Cluster algebras can be viewed as the "gluing together" of a bunch of smaller cluster algebras. Our goal here is to create new cluster polylogarithms by "gluing together" smaller cluster polylogarithms. More specifically, the problem we are trying to solve is: given some algebra H, define a function f_H which respects the automorphisms of H and is built out of terms which themselves respect automorphisms of subgroups of H.

In practice, we find these functions via a bootstrap procedure, starting simply and building into progressively more complicated algebras. Heuristically, the method is:

1. Begin with an algebra G and a function which respects the automorphisms of that algebra, f_G.
2. Take a larger algebra H which contains G as a subalgebra. Generically there will be many distinct instantiations of G across H, we'll label them $G^{(1)}, \ldots, G^{(n)}$.
3. Define an ansatz $f_H = \sum_{i=1}^{n} c_i f_{G^{(i)}}$ for some unknown constants c_i.
4. Fix the parameters by requiring that f_H respects the automorphisms of H.

We'll see that in some cases, for a given H and G, there are no choices of the c_i parameters which satisfy the automorphisms of H. And in many cases there will be multiple free parameters left, which we will aim to fix by other means.

Of course it is useful to keep in mind that the overall goal here is to define cluster polylogarithms which will be useful in writing down amplitudes. To that end, our target is $R_7^{(2)}$, which we can think of as living on the $\mathrm{Gr}(4,7) \simeq E_6$ cluster algebra. E_6 has the subalgebras $A_2, A_3, A_4, D_4, A_5, D_5$, so we will start with $G = A_2$ and try to bootstrap our way up to E_6. And while there are many cases in which these Dynkin-type cluster algebras are isomorphic to $\mathrm{Gr}(n,k)$ algebras, we will ignore that connection until we get to E_6. It is better to think of the intermediary algebras in terms of abstract coordinates (a_i and x_i for $\mathcal{A}$-coordinatesand $\mathcal{X}$-coordinates, respectively), which we can port over to Grassmannian coordinates (Pluckers and cross-ratios, respectively) when the time comes.

Furthermore, for the time being we will only be defining these functions at the level of their $B_2 \wedge B_2$ coproduct. We leave more complete definitions – up to the level of the symbol, for example – for future work.

A_2 function

Taking our seed cluster as $x_1 \to x_2$, we generate clusters of the form $\{1/\mathcal{X}_i, \mathcal{X}_{i+1}\}$ where the $\mathcal{X}$-coordinates are

$$\mathcal{X}_1 = \frac{1}{x_1}, \quad \mathcal{X}_2 = x_2, \quad \mathcal{X}_3 = x_1(1+x_2), \quad \mathcal{X}_4 = \frac{1+x_1+x_1x_2}{x_2}, \quad \mathcal{X}_5 = \frac{1+x_1}{x_1x_2}. \tag{1}$$

These satisfy the exchange relation $1 + \mathcal{X}_i = \mathcal{X}_{i-1}\mathcal{X}_{i+1}$ and Abel's identity $\sum\{\mathcal{X}_i\}_2 = 0$. As for symmetries, A_2 has a cyclic symmetry, $\sigma : \mathcal{X}_i \to \mathcal{X}_{i+1}$, and a flip symmetry $\tau : \mathcal{X}_i \to \mathcal{X}_{6-i}$.

To define a function on A_2, we:

1. Begin with an ansatz $f_{A_2} = \sum c_{ij}\{\mathcal{X}_i\}_2 \wedge \{\mathcal{X}_j\}_2$. This has 10 terms, only 6 of them are linearly independent thanks to Abel's identity.

2. Impose integrability (see old work for details). This fixes 5 parameters.

3. Impose symmetries: for each symmetry, we consider the case where f_{A_2} is either invariant or "covariant" – i.e. $\sigma(f_{A_2}) = f_{A_2}$ or $\sigma(f_{A_2}) = -f_{A_2}$.

In total there are 4 choices for how f_{A_2} behaves under automorphisms, however not all of them are compatible with integrability.

	$\sigma^+\tau^+$	$\sigma^+\tau^-$	$\sigma^-\tau^+$	$\sigma^-\tau^-$
f_{A_2}:	9	5	0	0