Published for SISSA by Springer

Received: *June 5, 2015*
Accepted: *August 1, 2015*
Published: *August 25, 2015*

Orientations of BCFW charts on the Grassmannian

Timothy M. Olson

Michigan Center for Theoretical Physics, Randall Laboratory,
Department of Physics, University of Michigan,
450 Church St, Ann Arbor, MI 48109, U.S.A.

E-mail: timolson@umich.edu

Abstract: The Grassmannian formulation of $\mathcal{N} = 4$ super Yang-Mills theory expresses tree-level scattering amplitudes as linear combinations of residues from certain contour integrals. BCFW bridge decompositions using adjacent transpositions simplify the evaluation of individual residues, but orientation information is lost in the process. We present a straightforward algorithm to compute relative orientations between the resulting coordinate charts, and we show how to generalize the technique for charts corresponding to sequences of any not-necessarily-adjacent transpositions. As applications of these results, we demonstrate the existence of a signed boundary operator that manifestly squares to zero and prove via our algorithm that any residues appearing in the tree amplitude sum are decorated with appropriate signs so all poles that appear twice cancel exactly, not just mod 2 as in previous works. We also identify other non-physical poles in the boundary of Grassmannian representations of the amplitude and justify their removal so that the final result contains only physical local poles.

Keywords: Supersymmetric gauge theory, Scattering Amplitudes, Extended Supersymmetry

ArXiv ePrint: 1411.6363

Contents

1 Introduction

Scattering amplitudes in 4d planar $\mathcal{N} = 4$ super Yang-Mills (SYM) theory can be formulated as contour integrals in the space of $k \times n$ matrices, modulo multiplication by a $GL(k)$ matrix; this is the Grassmannian manifold $\mathrm{Gr}(k, n)$ [1]. Reformulating scattering amplitudes in this new framework has led to many interesting and unexpected mathematical structures in $\mathcal{N} = 4$ SYM such as on-shell diagrams [1–3] and the amplituhedron [4–6]. Many of the results extend also to 3d $\mathcal{N} = 6$ ABJM theory [7, 8] where several novel properties have emerged [9–11].

This paper will focus on planar $\mathcal{N} = 4$ SYM theory, where the n-particle N^{k}MHV tree amplitude can be obtained by evaluating the following integral:

$$\mathcal{A}_n^{(k)} = \mathcal{A}_n^{\mathrm{MHV}} \oint_\Gamma \frac{d^{k\times n}C}{GL(k)\, M_1 M_2 \ldots M_n} \delta^{4k|4k}(C \cdot \mathcal{Z}). \tag{1.1}$$

The prefactor $\mathcal{A}_n^{\mathrm{MHV}}$ is the n-particle MHV amplitude, C is a $k \times n$ matrix of full rank, and $\mathcal{Z}$ encodes the external data (momentum, particle type, etc.) as super-momentum twistors. In the denominator of the measure, M_i is the i^{th} consecutive k-minor of C, which means that it is the determinant of the submatrix of C with ordered columns $i, i+1, \ldots, i+k-1 \pmod n$. The contour on which the integral should be evaluated is designated by Γ. The result is a sum over residues computed at poles of the integrand. There are generally many families of contours which produce equivalent representations of the amplitude due to residue theorems.

Each contour can be thought of as a product of circles wrapping around poles of the integrand, the points where certain minors vanish. The minors are degree-k polynomials in the variables of integration, so the pole structure is generally very difficult to describe in the formulation (1.1). However, a technique was presented in [1] for generating charts on the space such that any individual codimension-1 residue occurs at a simple logarithmic singularity. Using one of those charts, the measure on a d-dimensional submanifold can be decomposed into a product of $d\alpha/\alpha = \mathrm{dlog}\,\alpha$ quantities. We call this measure ω for future reference:

$$\omega = \frac{d^{k\times n}C}{GL(k)\, M_1 M_2 \ldots M_n} \to \mathrm{dlog}\,\alpha_d \wedge \mathrm{dlog}\,\alpha_{d-1} \wedge \ldots \wedge \mathrm{dlog}\,\alpha_1. \tag{1.2}$$

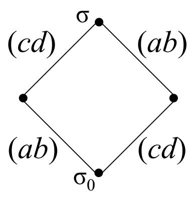
Advantages to this formulation are that such charts are easy to generate and that every codimension-1 residue can be reached as a dlog singularity using only a small atlas of charts. A potential disadvantage is that directly relating two distinct charts is non-trivial; this could lead to sign ambiguities when combining individual residues into the amplitude. This is especially important because the residues contain non-local singularities that should cancel in the tree amplitude sum. Previously, such divergences were shown to appear in pairs, so they at least cancel mod 2 [1]. In section 4 of this paper, we demonstrate that for doubly-appearing poles the cancellation is exact,[1] which follows from the main result of this paper: the master algorithm introduced in section 3. However, we observe the presence of other non-physical singularities that appear to contribute individually in some Grassmannian representations of tree amplitudes but not in others. We describe the sources of such poles and explain how momentum conservation allows one to eliminate those apparently problematic singularities.

Summary of results. The key development of this paper is a systematic algorithm that generates the relative sign between any two BCFW charts on a submanifold, or *cell*, of the Grassmannian. Each chart is defined by a sequence of transpositions $(a_1 b_1)(a_2 b_2) \ldots (a_d b_d)$

[1]The relative signs between NMHV residues were derived in [11], but that method does not easily generalize to higher-k amplitudes.

acting on a permutation labeling a 0-dimensional cell. Equivalently, each chart can be represented by a path through the poset (partially ordered set) of Grassmannian cells. Every transposition $(a_i b_i)$ corresponds to a factor of $\mathrm{dlog}\,\alpha_i$ in (1.2).

To get a sense of the result, it is illustrative to consider the simple example of two charts defined by the sequences $(ab)(cd)$ and $(cd)(ab)$ with $a < b < c < d < a{+}n$. In the poset of cells, these sequences define distinct paths from a 0-dimensional cell labeled by σ_0 to a 2-dimensional cell labeled by σ, shown in (1.3) with edges labeled by the corresponding transpositions:

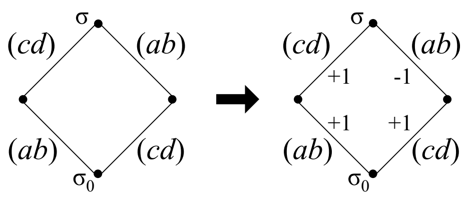


(1.3)

If we associate the coordinate α_1 with (ab) and α_2 with (cd), then the dlog forms generated by the sequences are, respectively,

$$\omega = \mathrm{dlog}\,\alpha_1 \wedge \mathrm{dlog}\,\alpha_2, \quad \text{and} \quad \omega' = \mathrm{dlog}\,\alpha_2 \wedge \mathrm{dlog}\,\alpha_1. \tag{1.4}$$

Clearly the two forms differ by an overall sign, so the two coordinate charts are oppositely oriented. We can encode this property in the poset by weighting the edges with ± 1 such that the product of the edge weights around the loop in (1.3) is -1. One choice of suitable signs is



(1.5)

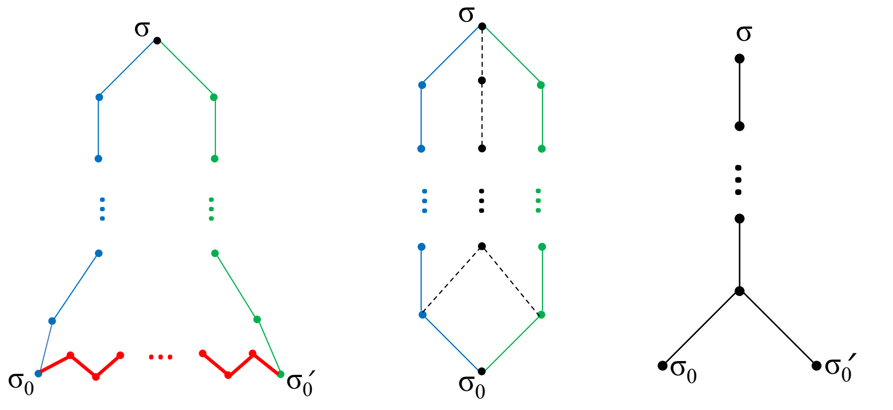
Then the relative orientation is given by the product of edge signs along the two paths.

We show in section 3.3 that the relative orientation between *any* two charts is equal to the product of edge signs around a closed loop in the poset. All the edges can be weighted such that the product of signs around every quadrilateral is -1, just as in the example (1.5).[2] The closed loop is obtained by concatenating each input path (blue and green solid lines in (1.6 A)) together with a sawtooth path connecting their 0d cells (thick red line in (1.6 A)) times -1 for each 1d cell along the connecting path. We call this method the master algorithm.

To prove that the master algorithm correctly yields the relative orientation, we introduce two preliminary algorithms in section 3.1. When the two paths meet at a common 0d cell, illustrated in (1.6 B) with blue and green solid lines, algorithm 1 splits the big loop into smaller loops, e.g. using the dashed black lines in (1.6 B), for which the relative

[2]A technique developed by T. Lam and D. Speyer for assigning edge weights is presented in appendix B [12].

orientations can be computed directly from the corresponding plabic networks.[3] If the paths end in different 0d cells, e.g. (1.6 C), algorithm 2 additionally computes the relative sign between the source cells:



(1.6)

(A) Concatenated paths (B) Shared 0d cell (C) Different 0d cells

We then use algorithms 1 and 2 in section 3.2 to demonstrate that the big loop (1.6 A) can always be split up into quadrilaterals, each of which contributes a factor of -1 to the overall sign. Finally in section 3.3 it is shown that if the edges are appropriately decorated with ± 1, then only those that make up the loop contribute to the end result (in addition to -1 from each 1d cell along the sawtooth path).

In section 4, we show that the edge-weighting rules can also be used to compute relative signs between distinct residues by requiring further that all residue theorems are mutually consistent. Consequently, residues contributing to the tree amplitude are always decorated with signs so that all non-physical singularities that appear in pairs cancel exactly in the sum of residues. In the context of the on-shell BCFW recursion relations described in [1], this forces every term to have an appropriate sign whereby all non-local poles are guaranteed to cancel. A similar argument demonstrates the existence of a boundary operator that manifestly squares to zero. Some individual non-physical poles appear with unit multiplicity in the boundary of amplitude representations and thus do not cancel, so we identify the origin of those poles and demonstrate that the associated residues vanish. In addition, we present an interpretation of charts corresponding to decompositions with non-adjacent transpositions. We review the necessary background in section 2 before presenting the algorithms in section 3, and a few details are relegated to appendices.

During the development of this analysis, a different method for determining relative orientations was proposed independently by J. Bourjaily and A. Postnikov using determinants of certain large matrices. We computed the orientations of 500 distinct charts on the 10d cell $\{5,3,8,9,6,7,12,10,14,11\} \in \mathrm{Gr}(3,10)$ with both methods and found perfect agreement [14]. The algorithms presented in this paper have also been verified by checking a variety of charts whose orientations are known by other methods. The cancellation of all doubly-appearing poles in the tree amplitude has been confirmed explicitly for all

[3]We thank R. Karpman for many helpful ideas and discussions regarding the details of the graph transformations [13].

$n = 5, \ldots, 13$ and $k = 3, \ldots, \lfloor n/2 \rfloor$, and the elimination of all individual non-physical poles using the criteria defined in section 4 have been checked for the same set of parameters. All of the algorithms described here have been implemented in Mathematica as an extension of the *positroids* package included with the arXiv version of [15].

2 Background

2.1 Positroid stratification

The positroid stratification is a decomposition of the Grassmannian $\mathrm{Gr}(k, n)$ into submanifolds called *positroid cells* (or just *cells*). Cells can be classified according to the ranks of submatrices constructed out of cyclically consecutive columns of a representative matrix. The *top cell* of $\mathrm{Gr}(k, n)$ is the unique cell of highest dimension, $d = k(n - k)$. All maximal $(k \times k)$ minors are non-vanishing in the top cell, so all chains of consecutive columns will be full rank. For example, a representative matrix of the top cell of $\mathrm{Gr}(2, 4)$ is

$$C = \begin{pmatrix} 1 & 0 & c_{13} & c_{14} \\ 0 & 1 & c_{23} & c_{24} \end{pmatrix}, \tag{2.1}$$

where $c_{ij} \in \mathbb{C}$ are coordinates on the manifold such that none of the 2×2 minors vanish. All four parameters must be fixed to specify a point in the top cell, which is consistent with the expected dimension $d = 2(4 - 2) = 4$.

Lower dimensional cells are reached by fixing relations among the entries so that additional linear dependencies arise among consecutive columns. From a given cell, the accessible codimension-1 submanifolds are called the *boundaries* of that cell. The extra linear relations imply that various minors vanish in the measure of (1.1). Thus going to the boundary should be interpreted as taking a residue at the corresponding pole. Note that choosing a particular representative matrix amounts to selecting a chart on the cell, so some poles may not be accessible in certain charts. In the example above, the boundary where columns $\vec{c}_2$ and $\vec{c}_3$ are parallel is accessible by setting $c_{13} = 0$, but there is no way to reach the boundary where columns $\vec{c}_1$ and $\vec{c}_2$ are parallel in the stated chart. The latter boundary could be accessed in a $GL(2)$-equivalent chart where a different set of columns were set to the identity.

