In the cluster algebra set up,

$$\begin{aligned} x_k x_k' &= \prod_{i\to k} x_i + \prod_{k\to j} x_j \\ &= \text{binomial determined by quiver} \end{aligned}$$

In the LP algebra set up,

$$x_k x_k' = \text{exchange Laurent polynomial at } k.$$

We keep all other cluster variables the same. We also keep the exchange relation at k the same. However, we replace the exchange relations at other indices as follows:

- **Substitution step**. We set
$$\left(f_i^\ell\right)' = f^\ell\big|_{x_i \leftarrow \frac{\hat{f}^i|_{x_\ell\leftarrow 0}}{y_i}}.$$
- **Cancellation step**. We divide out by any common factors $(f_i^\ell)'$ shares with $\hat{f}^i|_{x_\ell\leftarrow 0}$. This then defines f_i^ℓ up to a monomial multiplier.
- **Normalisation step**. We multiply through by a monomial in $x_1, \ldots, y_i, \ldots, x_n$ to make f_i^ℓ satisfy (LP1) and (LP2) as an exchange polynomial in S_i. Such a monomial will be uniquely defined only up to a unit. Thus f_i^ℓ is only defined up to a unit multiplier.

Examples:

1. Let
$$S = \begin{cases} x_1 & 1 + x_2 + x_3 \\ x_2 & x_1 + x_3 \\ x_3 & x_1^2 + x_2 \end{cases}$$
Mutate this seed at index three. The first step is to compute the exchange Laurent polynomials. We do not actually have to divide anything this time, so they're the same as the exchange polynomials. Next, we replace x_3 with
$$y_3 = \frac{x_1^2 + x_2}{x_3}.$$
Then we replace other exchange polynomials. In index two:
$$(x_1 + x_3)\big|_{x_3 \leftarrow \frac{x_1^2}{y_3}} = x_1 + \frac{x_1^2}{y_3}$$

This shares a factor of x_1 with $f_3|_{|x_2 \leftarrow 0}$, so we cancel this out, leaving $1 + \frac{x_1}{y_3}$. Finally we multiply through by the monomial y_3 to get $x_1 + y_3$ as the final exchange polynomial. Similarly with the first exchange polynomial, one eventually gets

$$x_2 * y_3 + x_2 + y_3$$

which gives a final seed of

$$S = \begin{cases} x_1 & x_2 y_3 + x_2 + x_3, \\ x_2 & x_1 + y_3, \\ y_3 & x_1^2 + x_2. \end{cases}$$

2. Let

$$S = \begin{cases} x_1 & 1 + x_2, \\ x_2 & 1 + x_1. \end{cases}$$

Exercise: Check that this seed encodes the *pentagon recurrence*. There should be five seeds in the exchange graph.

All cluster algebras of the form I gave above give rise to LP algebras (possibly with reducible polynomials; then we need to take incidence matrix to have primitive columns). But the key point is that nobody knows the classification of finite-type LP algebra seeds. For example, initial seed that comes from cluster variety induced by Cayley plane:

x_1	$1 + y_3$
x_2	$x_1 x_3 x_4 + x_1 x_3 + x_1 x_4 + x_1 y_3 + x_3 y_3 + x_3$
x_3	$x_2 + x_4 + y_3$
x_4	$x_1 x_3 + x_2 x_3 + x_2 + x_3 + y_3$
y_3	$x_1 x_2 x_4 + x_1 x_3 x_4 + x_1 x_3 + x_2 x_3 + x_1 x_4 + x_2 x_4 + x_2 + x_3$

This has 264 seeds and 32 cluster variables.

3 Linear LP algebras

This section of the talk is about what I have been studying.

3.1 Definition

A linear LP algebra is an LP algebra whose seed pattern contains a seed with linear exchange polynomials. Note that this does *not* imply all seeds in the pattern are linear. Easy examples and nonexamples.

Lam and Pylyavskyy conjectured in their initial paper that all linear LP algebras are finite type. Is this true? Yes!

3.2 Graph LP algebras

Given a directed graph Γ, we define an initial seed S_Γ with cluster variables $\{x_1, \ldots, x_n\}$ and exchange polynomials $f^\ell = a_\ell + \sum_{\ell \to k} x_k$. All LP algebras that arise like this are finite type (this is the subject of Lam and Pylyavskyy's second paper).

3.3 Proof that all linear LP algebras are finite type

Start with the fact we already know from Lam and Pylyavskyy: A rank two seed is finite-type *if and only if* the product of the degrees of the polynomials is less than 4.

Start with the "most generic" linear LP algebra with exchange polynomials

$$f^\ell = a_0^\ell + \sum_{i=1, i \neq \ell}^{n} a_i^\ell x_i.$$

I am imposing here that there are no relations among the coefficients a_i. Thinking of all possible linear LP algebra seeds with this many variables parameterised by the coefficients a_i^ℓ, this seed represents a generic point of this space. Using combinatorial arguments, we can show that seeds like this are finite-type. (Hint: It boils down to the pentagon recurrence!)

1. For a generic seed, mutating at distinct indices means we never have to perform the cancellation step of mutation. So mutation is just substitution + multiplying by a monomial.

2. Exchange polynomials stay linear in those variables we have not mutated at yet.

3. Exchange polynomials we have *already mutated at* are linear in the variables we have *just* mutated at.

4. This means at each step we can either mutate at a variable we have not mutated at yet, or we can mutate at a variable we have already mutated at, which restricts to two linear polynomials and is thus finite type (in fact, a pentagon recurrence).

5. Since we have only finitely many variables we have not mutated at yet, the seed is finite type. We can show this formally by showing that you cannot get distinct seeds by mutating at $\text{rank}(S) + 1$ indices.

Of course, we have modded out some important technical details.

The next step is to show that introducing relations among these coefficients does not increase the number of seeds in the mutation class. This is something I believe is true for all seeds, but is something you can show directly for linear seeds. Then *any* linear LP algebra seed will have finitely many seeds in its mutation class!

4 Code demonstration

The SageMath package I wrote can perform all these computations with LP algebra seeds over standard base rings ($\mathbb{Z}, \mathbb{Q}, \mathbb{Z}[a_1, ...a_n]$, etc.)

Future plans

1. Come up with a combinatorial model for (linear) LP algebra seeds (scattering diagrams?)
2. Show some of the properties I was alluding to earlier for general LP algebra seeds
3. Move away from LP algebras in the abstract and study closer the algebraic geometry problems that they model/solve
4. Improve my code: Implement graph LP algebras, make it work with reducible polynomials correctly, allow more base rings, maybe add some combinatorial tools.