A partial order can be defined on the set of positroid cells by setting $C \prec C'$ whenever C is a codimension-1 boundary of C' [16]. The poset structure is interesting for a number of reasons, several of which will be mentioned throughout the text. One such reason is that the poset of positroid cells in $\mathrm{Gr}(k, n)$ is isomorphic to a poset of *decorated permutations*. Decorated permutations are similar to standard permutations of the numbers $1, \ldots, n$, but differ in that k of the entries are shifted forward by n. To simplify the notation, we will use often 'permutation' to mean 'decorated permutation' since only the latter are relevant to us. Permutations will be written single-line notation using curly brackets; an example is given in (2.2). When referencing specific elements of a permutation, we will use the notation $\sigma(i)$ to mean the i^{th} element of the permutation σ, and with the understanding that $\sigma(i+n) = \sigma(i)+n$. A decorated permutation encodes the ranks of cyclically consecutive submatrices by recording, for each column $\vec{c}_a$, the first column $\vec{c}_b$ with $a \leq b \leq a+n$ such that $\vec{c}_a \in \mathrm{span}(\vec{c}_{a+1}, \vec{c}_{a+2}, \ldots, \vec{c}_b)$. The first inequality is saturated when $\vec{c}_a = 0$, and the second is saturated when $\vec{c}_a$ is linearly independent of all other columns. Continuing with the example (2.1), the top cell of Gr(2, 4) corresponds to the permutation

$$\sigma_{\text{top}} = \{3, 4, 5, 6\}, \tag{2.2}$$

which says that $1 \to 3$, $2 \to 4$, $3 \to 5 \equiv 1$, and $4 \to 6 \equiv 2$. In terms of the linear dependencies among columns of a representative matrix, it means $\vec{c}_1 \in \mathrm{span}(\vec{c}_2, \vec{c}_3)$, $\vec{c}_2 \in \mathrm{span}(\vec{c}_3, \vec{c}_4)$, $\vec{c}_3 \in \mathrm{span}(\vec{c}_4, \vec{c}_5) \equiv \mathrm{span}(\vec{c}_4, \vec{c}_1)$, and $\vec{c}_4 \in \mathrm{span}(\vec{c}_5, \vec{c}_6) \equiv \mathrm{span}(\vec{c}_1, \vec{c}_2)$.

Going to the boundary of a cell involves changing the linear relations among consecutive columns, so in permutation language, the boundary is accessed by exchanging two entries in σ. The transposition operation that swaps the elements at positions a and b is denoted by (ab), with $a < b < a+n$; we call this the *boundary operation*. Note that σ only has positions $1, \ldots, n$, but b can be greater than n, To account for this, we use

$$\sigma'(a) = \sigma(b) = \sigma(b-n) + n \quad \text{and} \quad \sigma'(b) = \sigma(a) \Rightarrow \sigma'(b-n) = \sigma(a) - n. \tag{2.3}$$

In the example (2.1), taking the boundary where $c_{13} = 0$ lands in a cell with linear dependencies specified by

$$\sigma' = \{4, 3, 5, 6\}. \tag{2.4}$$

This corresponds to exchanging the first and second elements of σ_{top}, i.e. the transposition $(1\,2)$.

Not all exchanges are allowed; for instance, applying the same transposition twice would revert back to the initial cell, which is clearly not a boundary. The allowed boundary transpositions satisfy the following criteria:

$$a < b \leq \sigma(a) < \sigma(b) \leq a+n \quad \text{and} \quad \sigma(q) \notin (\sigma(a), \sigma(b)) \;\; \forall q \in (a, b), \tag{2.5}$$

where (a, b) means the set $\{a + 1, b + 2, \ldots b - 1\}$. We will call (ab) an *adjacent* transposition if all $q \in (a, b)$ satisfy $\sigma(q) \equiv q \bmod n$ [1]. The reason for this distinction will become clear in the next section. A *strictly adjacent* transposition is of the form $(a\; a+1)$. For notational purposes, we define (ab) to act on the right, so if σ is a boundary of $\tilde{\sigma}$, then we write $\sigma = \tilde{\sigma} \cdot (ab)$. Of course, (ab) is its own inverse, so acting on the right with (ab) again yields the *inverse boundary operation* $\sigma \cdot (ab) = \tilde{\sigma}$. The boundary operation reduces the dimension by one, while the inverse boundary operation increases the dimension by one. Although the notation is identical, it should be clear what we mean from the context. Taking additional (inverse) boundaries leads to expressions like $\rho = \sigma \cdot (a_1b_1)(a_2b_2)\ldots$, which means first exchange $\sigma(a_1)$ and $\sigma(b_1)$, then swap the elements at positions a_2 and b_2, etc.[4]

[4]To avoid any confusion with accessing elements of the permutation, e.g. $\sigma(i)$, we use the $\cdot$ after σ to indicate that σ is a permutation, and (ab) is a transposition. The $\cdot$ operation is implicit between neighboring transpositions.

2.2 Plabic graphs

Permutations and positroid cells can be represented diagrammatically with *plabic* (planar-bicolored) *graphs* and *plabic networks*.[5] *Plabic graphs* are planar graphs embedded in a disk, in which each vertex is colored either black or white. Any bicolored graph can be made bipartite by adding oppositely-colored bivalent vertices on edges between two identically-colored vertices, so we will assume all graphs have been made bipartite. Some edges are attached to the boundary; we will call these *external legs* and number them in clockwise order $1, 2, \ldots, n$. If a monovalent leaf is attached to the boundary, it will be called a *lollipop* together with its edge. *Plabic networks* are plabic graphs together with weights $(t_1, t_2, \ldots, t_e)$ assigned to the edges. The weights are related to coordinates on the corresponding cell, as will be discussed in the next subsection.

Given a plabic graph G, one can define a *trip permutation* by starting from an external leg a and traversing the graph, turning (maximally) left at each white vertex and (maximally) right at each black vertex until the path returns to the boundary at some vertex b. When $a \neq b$, the permutation associated with this trip has

$$\sigma_G(a) = \begin{cases} b & b > a \\ b{+}n & b < a \end{cases}. \tag{2.6}$$

We will explain the case $a = b$ momentarily.

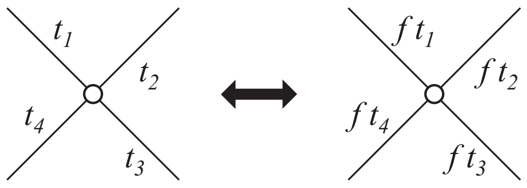
For tree amplitudes, we will be concerned with *reduced* plabic graphs. A plabic graph is reduced if it satisfies the following after deleting all lollipops [17]:

1. It has no leaves;
2. No trip is a cycle;
3. No trip uses a single edge twice;
4. No two trips share two edges e_1 and e_2 in the same order.

It follows that any external leg a for which $\sigma_G(a) \equiv a \bmod n$ must be a lollipop [17]. Specifically, we define $\sigma_G(a) = a$ to be a black lollipop, and $\sigma_G(a) = a{+}n$ to be a white lollipop.

Thus each reduced plabic graph/network corresponds to a unique decorated permutation. However, the correspondence is not a bijection; rather each permutation labels a family of reduced plabic graphs/networks. Members of each family are related by *equivalence moves* that modify the edge weights but leave the permutation unchanged [1, 18]:

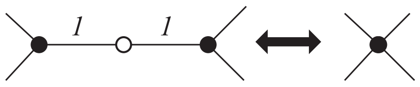
(E1) ***GL*(1) rotation:** at any vertex, one can perform a $GL(1)$ rotation that uniformly scales the weights on every attached edge, e.g. for a scaling factor f,



(2.7)

[5]Physical interpretations of certain plabic networks, also known as 'on-shell diagrams,' have been explored in several recent papers, including [1, 3].

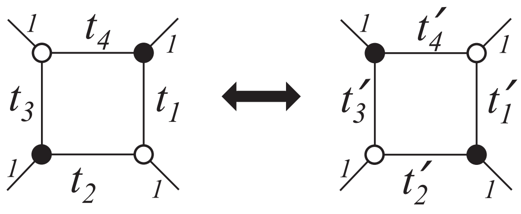
(E2) Merge/delete: any bivalent vertex whose edges both have weight 1 can be eliminated by merging its neighbors into one combined vertex and deleting the bivalent vertex and its edges, e.g.



(2.8)

If one of its neighbors is the boundary, then the bivalent vertex should be merged with the boundary instead. The inverse operation can also be used to 'unmerge' a vertex or boundary.

(E3) Square move: a four-vertex square with one pattern of coloring is equivalent to the four-vertex square with opposite coloring, e.g.



(2.9)

The edge weights are related (using the sign conventions of [2]):

$$t_1' = \frac{t_3}{t_1 t_3 + t_2 t_4}, \quad t_2' = \frac{t_4}{t_1 t_3 + t_2 t_4}, \quad t_3' = \frac{t_1}{t_1 t_3 + t_2 t_4}, \quad t_4' = \frac{t_2}{t_1 t_3 + t_2 t_4}. \tag{2.10}$$

From the plabic network, one can also read off a representative $k \times n$ matrix for the corresponding positroid cell, which is called the *boundary measurement matrix* (or *boundary measurement map*). Several related methods exist for defining boundary measurements such as using perfect orientations [1, 2, 18] or perfect matchings [2, 17, 19]. The equivalence moves above can be derived by requiring that the boundary measurements are unchanged by each transformation. In the next subsection, we will show how to construct the boundary measurements systematically for the types of plabic networks relevant to amplitudes. We refer the reader to the above references for more details, though we caution that the transformation rules are presented slightly differently.

These three moves are sufficient to transform any plabic network into any other in its equivalence class. In addition, using these moves one can always fix all but d of the edge weights to unity in any network that represents a d-dimensional cell; throughout the rest of this paper, edges with unspecified weights will have weight 1. As we will see shortly, certain planar networks lead to especially convenient paramaterizations of positroid cells. The equivalence moves will allow us to compare the resulting oriented forms.

2.3 Bridge decompositions and charts

To write down a coordinate chart on a given cell, it is sufficient to construct a representative matrix with the appropriate linear dependencies among its columns. However, in a general chart the boundary structure could be very difficult to identify. Fortunately, there is a straightforward method to construct charts with simple dlog forms as in (1.2) [1]; we review this technique below and how it relates to paths in the poset of cells. Some paths through the poset do not correspond to any such charts, so we suggest an interpretation for those paths in section 2.3.2 and explain the consequences for residues in section 4.1.

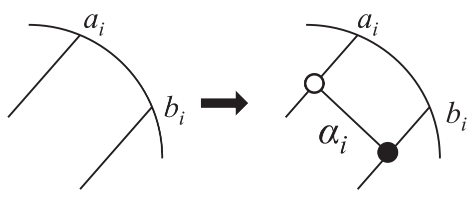
2.3.1 Standard BCFW bridge decompositions

Positroid cells of dimension zero in $\mathrm{Gr}(k,n)$ correspond to unique plabic graphs made solely out of lollipops with edge weight 1. The k legs with $\sigma(a) = a+n$ have white vertices while the rest are black. The boundary measurement matrix is zero everywhere except the submatrix composed out of the k columns corresponding to the white vertices, which together form a $k \times k$ identity matrix. There are no degrees of freedom, so the differential form (1.2) is trivial, $\omega = 1$.

Since there is a unique representative plabic network for each 0d cell, we will build higher-dimensional representatives out of the set of 0d networks. In the poset, higher-dimensional cells can be reached from 0d cells by repeatedly applying the inverse boundary operation defined below (2.5). Equivalently, a d-dimensional cell can be decomposed into a sequence of adjacent transpositions acting on a 0d permutation, e.g.[6]

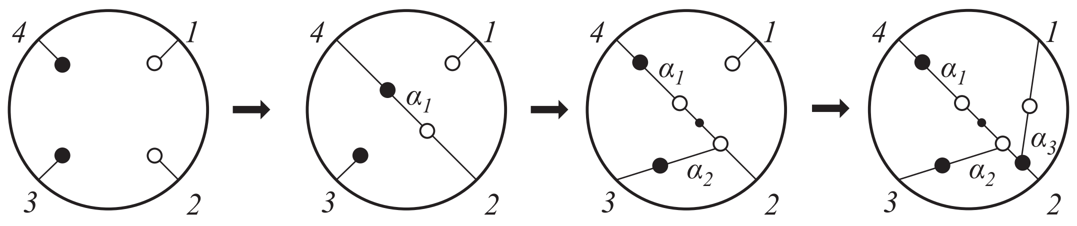
$$\{3,5,4,6\} = \{5,6,3,4\} \cdot (24)(23)(12). \tag{2.11}$$

This is called a *BCFW decomposition* and leads to a convenient graphical representation. In a planar network, an adjacent transposition $(a_i b_i)$ amounts to simply adding a white vertex on leg a_i, a black vertex on leg b_i, and an edge between them with weight α_i:



(2.12)

This is called a *(BCFW) bridge*. Note that if one of the legs is initially a lollipop, then the resulting leaf should be deleted after adding the bridge [17, 18]. The example (2.11) generates the following sequence of graphs:



(2.13)

[6]The transposition (24) is adjacent because it acts on a permutation whose third element is self-identified.

We have added a black vertex between the first two bridges to make the graph bipartite; it is drawn slightly smaller to distinguish it from the bridge vertices. Many simplifications are possible using the equivalence moves (2.7)–(2.9).

Adding a BCFW bridge affects the trip permutation by exchanging $\sigma_G(a) \leftrightarrow \sigma_G(b)$ as desired, and the boundary measurement matrix transforms in a simple way [1],

$$\vec{c}_b \to \vec{c}_b + \alpha_i \vec{c}_a. \tag{2.14}$$

One can easily check that the linear dependencies of the shifted matrix agree with the expected permutation as long as $(a_i b_i)$ is an adjacent transposition. If ω_{i-1} is the differential form associated with the initial cell, then after adding the bridge, the new form is

$$\omega_i = \operatorname{dlog} \alpha_i \wedge \omega_{i-1}. \tag{2.15}$$

This prescription provides a robust way to generate coordinates on any cell in $\mathrm{Gr}(k,n)$.

There are generally many ways to decompose a d-dimensional permutation σ into a sequence of adjacent transpositions acting on a 0d cell. In fact, no single chart covers all boundaries of a generic cell. However, an atlas of at most n standard BCFW charts is sufficient to cover all boundaries [1]. Every such chart defines a unique path through the poset from σ to some σ_0. In the example (2.11), the path was

$$\sigma = \{3,5,4,6\} \xrightarrow{(12)} \{5,3,4,6\} \xrightarrow{(23)} \{5,4,3,6\} \xrightarrow{(24)} \{5,6,3,4\} = \sigma_0. \tag{2.16}$$

Not all BCFW decompositions of σ end in the same 0d cell, however. Another BCFW decomposition of $\{3,5,4,6\}$ ends in $\sigma_0' = \{5,2,7,4\}$:

$$\sigma = \{3,5,4,6\} \xrightarrow{(12)} \{5,3,4,6\} \xrightarrow{(46)} \{5,2,4,7\} \xrightarrow{(34)} \{5,2,7,4\} = \sigma_0'. \tag{2.17}$$

The important point is that every BCFW decomposition corresponds to a unique path of length d that starts at σ and ends in a 0d cell. The converse is not true; some paths of length d that start at σ and end in a 0d cell do not correspond to any BCFW decomposition.

2.3.2 Generalized decompositions

When evaluating residues, one will often encounter paths through the poset that do not coincide with any single standard chart. These paths contain edges that represent non-adjacent transpositions. Continuing with the earlier example, the following path ends in the same 0d cell as (2.11), but the first transposition (13) crosses a non-self-identified leg:

$$\sigma = \{3,5,4,6\} \xrightarrow{(13)} \{4,5,3,6\} \xrightarrow{(24)} \{4,6,3,5\} \xrightarrow{(14)} \{5,6,3,4\} = \sigma_0. \tag{2.18}$$

Nevertheless, this path is certainly a possible route when evaluating residues since every codimension-1 boundary is accessible from some adjacent chart [1]. We could, for instance, take the decomposition from (2.11), and take $\alpha_2 \to 0$. It is easy to see from the boundary measurement matrix that this yields the desired linear dependencies among columns:

$$\begin{pmatrix} 1 & \alpha_3 & 0 & 0 \\ 0 & 1 & \alpha_2 & \alpha_1 \end{pmatrix} \xrightarrow{\alpha_2 \to 0} \begin{pmatrix} 1 & \alpha_3 & 0 & 0 \\ 0 & 1 & 0 & \alpha_1 \end{pmatrix}. \tag{2.19}$$

The generalization to any path through the poset is clear; each successive step involves computing the residue at a logarithmic singularity in some chart. Thus every path corresponds to some dlog form, not just the paths for which we have explicit plabic graphical representations. We will call these *generalized decompositions* and their coordinates *generalized charts*. In practice, computing residues along a particular path through the poset can always be done using standard adjacent charts, though it may involve changing coordinates at several steps along the way. As we will see in section 4.1, the sign of the resulting residue depends only on the path taken, and not on the choice of reference charts along the way.

3 Relating distinct charts

Recall that a decomposition for a d-dimensional cell C corresponds to a sequence of transpositions applied to a permutation σ_0 labeling a 0-dimensional cell C_0. Each sequence defines a particular path through the poset with endpoints at C and C_0. In this section, we will show that the relative sign between the dlog forms generated by two distinct charts can be obtained systematically. We will first show this result for standard BCFW charts constructed using only adjacent transpositions. Subsequently, we will discuss the generalized situation where we do not always have a simple graphical representation; the convention for deriving signs will be extended to cover the additional possibilities while maintaining consistency with the standard setup. The extended conventions will also lead to a simpler method for comparing charts.

3.1 Standard BCFW charts

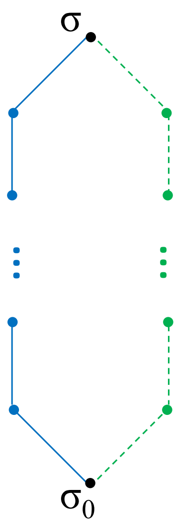
There are two cases that we need to address depending on whether the decompositions end in identical 0d cells or distinct ones. We will cover the identical case first and then deal with the other situation.

3.1.1 Charts with identical 0d cells

We assume first that the 0d cell labeled by σ_0 is the same for both paths. The d-dimensional cell labeled by σ is connected to σ_0 by two sequences of transpositions

$$\sigma = \sigma_0 \cdot (a_1 b_1)(a_2 b_2)\dots(a_d b_d) = \sigma_0 \cdot (a'_1 b'_1)(a'_2 b'_2)\dots(a'_d b'_d). \tag{3.1}$$

Graphically, the concatenation of the two paths creates a closed loop of length $2d$ in the poset, shown here schematically with one path denoted by a blue solid line and the other by a green dashed line:



(3.2)

The relative orientation of the two dlog forms corresponding to the sequences can be obtained by a simple algorithm. Before presenting that result, we will need the following lemma:

Lemma 1. *The top cell can be reached from any cell, C, of dimension d by a sequence of $k(n-k)-d$ strictly adjacent transpositions.*

Proof. Let σ be the permutation labeling $C \in \mathrm{Gr}(k,n)$. When σ contains two neighboring elements satisfying $\sigma(i) > \sigma(i+1)$ (with $\sigma(n+1) = \sigma(1)+n$), this is called an *inversion*. Such an inversion can be removed by applying the (strictly adjacent) transposition $(i\ i+1)$. Since all entries in σ_{top} are ordered, one can reach the top cell by iteratively eliminating all inversions. □

For example, the top cell of $\mathrm{Gr}(2,6)$ can be reached from the 5-dimensional cell $\{2,3,4,6,7,11\}$ by the sequence of transpositions $(6\,7)(1\,2)(2\,3)$:

$$\{2,3,4,6,7,11\} \xrightarrow{(6\,7)} \{5,3,4,6,7,8\} \xrightarrow{(1\,2)} \{3,5,4,6,7,8\} \xrightarrow{(2\,3)} \{3,4,5,6,7,8\}. \tag{3.3}$$

Using this procedure, any BCFW sequence can be extended to reach the top cell using only strictly adjacent transpositions. We thank R. Karpman for pointing this out.

Since $(i, i+1)$ does not cross any legs, the resulting sequence will be a valid BCFW sequence. Therefore, we may assume without loss of generality that the cell on which we seek to compare orientations is the top cell because two sequences which lead to a cell of lower dimension can be trivially extended to top cell sequences by appending the same transpositions to both paths. This will not affect the relative sign of the forms since both will have identical pieces appended to them.

We turn now to the sign-comparison algorithm. The idea is to compare each BCFW chart to specially chosen reference charts whose relative orientation is easy to compute. They are chosen so that at each iteration, the loop in the poset (initially of length $2d$) is shortened. Then the final relative orientation is the product over all the intermediate orientations.

Algorithm 1

Input: two BCFW sequences of length $d = k(n-k)$: $\mathbf{w} = (a_1b_1)(a_2b_2)\dots(a_db_d)$ and $\mathbf{w}' = (a'_1b'_1)(a'_2b'_2)\dots(a'_db'_d)$

Output: ± 1

Procedure:

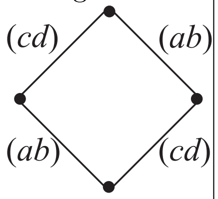
1) Let j be the smallest index such that $(a_jb_j) \neq (a'_jb'_j)$. The transpositions with $i > j$ yield a closed loop of length $\ell \leq 2d$. If there is no such position, then the paths are identical, so return $+1$.

2) Let σ label the j-dimensional cell reached by the sequence of transpositions $(a_1b_1)(a_2b_2)\dots(a_jb_j)$ and σ' label that reached by $(a'_1b'_1)(a'_2b_2)'\dots(a'_jb'_j)$. Using the following rules, construct reference charts to which the initial charts should be compared. Comparing the two reference charts produces a known sign; the relevant parts are displayed with each step, and their relative signs are derived in appendix A. There are several cases to consider (with $a < b < c < d < a+n$):

i) $\boldsymbol{(a_jb_j) = (ab),\ (a'_jb'_j) = (cd)}$

The j-dimensional cells σ and σ' have a shared $(j+1)$-dimensional neighbor $\tilde{\sigma} = \sigma \cdot (cd) = \sigma' \cdot (ab)$. Let $\mathfrak{u}$ be the sequence generated by Lemma 1 for the cell labeled by $\tilde{\sigma}$. Then the reference sequences and relative sign are:

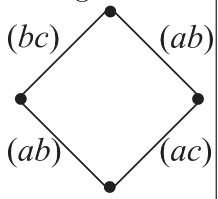
- ref. sequence 1: $\tilde{\mathbf{w}} = (a_1b_1)(a_2b_2)\dots(a_{j-1}b_{j-1})(ab)(cd)\mathfrak{u}$
- ref. sequence 2: $\tilde{\mathbf{w}}' = (a'_1b'_1)(a'_2b'_2)\dots(a'_{j-1}b'_{j-1})(cd)(ab)\mathfrak{u}$
- The relative sign between reference charts is -1.



ii) $\boldsymbol{(a_jb_j) = (ab),\ (a'_jb'_j) = (ac)}$

In this case σ and σ' have a shared $(j+1)$-dimensional neighbor $\tilde{\sigma} = \sigma \cdot (bc) = \sigma' \cdot (ab)$. Let $\mathfrak{u}$ be the sequence generated by Lemma 1 for the cell labeled by $\tilde{\sigma}$. Then the reference sequences and relative sign are:

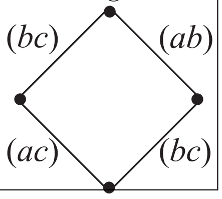
- ref. sequence 1: $\tilde{\mathbf{w}} = (a_1b_1)(a_2b_2)\dots(a_{j-1}b_{j-1})(ab)(bc)\mathfrak{u}$
- ref. sequence 2: $\tilde{\mathbf{w}}' = (a'_1b'_1)(a'_2b'_2)\dots(a'_{j-1}b'_{j-1})(ac)(ab)\mathfrak{u}$
- The relative sign between reference charts is -1.



iii) $\boldsymbol{(a_jb_j) = (ac),\ (a'_jb'_j) = (bc)}$

Again σ and σ' have a shared $(j+1)$-dimensional neighbor $\tilde{\sigma} = \sigma \cdot (bc) = \sigma' \cdot (ab)$. Let $\mathfrak{u}$ be the sequence generated by Lemma 1 for the cell labeled by $\tilde{\sigma}$. Then the reference sequences and relative sign are:

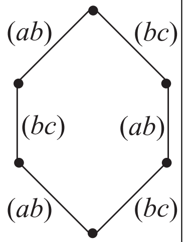
- ref. sequence 1: $\tilde{\mathbf{w}} = (a_1b_1)(a_2b_2)\dots(a_{j-1}b_{j-1})(ac)(bc)\mathfrak{u}$
- ref. sequence 2: $\tilde{\mathbf{w}}' = (a'_1b'_1)(a'_2b'_2)\dots(a'_{j-1}b'_{j-1})(bc)(ab)\mathfrak{u}$
- The relative sign between reference charts is -1.



iv) $\boldsymbol{(a_jb_j) = (ab),\ (a'_jb'_j) = (bc)}$

Using only adjacent transpositions, σ and σ' do not have a common $(j+1)$-dimensional neighbor. However, certain neighbors of σ and σ' do have a common neighbor of dimension $(j+2)$. Specifically, let $\tilde{\sigma} = \sigma \cdot (bc)$ and $\tilde{\sigma}' = \sigma' \cdot (ab)$. Then $\tilde{\sigma}$ and $\tilde{\sigma}'$ have a common $(j+2)$-dimensional neighbor $\rho = \tilde{\sigma} \cdot (ab) = \tilde{\sigma}' \cdot (bc)$. Let $\mathfrak{u}$ be the sequence generated by Lemma 1 for the cell labeled by ρ. Then the reference sequences and relative sign are:

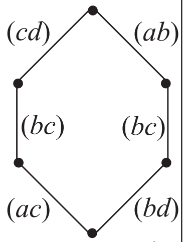
- ref. sequence 1: $\tilde{\mathbf{w}} = (a_1b_1)(a_2b_2)\ldots(a_{j-1}b_{j-1})(ab)(bc)(ab)\mathfrak{u}$
- ref. sequence 2: $\tilde{\mathbf{w}}' = (a'_1b'_1)(a'_2b'_2)\ldots(a'_{j-1}b'_{j-1})(bc)(ab)(bc)\mathfrak{u}$
- The relative sign between reference charts is $+1$.



v) $\boldsymbol{(a_jb_j) = (ac),\ (a'_jb'_j) = (bd)}$

Similar to the previous case, σ and σ' do not have a common $(j+1)$-dimensional neighbor using only adjacent transpositions. Nonetheless, with $\tilde{\sigma} = \sigma \cdot (bc)$ and $\tilde{\sigma}' = \sigma' \cdot (bc)$, then $\tilde{\sigma}$ and $\tilde{\sigma}'$ have a common $(j+2)$-dimensional neighbor $\rho = \tilde{\sigma} \cdot (cd) = \tilde{\sigma}' \cdot (ab)$. Let $\mathfrak{u}$ be the sequence generated by Lemma 1 for the cell labeled by ρ. Then the reference sequences and relative sign are:

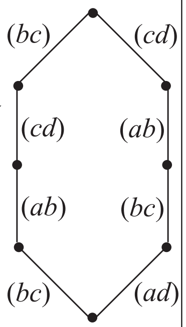
- ref. sequence 1: $\tilde{\mathbf{w}} = (a_1b_1)(a_2b_2)\ldots(a_{j-1}b_{j-1})(ac)(bc)(cd)\mathfrak{u}$
- ref. sequence 2: $\tilde{\mathbf{w}}' = (a'_1b'_1)(a'_2b'_2)\ldots(a'_{j-1}b'_{j-1})(bd)(bc)(ab)\mathfrak{u}$
- The relative sign between reference charts is -1.



vi) $\boldsymbol{(a_jb_j) = (bc),\ (a'_jb'_j) = (ad)}$

In this case, we must look further to find a shared cell above σ and σ' using only adjacent transpositions. They have a shared $(j+3)$-dimensional great-grandparent $\tilde{\rho} = \sigma \cdot (ab)(cd)(bc) = \sigma' \cdot (bc)(ab)(cd)$. Let $\mathfrak{u}$ be the sequence generated by Lemma 1 for the cell labeled by $\tilde{\rho}$. Then the reference sequences and relative sign are:

- ref. sequence 1: $\tilde{\mathbf{w}} = (a_1b_1)(a_2b_2)\ldots(a_{j-1}b_{j-1})(bc)(ab)(cd)(bc)\mathfrak{u}$
- ref. sequence 2: $\tilde{\mathbf{w}}' = (a'_1b'_1)(a'_2b'_2)\ldots(a'_{j-1}b'_{j-1})(ad)(bc)(ab)(cd)\mathfrak{u}$
- The relative sign between reference charts is -1.



3) Repeat this algorithm to compare $\mathbf{w}$ to $\tilde{\mathbf{w}}$ and $\mathbf{w}'$ to $\tilde{\mathbf{w}}'$.

4) Return the product of the relative sign from step (2) times the result of each comparison in step (3).

The relative orientation of any two BCFW charts with identical endpoints can be compared with this algorithm. In the second part of the proof below, we show that the signs and reference charts presented in step (2i) are correct; this should also serve as an illustrative example of the algorithm in action.

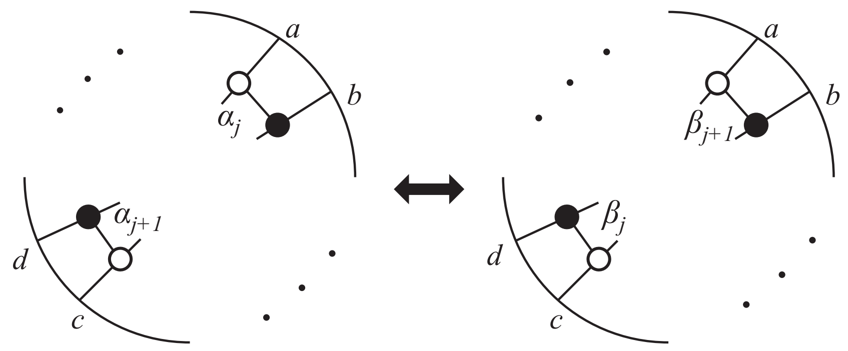
Proof. We need to show that the algorithm will terminate in a finite number of iterations, and that the sign generated at each step is correct.

- We will first show that the algorithm will terminate after a finite number of iterations. The reference sequences constructed in step (2) are chosen so that when the algorithm is called again in step (3) to compare $\mathbf{w}$ to $\tilde{\mathbf{w}}$, the new inputs satisfy $(a_i b_i) = (a'_i b'_i)$ for all $i \leq j$. The same is true for the comparison of $\mathbf{w}'$ and $\tilde{\mathbf{w}}'$. Step (1) searches for the first point at which the input sequences differ, so by construction, the next position will be at least $j + 1$, which is larger than in the previous iteration. Since j is bounded by d, the algorithm will eventually terminate.

- Next we will explain the results presented in step (2i). Since $a < b < c < d < a+n$, the two transpositions can be applied in either order without violating the adjacent requirement of BCFW sequences. Therefore σ and σ' have a common neighbor $\tilde{\sigma}$, and both $(a_1b_1)(a_2b_2)\ldots(a_{j-1}b_{j-1})(ab)(cd)\mathfrak{u}$ and $(a'_1b'_1)(a'_2b'_2)\ldots(a'_{j-1}b'_{j-1})(cd)(ab)\mathfrak{u}$ are valid BCFW sequences for the top cell. Moreover, since $(a'_i b'_i) = (a_i b_i)$ for all $i < j$, their corresponding forms differ only in positions j and $j + 1$:

$$\begin{aligned}\omega &= \mathrm{dlog}\,\alpha_d \wedge \mathrm{dlog}\,\alpha_{d-1} \ldots \wedge \mathrm{dlog}\,\alpha_{j+2} \wedge \mathrm{dlog}\,\alpha_{j+1} \wedge \mathrm{dlog}\,\alpha_j \wedge \mathrm{dlog}\,\alpha_{j-1} \ldots \wedge \mathrm{dlog}\,\alpha_1, \\ \omega' &= \mathrm{dlog}\,\alpha_d \wedge \mathrm{dlog}\,\alpha_{d-1} \ldots \wedge \mathrm{dlog}\,\alpha_{j+2} \wedge \mathrm{dlog}\,\beta_{j+1} \wedge \mathrm{dlog}\,\beta_j \wedge \mathrm{dlog}\,\alpha_{j-1} \ldots \wedge \mathrm{dlog}\,\alpha_1,\end{aligned} \tag{3.4}$$

  where α_j and α_{j+1} are the weights associated with (ab) and (cd) in the first sequence, while β_j and β_{j+1} are associated with (cd) and (ab) in the second sequence. To determine the relationship between the two forms, we will use the plabic graph representations of the transpositions as BCFW bridges. Focusing on the j and $j + 1$ parts of the graph, we find the following:



(3.5)

The left and right diagrams can only be equivalent if $\beta_j = \alpha_{j+1}$ and $\beta_{j+1} = \alpha_j$, which implies

$$\omega' = \mathrm{dlog}\,\alpha_d \wedge \ldots \wedge \mathrm{dlog}\,\alpha_{j+2} \wedge \mathrm{dlog}\,\alpha_j \wedge \mathrm{dlog}\,\alpha_{j+1} \wedge \mathrm{dlog}\,\alpha_{j-1} \ldots \wedge \mathrm{dlog}\,\alpha_1 = -\omega. \tag{3.6}$$

Thus the relative sign between the reference charts is -1.

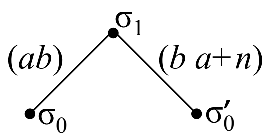
The remaining cases are described in appendix A. □

Thus we now have an algorithm that correctly computes the relative signs for BCFW forms generated from identical 0d cells.

3.1.2 Charts with distinct 0d cells

The next step will be to extend the result to decompositions that terminate in distinct 0d cells. Let us start with the simplest case: finding the relative sign between two charts on a 1d cell labeled by σ_1. Since it is 1-dimensional, all but two entries in σ are self-identified mod n. The two non-trivial positions can be labeled a and b such that $a < b < a+n$.

The 1d cell has exactly two boundaries, which can be accessed respectively by (ab) or $(b\ a+n)$, so the two BCFW sequences whose charts we will compare are (ab) and $(b\ a+n)$:

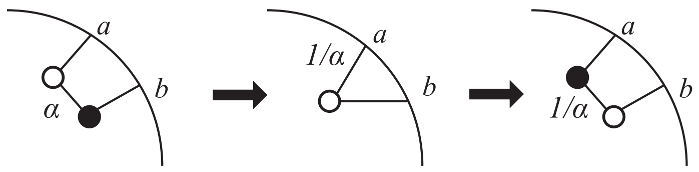


(3.7)

It is straightforward to find the relative orientations of the corresponding forms,

$$\omega_1 = \mathrm{dlog}\,\alpha \quad \text{and} \quad \omega_1' = \mathrm{dlog}\,\beta, \tag{3.8}$$

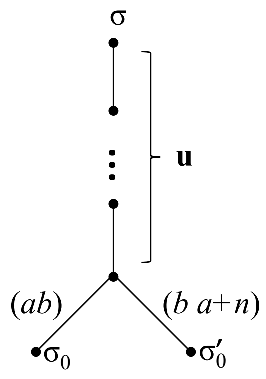
using basic plabic graph manipulations. As we see in the following diagrams, several $GL(1)$ rotations (2.7) combined with a merge and unmerge with the boundary (2.8) are sufficient to discover that $\beta = 1/\alpha$,



(3.9)

Therefore, the forms are oppositely oriented, i.e. $\omega_1' = -\omega_1$.

More generally, we could consider charts on d-dimensional cells whose corresponding BCFW sequences are identical everywhere except for the first transposition:



(3.10)

The associated forms are

$$\begin{aligned}\omega_d &= \text{dlog}\,\alpha_d \wedge \text{dlog}\,\alpha_{d-1} \wedge \ldots \wedge \text{dlog}\,\alpha_1, \\ \omega'_d &= \text{dlog}\,\alpha_d \wedge \text{dlog}\,\alpha_{d-1} \wedge \ldots \wedge \text{dlog}\,\beta_1.\end{aligned} \tag{3.11}$$

For example, we could find a Lemma 1 sequence, $\mathfrak{u}$, for the 1d cell above and compare the two charts defined by $\sigma_0 \cdot (ab)\mathfrak{u}$ and $\sigma'_0 \cdot (b\,a{+}n)\mathfrak{u}$. Since BCFW sequences such as $\mathfrak{u}$ use only adjacent transpositions, no bridge will ever be attached to legs a and b further from the boundary than the first bridge (ab), resp., $(b\,a{+}n)$. Therefore, one can apply very similar logic as in the 1d case[7] to find that $\beta_1 = 1/\alpha_1$, so $\omega'_d = -\omega_d$.

We can easily extend this to any two charts whose 0d endpoints share a common 1d neighbor, σ_1, by applying algorithm 1. Each sequence can be related to a reference sequence with the same 0d cell, but which goes through σ_1 and then follows some arbitrarily chosen path, say $\mathfrak{u}$, to the top cell. If the same path is chosen to compare to both charts, then the relative orientation of the reference charts is -1.

Finally, this can be extended to any two charts terminating in arbitrarily separated 0d cells by iterating the previous step together with algorithm 1. We combine this into algorithm 2.

Algorithm 2

Input: two BCFW sequences of length $d = k(n-k)$: $\mathbf{w} = (a_1b_1)(a_2b_2)\ldots(a_db_d)$ and $\mathbf{w}' = (a'_1b'_1)(a'_2b'_2)\ldots(a'_db'_d)$; and their 0d endpoints: σ_0 and σ'_0.

Output: ±1

Procedure:

1) If $\sigma_0 = \sigma'_0$, compute the relative sign of the two forms using algorithm 1. Return the result.

[7] The only difference being that the merge/unmerge moves may be with other vertices instead of the boundary.

2) Else, let a be the smallest index such that $\sigma_0(a) < \sigma'_0(a)$, and let b be the smallest index such that $\sigma_0(b) > \sigma'_0(b)$. We assume that $a < b$; if not, then the roles of σ_0 and σ'_0 should be exchanged. Let $\tilde{\sigma} = \sigma_0 \cdot (ab)$, whose boundaries are σ_0 and $\tilde{\sigma}_0 = \tilde{\sigma} \cdot (b\ a{+}n)$, and define $\mathfrak{u}$ to be the Lemma 1 sequence from $\tilde{\sigma}$ to the top cell. Construct two reference sequences: $\tilde{\mathbf{w}} = (ab)\mathfrak{u}$ and $\tilde{\mathbf{w}}' = (b\ a{+}n)\mathfrak{u}$.

3) Repeat this algorithm to compare $(\mathbf{w}, \sigma_0)$ to $(\tilde{\mathbf{w}}, \sigma_0)$, and $(\mathbf{w}', \sigma'_0)$ to $(\tilde{\mathbf{w}}', \tilde{\sigma}_0)$.

4) Return the product of the results from step (3) times -1 due to the relative sign between $\tilde{\mathbf{w}}$ and $\tilde{\mathbf{w}}'$.

Proof. Assuming that the cells and edges in step (2) exist, the sign at each iteration is valid because it uses algorithm 1 to compare charts with identical 0d cells, and it returns -1 for each pair of sequences that differ only in the first position. It remains to show that step (2) is correct. Since all entries in the permutations labeling 0d cells are self-identified mod n, the definitions of a and b imply that:

$$\sigma_0(a) = a, \quad \sigma_0(b) = b{+}n, \quad \sigma'_0(a) = a{+}n, \quad \text{and} \quad \sigma'_0(b) = b. \tag{3.12}$$

There exists another 0d cell $\tilde{\sigma}_0$, which is identical to σ_0 except

$$\tilde{\sigma}_0(a) = \sigma_0(a){+}n = a{+}n = \sigma'_0(a) \quad \text{and} \quad \tilde{\sigma}_0(b) = \sigma_0(b){-}n = b = \sigma'_0(b). \tag{3.13}$$

The new cell $\tilde{\sigma}_0$ has two important properties:

- The first is that σ_0 and $\tilde{\sigma}_0$ have a common 1d neighbor, $\tilde{\sigma} = \sigma_0 \cdot (ab) = \tilde{\sigma}_0 \cdot (b\ a{+}n)$. Since all other entries in σ_0 and $\tilde{\sigma}_0$ are self-identified mod n, both (ab) and $(b\ a{+}n)$ are adjacent. Thus the cells and reference sequences of step (2) are uniquely defined and satisfy the standard adjacency requirements.
- The second is that $\tilde{\sigma}_0$ differs from σ'_0 at fewer sites than σ_0 differs. If there are $m \geq 1$ locations where $\sigma_0(i) - \sigma'_0(i) \neq 0$, then there are only $m - 2$ locations where $\tilde{\sigma}_0(i) - \sigma'_0(i) \neq 0$. Since all entries in 0d cells are self-identified mod n, and k entries are greater than n, then m must be even and no larger than $2k$. Hence, algorithm 2 will complete after at most k iterations.

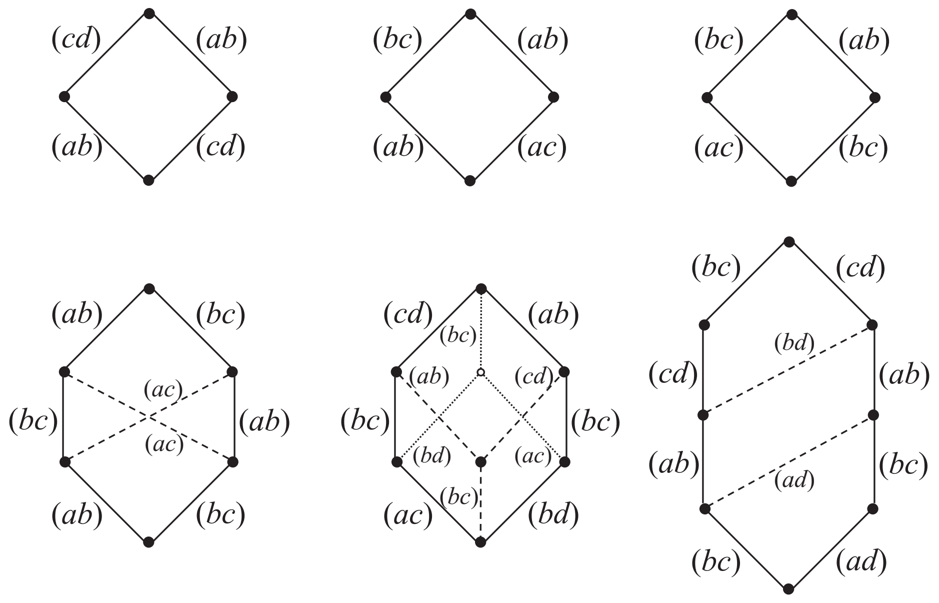
$\square$

Therefore algorithms 1 and 2 are together sufficient to find the relative orientation of any two standard BCFW charts.

3.2 Generalized decompositions

The plabic graph representation explained in section 2.3.1 is convenient for the study of standard BCFW charts. They are a subset of the generalized decompositions, so the rules for defining reference charts and relative signs in algorithms 1 and 2 will still apply. Since cases (i)-(vi) in step (2) of algorithm 1 cover all possible comparisons, the techniques introduced in the previous section are in fact sufficient to find the relative orientation of *any* two charts.

However, there is an advantage to studying the generalized charts more closely. Cases (iv)-(vi) in algorithm 1 were distinctly different than the other three cases because the reference charts required two or three steps to meet at a common cell as opposed to only one step in the earlier cases. Comparing the reference paths of the first three cases shows that they define quadrilaterals, while cases (iv) and (v) define hexagons, and (vi) defines an octagon. The relevant sections of the poset are depicted in (3.14) with the solid lines indicating the paths used in algorithm 1, and the dashed lines showing the additional edges that were not used (the finely dotted lines indicate edges that do not always exist). We include the quadrilaterals from cases (i)-(iii) for completeness:



(3.14)

The extra transpositions permitted in generalized charts allow the hexagons and octagons to be refined into quadrilaterals. Some of the internal quadrilaterals are equivalent to those from cases (i)-(iii), but there are also new ones. The relative orientation of two charts which differ only by one of the new quadrilaterals needs to be determined. To fix the signs, we require that the refined polygons produce the same signs as above when split into charts that differ by the interior quadrilaterals.

The hexagon from case (iv) can be split into a pair of quadrilaterals two ways. Either way, the top quadrilateral appears in one of the first three cases, which implies that the relative orientation around the lower quadrilateral must be -1 in order to agree with the overall sign of $+1$ derived in appendix A.

In case (v), the edges shown with dashed lines always exist, so the hexagon can be split into three quadrilaterals. The one on the lower left is equivalent to case (iii), the lower right is equivalent to case (ii), and the top quadrilateral is identical to case (i). Hence the product of the three individual signs is $(-1)^3 = -1$, in agreement with the result found in appendix A. In some situations, the hexagon can also be split up using the finely dotted lines. Then the top two hexagons are equivalent to cases (ii) and (iii), which implies that the relative orientation around the lower quadrilateral must also be -1.

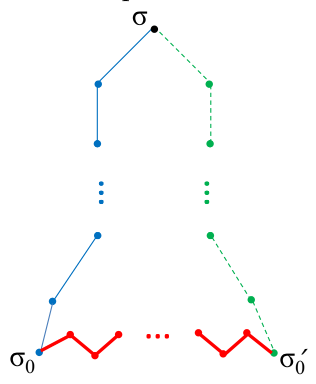
Finally, the octagon from case (vi) can be also be split into three quadrilaterals. The top one matches case (iii), and the middle is equivalent to case (ii), so the relative orientation around the lower one must be -1 to agree with appendix A. This is unsurprising considering that the two transpositions are completely disjoint, so applying them in opposite order would suggest that the forms differ by a minus signs, similar to example (1.5) in the introduction.

This exhausts all possible quadrilaterals that could appear in the poset. Hence the relative orientation between any two charts that differ by a quadrilateral is -1. A significant consequence of this result is the existence of a boundary operator which manifestly squares to zero, as we discuss further in section 4.

3.3 The master algorithm

Before proceeding to discuss various applications, we present a more efficient method to compute the relative orientation of any two charts. In each iteration of algorithm 1, every edge in the reference charts enters into two comparisons (once to the corresponding initial chart, and once to the other reference chart), while the edges in the initial charts enter only one comparison (to the associated reference chart). Therefore, if we assign ± 1 to each *edge* such that the product of signs around any quadrilateral is -1, then the signs on the reference chart edges will appear twice and hence square to 1, while the product of signs on the initial chart edges will combine to produce the same overall sign as found by algorithm 1. One method for producing a consistent set of edge signs is presented in appendix B [12].

When changing 0d cells with algorithm 2, one should think of taking a closed loop that traverses down and back each branch in (3.10), thus encountering each edge twice. Consequently, every edge sign appears twice, thus squaring to 1, so the only sign from this step will be -1 due to the relative sign between the reference charts. However, one can easily check that applying algorithm 1 will introduce one additional copy of each branch, so those edges should be included in the overall loop. The result of chaining several of these together is a sawtooth path between the two 0d cells (the bold red line in (3.15)) that contributes signs from each edge along the way and a minus sign for each 1d cell along the path. Schematically, the combined loop will look like the following:



(3.15)

This method for computing signs is summarized in the master algorithm:

Master Algorithm

Input: two BCFW sequences of length $d = k(n-k)$: $\mathbf{w} = (a_1b_1)(a_2b_2)\dots(a_db_d)$ and $\mathbf{w}' = (a'_1b'_1)(a'_2b'_2)\dots(a'_db'_d)$

Output: ± 1

Procedure:

1) If both BCFW paths terminate in the same 0-dimensional cell, then the relative orientation is given by the product of edge signs along the paths. Equivalently, it is the product of signs around the closed loop of length $2d$ obtained by traversing down one path and back up the other.

2) If they terminate in different 0d cells, then the relative orientation also depends on the signs along a path connecting the two 0d cells. One can obtain such a path by a sawtooth pattern between 0d and 1d cells. The sign of this path is given by the product of edge signs along the path, times $(-1)^{m/2}$, where m is the number of locations i satisfying $\sigma_0(i) - \sigma'_0(i) \neq 0$ ($m/2$ is the number of 1d cells in the sawtooth path). Thus the relative orientation is given by the product of signs along each BCFW path, times the connecting path sign. Equivalently, it is the product of signs around the closed loop obtained by concatenating the three paths (3.15), times the signs for the 1d cells.

So far, all the charts have been assumed to have minimal length, i.e. $k(n-k)$ for charts on the top cell. In other words, each transposition in the sequence increases the dimension by 1. However, the master algorithm indicates that *any* path through the poset can be compared to any other path, even if they zig-zag up and down.

We have verified this algorithm by implementing it in Mathematica and applying it to a variety of charts whose orientations are known by other methods. This includes pairs of randomly generated NMHV charts of the type studied in [11], as well as higher k charts whose matrix representatives have identical $GL(k)$ gauge fixings; the latter can be compared by directly equating the entries. In addition, the orientations of 500 distinct charts on the 10d cell $\{5,3,8,9,6,7,12,10,14,11\} \in \mathrm{Gr}(3,10)$ were computed using an independent method due to J. Bourjaily and A. Postnikov [14]. The results agreed perfectly with our algorithm.

4 Applications

In the remainder of this paper, we will discuss several areas in which the relative orientations are important. The end results are not surprising; they were anticipated and used in several previous works. The new contribution of this section will be to put these ideas on firm combinatorial footing, so for example, spurious poles in the tree contour will cancel exactly instead of mod 2 as in [1].

4.1 Comparing residue orientations

We have shown that any two charts can be compared by taking the product of edge signs around a closed loop in the poset of cells. This applies to any charts, even those corresponding to generalized decompositions with non-adjacent transpositions. As explained in section 2.3.2, one can follow any path through the poset by taking residues out of order in standard BCFW charts and changing coordinates as needed. Due to the master algorithm, the final sign on the residue depends only on the path, not on the intermediate choices of charts. We will demonstrate this with a convincing example.

There is a standard chart on $\{4, 3, 6, 5\}$ obtained by the path P_1:

$$\{4,3,6,5\} \xrightarrow{(23)} \{4,6,3,5\} \xrightarrow{(14)} \{5,6,3,4\}. \tag{4.1}$$

Both transpositions are adjacent. The corresponding matrix representative and form are

$$C = \begin{pmatrix} 1 & 0 & 0 & \alpha_1 \\ 0 & 1 & \alpha_2 & 0 \end{pmatrix} \qquad \omega = \mathrm{dlog}\,\alpha_2 \wedge \mathrm{dlog}\,\alpha_1. \tag{4.2}$$

Taking either coordinate to vanish lands in a codimension-1 boundary, so we are allowed to take them to zero in either order. There is also a non-adjacent path P_2 that ends in the same 0d cell:

$$\{4,3,6,5\} \xrightarrow{(14)} \{5,3,6,4\} \xrightarrow{(23)} \{5,6,3,4\}. \tag{4.3}$$

We will now evaluate the 0d residue along both paths using the coordinate chart (4.2), ignoring the delta functions in (1.1). The convention for evaluating residues is to take a contour around $\alpha_i = 0$ only when α_i is the first variable in the form. Along P_1 we first take $\alpha_2 \to 0$ and then $\alpha_1 \to 0$, which is the order they appear in the form; hence the residue is $+1$. We can follow P_2 by taking $\alpha_1 \to 0$ first, so we pick up a factor of -1 from reversing the order of the wedge product. Thus the residue along P_2 is -1. The relative sign is -1, exactly as our algorithm predicts because the paths differ by a quadrilateral.

In terms of edge signs, there is a $\mathbb{Z}_2$ symmetry at every vertex that allows us to flip the signs on all the attached edges without changing the overall sign of any closed loop. Since the product of signs around a quadrilateral is -1, we can use the symmetry to fix the edge signs so they exactly agree with the above computation at each step:

$$\begin{tikzcd} & \{4, 3, 6, 5\} \arrow[dl, dash, "+1"'] \arrow[dr, dash, "-1"] & \\ \{4, 6, 3, 5\} \arrow[dr, dash, "+1"'] & & \{5, 3, 6, 4\} \arrow[dl, dash, "+1"] \\ & \{5, 6, 3, 4\} & \end{tikzcd} \tag{4.4}$$

The generalization to more complicated charts is straightforward.

4.2 Boundary operator

Define the signed boundary operator ∂ acting on a cell C to be the sum of all boundaries of C weighted by the ± 1 weight on the edge connecting each boundary to C,

$$\partial C = \sum_i w(C, C'_i)\, C'_i, \tag{4.5}$$

where the sum is over all cells C'_i in the boundary of C and $w(C, C'_i) = \pm 1$ is the weight on the edge between C and C'_i. Equivalently, we could take the sum over all cells of the appropriate dimension and define $w(C, C'_i) = 0$ whenever there is no edge between them.[8] Applying the boundary operator again therefore yields a sum of codimension-2 boundaries of C, each one weighted by the product of the sign on the edge connecting it to its parent times the sign on its parent from the first application of ∂:

$$\partial^2 C = \sum_i \sum_{j(i)} w(C, C'_i)\, w\big(C'_i, C''_{j(i)}\big) C''_{j(i)}, \tag{4.6}$$

where i runs over boundaries C'_i of C and $j(i)$ runs over boundaries $C''_{j(i)}$ of C'_i. In order for this result to vanish, every codimension-2 cell must appear twice and with opposite signs.

We will first show that each cell appears twice,[9] and then it will be clear from our setup that that signs are opposite. Let σ be the permutation labeling C. Each edge represents a transposition acting on σ, so codimension-2 cells will arise from pairs of transpositions $(ab)(cd)$. If a, b, c, d are all distinct, then the transpositions can be applied in either order and thus each cell appears twice. If only three are distinct, then there are a few cases to consider:

$$\begin{array}{rl} (ac)(ab) \equiv (ab)(bc) & \sigma(a) < \sigma(c) < \sigma(b) \\ (ac)(bc) \equiv (bc)(ab) & \sigma(b) < \sigma(a) < \sigma(c) \\ (bc)(ac) \equiv (ab)(bc) \text{ and } (ab)(ac) \equiv (bc)(ab) & \sigma(a) < \sigma(b) < \sigma(c). \end{array} \tag{4.7}$$

Thus there are two unique routes from C to every codimension-2 cell in $\partial^2 C$. Each pair of routes defines a quadrilateral in the poset, which we have seen implies a relative minus sign between the residues. Hence the boundary operator manifestly squares to zero.

4.3 Locality of tree contours

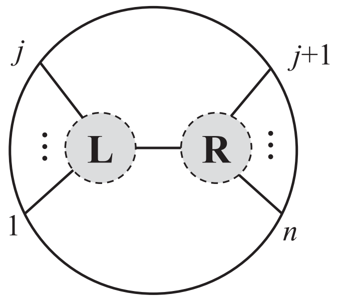
The n-particle N^{k}MHV tree amplitude can be computed as a linear combination of residues of (1.1) with coefficients ± 1. Each residue appearing in the amplitude corresponds to a $4k$-dimensional cell, whose the remaining degrees of freedom are fixed by the $4k$ bosonic delta functions in (1.1). A *tree contour* is defined as any choice of contour on the top cell that produces a valid representation of the tree amplitude; there are many equivalent representations due to residue theorems. Tree-level BCFW recursion relations [20–22] written in terms of on-shell diagrams [1] provide one technique to find an appropriate set

[8]This is not a unique definition since the edge weights can be flipped without affecting signs around closed loops, but any consistent set of signs such as those defined in appendix B will be sufficient for our purposes.

[9]See also section 6.3 of [1] for a similar proof that they appear twice.

of cells, but the on-shell diagram formulation does not generate the relative signs between them. This will be resolved shortly.

The tree amplitude diverges for certain configurations of the external momenta; these are called local poles — physically, they are interpreted as factorization channels in which an internal propagator goes on-shell such that the diagram splits into two on-shell subamplitudes **L** and **R**, e.g. all cyclic permutations of (4.8):



(4.8)

In the Grassmannian residue representation, such poles correspond to $(4k-1)$-dimensional cells, i.e. boundaries of the $4k$-dimensional cells. The amplitude is not manifestly local in this formulation, meaning that some boundary cells translate to non-local poles, which are momentum configurations with non-physical divergences. A key feature of the tree contour is that all of the local poles appear precisely once in the residue representation of the amplitude, while non-local poles appear twice [1]. It was conjectured that the two appearances of each non-local pole should come with opposite signs so they cancel in the sum, similar to the vanishing of ∂^2. We are now equipped to prove that claim.

The boundary operator (4.5) can be used to define signed residue theorems. Even though the sign of each term in (4.5) is not fixed, the *relative* sign between any two cells in the boundary, say C_1' and C_2', will always agree whenever they both appear in the boundary of a cell. It is easy to see that this is true if C_1' and C_2' share a common boundary since they form a quadrilateral in the poset. The edges connecting C_1' and C_2' to their shared boundary are the same no matter which C is used in the initial boundary operation, so the relative sign between the edges connecting C to C_1' and C_2' must not depend on that choice either. There is one situation in which they may not share a boundary, but in that case there will always be a third cell C_3' which has common boundaries with both of them; cf. case (v) in section 3.2. Now following the intuition of [1], we find residue theorems by requiring that the boundary of every $(4k+1)$-dimensional cell vanishes:

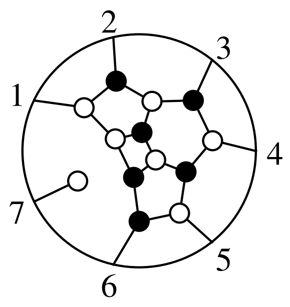
$$\partial C^{(4k+1)} = \sum_i w\left(C^{(4k+1)}, C_i^{(4k)}\right) C_i^{(4k)} = 0. \tag{4.9}$$

We define the tree contour to encircle each enclosed singularity of the measure exactly once, so any residue appearing in the amplitude will have a coefficient ± 1. The residue theorems (4.9) can change which poles are included in the contour, but they will never cause residues to appear more than once. This implies that the relative sign between any two cells in the amplitude must match the relative sign of those cells in the residue theorems. Therefore, by the same logic that showed $\partial^2 = 0$, it follows that any $(4k-1)$-dimensional cell appearing twice in the boundary of the tree amplitude will show up with opposite signs. Hence every doubly-appearing pole cancels in the sum.

Note that we have presented these results in terms of the momentum twistor representation in which the amplitude is a linear combination of $4k$-dimensional cells. The method works analogously in the twistor version where the amplitude is described by $(2n-4)$-dimensional cells. We have checked numerically that this choice of signs correctly cancels all paired poles for BCFW representations of the tree amplitude with $n = 5, \ldots, 13$ and $k = 3, \ldots, \lfloor n/2 \rfloor$.

4.4 Other non-physical poles

After canceling all pairs of poles as described above, all remaining terms in the boundary sum will have unit multiplicity as expected for local poles. However, some of the surviving boundaries may yet correspond to non-physical divergences. For example, one representation of the 7-particle N^{2}MHV amplitude (accessible by `treeContour`[7, 4] in the *positroids* package [15]) has 52 terms in the boundary. Eight of them appear twice and hence cancel in the sum leaving 36 unit multiplicity poles. One of those poles is labeled by the permutation $\{4, 5, 6, 8, 9, 10, 14\}$ and can be represented graphically by the following diagram:

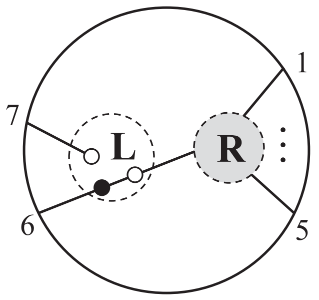


(4.10)

This diagram does not indicate any internal propagators going on-shell, so it does not represent a physical singularity of the form (4.8).

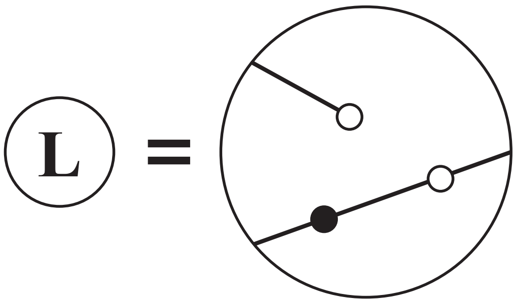
Since the boundary of the amplitude is ultimately expected to contain all (and only) physical poles, it is therefore not surprising that some Grassmannian representations of the amplitude do not contain a pole of this form. However, we must then explain how such a diagram can apparently contribute to the boundary of *any* representations. In fact, we will see that momentum conservation forces this diagram to vanish.

To illustrate the result, let us group legs 6 and 7 together in (4.10) so that it looks schematically like (4.8):



(4.11)

We have added two bivalent vertices on leg 6 and grouped those vertices with leg 7 in the subamplitude **L**. The other subamplitude **R** contains all other vertices from (4.10). No extra internal propagators have gone on-shell since leg 6 was already an on-shell external propagator and adding the two bivalent vertices does not affect the momentum flowing through them. Recall that momentum conservation requires both **L** and **R** to be on-shell subamplitudes in (4.8), but in (4.11) the left-hand subamplitude,



(4.12)

does not have support for generic "external" momenta and therefore vanishes. Consequently, the whole diagram (4.10) is identically zero and can be neglected in the boundary sum whenever it appears.

The example (4.11) is a special case of a much more general phenomenon wherein an apparently valid boundary term with unit multiplicity actually vanishes after taking into account momentum conservation on the left and right subamplitudes independently. Any boundary labeled by a permutation with a self-identified element will have a lollipop in its diagram and thus vanish by the same logic as above. In addition, for Grassmannian amplitude representations with large enough k and n there will be many lollipop-free boundary diagrams that factorize in the form (4.8) yet still fail to satisfy momentum conservation on the left or right. Those diagrams evaluate to zero due to the momentum delta functions and can be dropped from the sum. Given an arbitrary diagram of the form (4.8), it is straightforward to check whether the left and right subamplitudes have kinematical support by computing their respective *intersection numbers* as defined in section 10 of [1]. If either intersection number is zero, then the whole diagram vanishes and can be dropped from the boundary sum.[10]

After removing from the boundary of the amplitude any cells with an unsupported subamplitude, the remaining terms will all correspond to physical, local poles of the form (4.8) with non-vanishing residues. Note that there exist multiple equivalent representations of amplitudes, so the subamplitudes that appear in the boundaries of different representations of the full amplitude may not be manifestly identical. Residue theorems ensure that as long as the boundary cells can be grouped into subamplitudes as in (4.8), then every amplitude representation will have the same *number* of supported boundaries in every channel of (4.8). We have checked for multiple Grassmannian BCFW representations of n-particle

[10]This is the simplest method to check the kinematical support for subamplitudes with at least four particles. A 3-particle subamplitude is either equivalent to a single black or white 3-vertex, in which case it is non-vanishing due to special 3-particle kinematics, or it is zero as in (4.12).

N^{k}MHV amplitudes with $n = 5, \ldots, 13$ and $k = 3, \ldots, \lfloor n/2 \rfloor$ that the above method for removing unsupported poles produces boundaries with matching numbers of terms in every factorization channel.

The same principles must be valid for each of the subamplitudes as well, namely they must be local on-shell amplitudes. Therefore, given an arbitrary list of $4k$-dimensional cells,[11] one can determine whether that list represents a physical amplitude by recursively checking that the number of supported boundaries of each subamplitude matches the number of boundaries of a known local BCFW bridge representation of the same amplitude. However, this is not a very efficient solution in practice. We expect there should be a more straightforward method to test whether a given set of cells forms an amplitude directly at the level of those cells and their boundaries. We leave this as an open question for the reader to explore.

Acknowledgments

A great deal of gratitude is owed to Rachel Karpman for many helpful discussions and clarification of several important concepts throughout the course of this project. I am grateful also to Thomas Lam for sharing the edge-weighting algorithm of appendix B, and to Jake Bourjaily for sharing his Mathematica code and for related discussions. I would like to thank Henriette Elvang, Yu-tin Huang, Cindy Keeler, Samuel Roland, and David Speyer for many helpful conversations along the way. Thanks also to Emily Olson for providing feedback and comments on the paper. This material is based upon work supported by the National Science Foundation Graduate Research Fellowship under Grant No. F031543.

A Details for algorithm 1

In this appendix, we derive the reference sequences and compute the relative signs for step (2) of algorithm 1. Many of the transformations presented here were computed independently by R. Karpman who found agreement with these results [13].

i) (ab) vs. (cd)
This case was discussed in the main text so we do not repeat the argument here.

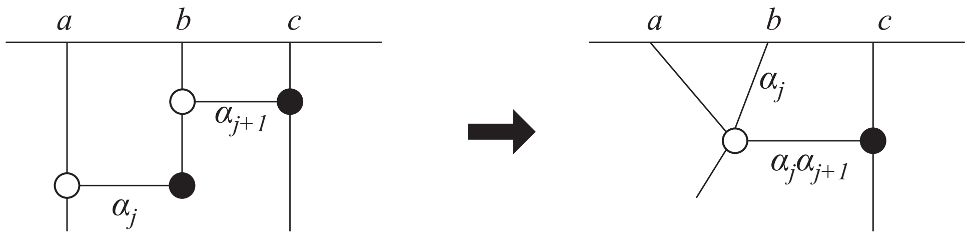
ii) (ab) vs. (ac)
These transpositions share a leg, so they do not commute as simply as in the previous case. Nonetheless, in this case we can use the fact that both (ab) and (ac) are allowed transpositions on the initial permutation to find a common parent cell. Specifically, since (ab) was the last transposition before σ, we know $a < b \leq \sigma(a) < \sigma(b) \leq a+n$ and there is no $q \in (a, b)$ such that $\sigma(q) \in (\sigma(a), \sigma(b))$. Similarly from (ac) we know $a < c \leq \sigma'(a) < \sigma'(c) \leq a+n$ and there is no $q \in (a, c)$ such that $\sigma'(q) \in (\sigma'(a), \sigma'(c))$. Moreover, since σ and σ' come from the same initial permutation, we also know that $\sigma(a) = \sigma'(b)$, $\sigma(b) = \sigma'(c)$, $\sigma(c) = \sigma'(a)$, and $\sigma(q) = \sigma'(q)$ for all other legs.

[11]Or $(2n-4)$-dimensional cells in momentum space.

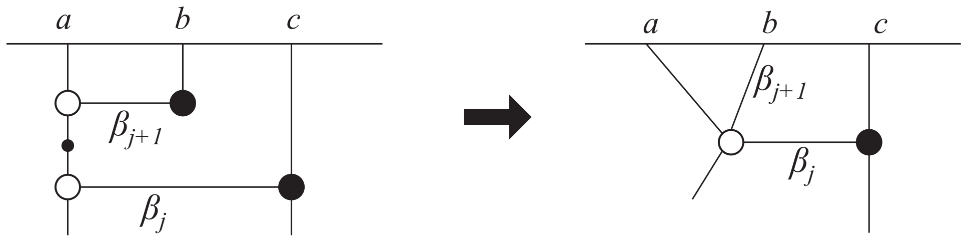
To show that (bc) can be applied to σ, we need to show that $\sigma(c) < \sigma(b)$ and that there is no $q \in (b,c)$ such that $\sigma(q) \in (\sigma(c), \sigma(b))$. The condition that $\sigma(c) < \sigma(b)$ is satisfied since it is equivalent to $\sigma'(a) < \sigma'(c)$, and since no legs are touched between b and c, the condition on $q \in (a,c)$ implies that no $q \in (b,c)$ has $\sigma(q) \in (\sigma(c), \sigma(b))$. Thus (bc) is an allowed transposition on σ, so the path $\tilde{\mathbf{w}}$ exists.

We also need to show that (ab) can be applied to σ', so we must show $\sigma'(b) < \sigma'(a)$ and that there is no $q \in (a,b)$ such that $\sigma'(q) \in (\sigma'(b), \sigma'(a))$. Since $b \in (a,c)$, we must have either $\sigma'(b) < \sigma'(a) < \sigma'(c)$ or $\sigma'(a) < \sigma'(c) < \sigma'(b)$. Since $\sigma(a) < \sigma(b) \Rightarrow \sigma'(b) < \sigma'(c)$, only the former condition is allowed, hence $\sigma'(b) < \sigma'(a)$. Furthermore, there is no $q \in (a,b)$ such that $\sigma'(q) \in (\sigma'(b), \sigma'(c))$, and $(\sigma'(b), \sigma'(a))$ is a subset of that range. Hence (ab) is allowed on σ', and $\tilde{\mathbf{w}}'$ exists.

Note that the above analysis was valid for any charts, not just adjacent ones. However, to compute the relative sign, we will focus on the restricted set of charts for which all $q \in (a,c)$ satisfy $\sigma'(q) \equiv q \bmod n$, namely the standard BCFW charts. The corresponding plabic graphs can be manipulated to a common layout using the equivalence moves (2.7) and (2.8):



(A.1)



(A.2)

We find that they are equivalent only with the identifications

$$\beta_j = \alpha_j \alpha_{j+1}, \qquad \beta_{j+1} = \alpha_j. \tag{A.3}$$

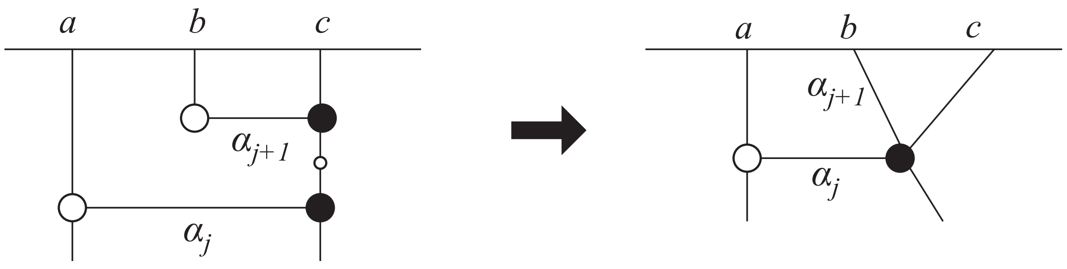
Plugging this into the dlog forms and using that

$$\mathrm{dlog}\,\alpha_j \wedge \mathrm{dlog}\,(\alpha_j\alpha_{j+1}) = \mathrm{dlog}\,\alpha_j \wedge \Big(\mathrm{dlog}\,\alpha_j + \mathrm{dlog}\,\alpha_{j+1}\Big) = -\mathrm{dlog}\,\alpha_{j+1} \wedge \mathrm{dlog}\,\alpha_j, \tag{A.4}$$

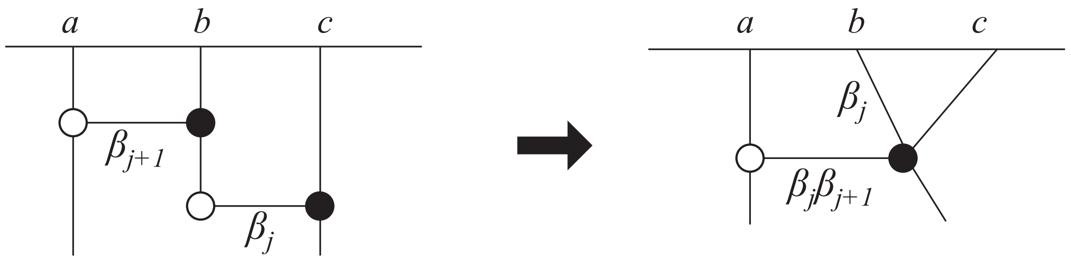
we find that the two forms are oppositely oriented.

iii) (ac) vs. (bc)

This situation is analogous to case (ii), so we can skip directly to comparing the graphs. Restricting again to the standard BCFW situation where $\sigma(q) \equiv q \bmod n$ for all $q \in (a, c)$, we can perform a sequence of merge/delete and $GL(1)$ rotations on the corresponding plabic graphs:



(A.5)



(A.6)

We therefore identify

$$\beta_j = \alpha_{j+1}, \qquad \beta_{j+1} = \alpha_j/\alpha_{j+1}. \tag{A.7}$$

Then using that $\mathrm{dlog}\,(\alpha_j/\alpha_{j+1}) = \mathrm{dlog}\,\alpha_j - \mathrm{dlog}\,\alpha_{j+1}$, we find that the two forms are oppositely oriented.

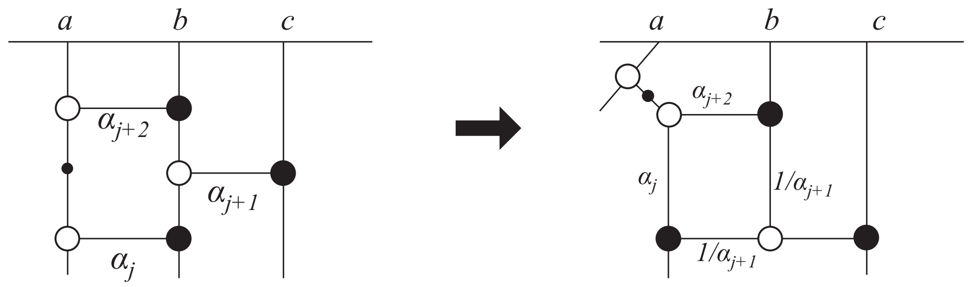
iv) (ab) vs (bc)

Using only adjacent transpositions, σ and σ' do not have a common parent cell. However, we can show that $\rho = \sigma \cdot (bc)(ab) = \sigma' \cdot (ab)(bc)$ is a shared grandparent. From the initial (ab), we know $a < b \leq \sigma(a) < \sigma(b) \leq a+n$, while from the initial (bc), we have $b < c \leq \sigma'(b) < \sigma'(c) \leq b+n$. From their shared origin cell, we also have $\sigma(a) = \sigma'(c)$, $\sigma(b) = \sigma'(a)$, and $\sigma(c) = \sigma'(b)$. We will focus on adjacent charts wherein $\sigma(q) = \sigma'(q) \equiv q \bmod n$ for all $q \in (a, b) \cup (b, c)$. The general case is covered in section 3.2.

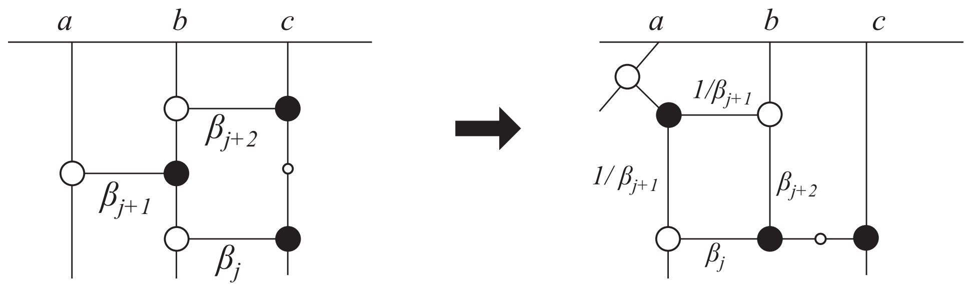
Since $\sigma'(b) < \sigma'(c)$ is equivalent to $\sigma(c) < \sigma(a)$, and $\sigma(a) < \sigma(b)$, we can apply (bc) to reach $\tilde{\sigma} = \sigma \cdot (bc)$. Now $\tilde{\sigma}(b) < \tilde{\sigma}(a) < \tilde{\sigma}(c)$, so (ab) is a valid transposition, which arrives at ρ. Hence the reference chart $\tilde{\mathbf{w}}$ exists.

On the other side, $\sigma(a) < \sigma(b)$ is equivalent to $\sigma'(c) < \sigma'(a)$, and $\sigma'(b) < \sigma'(c)$, so (ab) can be applied to σ', yielding $\tilde{\sigma}' = \sigma' \cdot (ab)$. Since $\tilde{\sigma}'(a) < \tilde{\sigma}'(c) < \tilde{\sigma}'(b)$, we can apply (bc), which also arrives at ρ. Thus $\tilde{\mathbf{w}}'$ is also a valid reference chart.

Finally, we can compare the plabic graphs to find their relative orientation:



(A.8)



(A.9)

After performing a square move (2.9) on (A.9), we find

$$\beta_j = \frac{\alpha_{j+1}\alpha_{j+2}}{\alpha_j + \alpha_{j+2}}, \quad \beta_{j+1} = \alpha_j + \alpha_{j+2}, \quad \beta_{j+2} = \frac{\alpha_j\alpha_{j+1}}{\alpha_j + \alpha_{j+2}}. \tag{A.10}$$

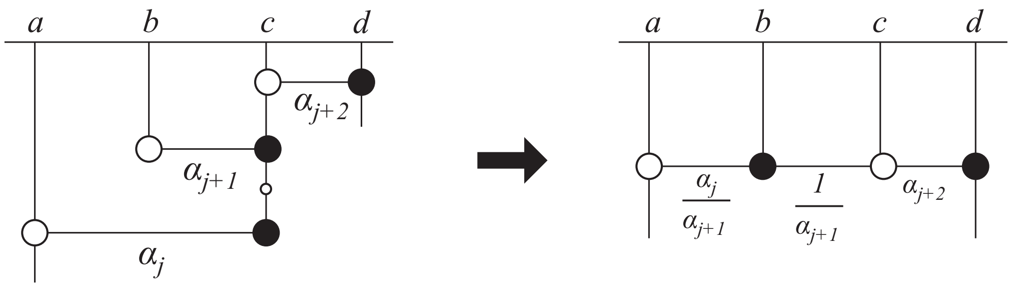
Chugging through a bit of algebra, the result is a positive relative orientation.

v) **(ac) vs. (bd)**
There is no way to find a common parent using only adjacent transpositions. However, there is a common grandparent $\rho = \sigma \cdot (bc)(cd) = \sigma' \cdot (bc)(ab)$. Since σ and σ' come from a shared origin, we have the following: $a < c \leq \sigma(a) < \sigma(c) \leq a{+}n$ and $b < d \leq \sigma'(b) < \sigma'(d) \leq b{+}n$; and $\sigma(a) = \sigma'(c)$, $\sigma(b) = \sigma'(d)$, $\sigma(c) = \sigma'(a)$, and $\sigma(d) = \sigma'(b)$. We focus on adjacent charts, so $\sigma(q) = \sigma'(q) \equiv q \bmod n$ for all $q \in (a,d)\backslash\{b,c\}$. In addition, $\sigma(b) = \sigma'(d) = b{+}n$ and $\sigma'(c) = \sigma(a) = c$.

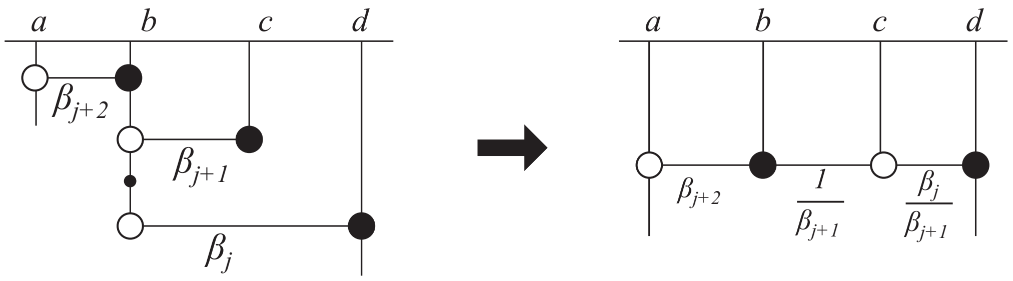
Since $\sigma(b) = b{+}n > a{+}n \geq \sigma(c)$, the transposition (bc) is allowed, which leads to $\tilde{\sigma} = \sigma \cdot (bc)$. Then $\tilde{\sigma}(a) < \tilde{\sigma}(b) < \tilde{\sigma}(c) = b{+}n$ and $\tilde{\sigma}(d) = \sigma(d) = \sigma'(b) < b{+}n$. Therefore (cd) is also allowed, and we arrive at ρ. Therefore $\tilde{\mathbf{w}}$ is a valid reference chart.

For σ', we use that $\sigma'(c) = c < d \leq \sigma'(b)$, so (bc) is allowed. Thus with $\tilde{\sigma}' = \sigma' \cdot (bc)$, we have that $c = \tilde{\sigma}'(b) < \tilde{\sigma}'(c) < ts'(d)$ and $\tilde{\sigma}'(a) = \sigma'(a) = \sigma(c) > c$. Hence we can apply (ab), which yields ρ, so $\tilde{\mathbf{w}}'$ is a good reference chart.

To compute the relative sign, we study the corresponding plabic graphs:



(A.11)



(A.12)

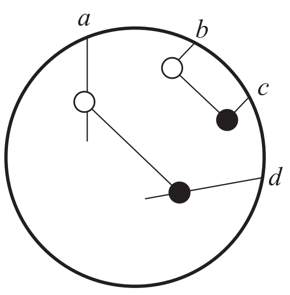
They are equivalent under the identifications

$$\beta_j = \alpha_{j+1}\alpha_{j+2}, \quad \beta_{j+1} = \alpha_{j+1}, \quad \beta_{j+2} = \alpha_j/\alpha_{j+1}. \tag{A.13}$$

Plugging this into ω', one finds that the two forms are oppositely oriented.

vi) (bc) vs. (ad)

From the point of view of a graph embedded in a disk, these bridges can be added in either order:



(A.14)

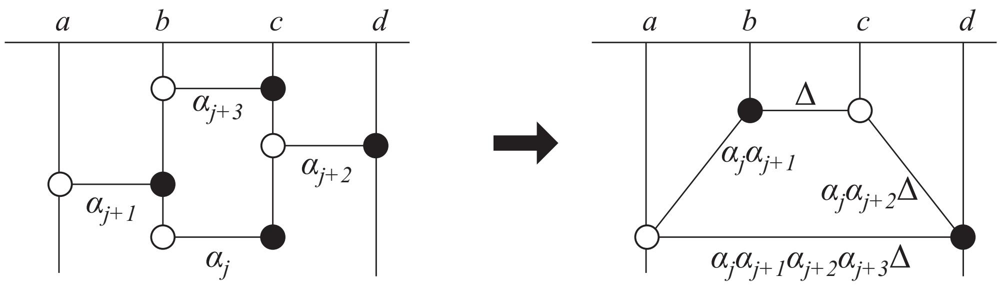
However, the adjacency requirement forbids applying (ad) after (bc). Thus we must look further to find a meeting point, $\tilde{\rho}$, for their reference charts. We will focus on adjacent charts, leaving the general case for section 3.2.

Since $b, c \in (a, d)$, we know $\sigma(b), \sigma(c) \equiv c, b \bmod n$, so it follows that $\sigma(b) = c$ and $\sigma(c) = b+n$. Then since $\sigma(a) = \sigma'(d) \geq d > c = \sigma(b)$, we can next apply (ab).

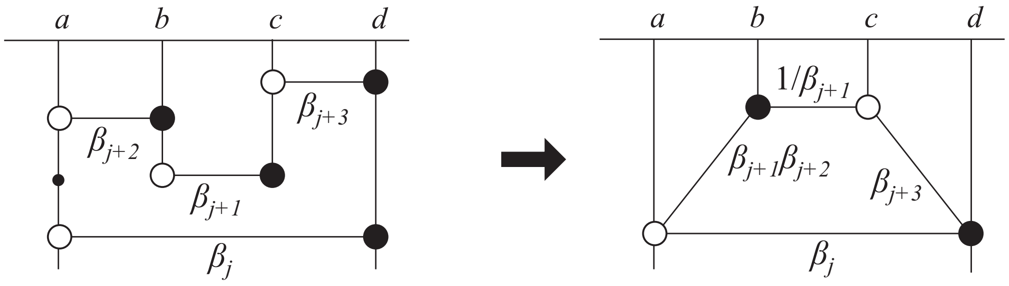
Similarly, $\sigma(d) = \sigma'(a) \leq a + n < b + n = \sigma(c)$, we could also apply (cd); this is unaffected by applying (ab), so we apply it next to arrive at $\rho = \sigma \cdot (ab)(cd)$. Finally, (bc) is allowed because $\rho(b) = \sigma(a) = \sigma'(d) > \sigma'(a) = \sigma(d) = \rho(c)$ and $c < d \leq \sigma'(a) < \sigma'(d) \leq a+n < b+n$. Hence $\tilde{\mathbf{w}}$ is a valid reference chart.

From σ', we are certainly allowed to apply (bc) after (ad), thus arriving at $\tilde{\sigma} = \sigma \cdot (bc)$. We can add (ab) and then (cd) by essentially the same argument as above. Therefore both $\tilde{\mathbf{w}}$ and $\tilde{\mathbf{w}}'$ are valid reference charts.

We compare the plabic graphs to find the relative sign. One can either use the equivalence moves (2.7)–(2.9) or repeatedly apply the transformation rules from cases (ii)-(iv) to find



(A.15)



(A.16)

where we have defined $\Delta = 1\big/(\alpha_j + \alpha_{j+3})$. The graphs are equivalent after the following identifications:

$$\beta_j = \frac{\alpha_j\alpha_{j+1}\alpha_{j+2}\alpha_{j+3}}{\alpha_j + \alpha_{j+3}}, \quad \beta_{j+1} = \alpha_j + \alpha_{j+3}, \quad \beta_{j+2} = \frac{\alpha_j\alpha_{j+1}}{\alpha_j + \alpha_{j+3}}, \quad \beta_{j+3} = \frac{\alpha_j\alpha_{j+2}}{\alpha_j + \alpha_{j+3}}. \tag{A.17}$$

Rearranging the corresponding forms demonstrates that the relative orientation of the reference charts is -1.

B Assigning edge weights

In this appendix we provide one technique for assigning weights ± 1 to every edge in the poset such that the product of signs around every quadrilateral is -1. The method was developed by T. Lam and D. Speyer [12].

B.1 Reduced words

Each decorated permutation can equivalently be represented by one or more *reduced words*: minimal-length sequences of *letters* which obey certain equivalence relations. In this case, the letters s_i are generators of the affine permutation group $\tilde{S}_n$. They are defined with the following properties and relations:

$$\begin{gathered} s_i := \begin{cases} \sigma(i) \to \sigma(i+1) \\ \sigma(i+1) \to \sigma(i) \end{cases} 1 \leq i \leq n, \qquad s_{i+n} \equiv s_i, \\ s_i^2 \equiv 1, \qquad s_i s_{i+1} s_i \equiv s_{i+1} s_i s_{i+1}, \qquad s_i s_j \equiv s_j s_i\,, \ |i-j| > 1. \end{gathered} \tag{B.1}$$

We will refer to the relations in the second line as, respectively, the *reduction move*, the *braid move*, and the *swap move*. As we showed in Lemma 1, any d-dimensional cell C in $\mathrm{Gr}(k,n)$ can be reached from the top cell by a sequence of $k(n-k)-d$ transpositions s_i, and that sequence defines a reduced word labeling C. There are generally many distinct reduced words for a given cell, but they are equivalent due to the relations in (B.1).

Any two cells C and C' of dimensions d and $d+1$, which share an edge in the poset, i.e. are related by a single boundary operation, are straightforwardly connected in the language of reduced words. Given a reduced word f of length $\delta = k(n-k)-d$ on the lower-dimensional cell C, there is a unique reduced word f' on C' of length $\delta' = \delta - 1$ obtained by deleting a single letter from f [23]. Specifically, for $f = s_{i_1} s_{i_2} \dots s_{i_j} \dots s_{i_\delta}$, there is a unique s_{i_j} such that $f' = s_{i_1} s_{i_2} \dots \hat{s}_{i_j} \dots s_{i_\delta}$, where the hat denotes deletion. It is easy to construct a (generally non-reduced) word t such that $f' \equiv tf$; one can check that $t = s_{i_1} s_{i_2} \dots s_{i_{j-1}} s_{i_j} s_{i_{j-1}} \dots s_{i_2} s_{i_1}$ accomplishes the desired effect by repeated reduction moves. Given that the two cells are also related by a transposition of the form (ab), it is not surprising that repeated application of the relations (B.1) shows that $t \equiv s_a s_{a+1} \dots s_{b-2} s_{b-1} s_{b-2} \dots s_{a+1} s_a$, which is a reduced word representation of (ab).

B.2 Decorating edges

Choose a representative reduced word for every cell in the poset; we will refer to this as the *standard word* on that cell. This is similar to choosing a particular Lemma 1 sequence for each cell. From a cell C labeled by the standard word f, every cell C' that has C as a boundary can be reached by deleting some s_{i_j} from f. This yields a word f' on C' as explained above. If f' is identical to the standard word on C', then weight the corresponding edge with $(-1)^j$. It is easy to check that deleting two letters $s_{i_{j_1}}$ and $s_{i_{j_2}}$ in opposite orders will produce the desired factor of -1 around a quadrilateral. For one order, we would find $(-1)^{i_{j_1}+i_{j_2}}$, while for the other order we would find $(-1)^{i_{j_1}+i_{j_2}-1}$, so they differ by -1.

However, it is not always possible to delete both transpositions in either order. One may have to delete different transpositions to reach the same cell by two different routes. Moreover, the final reduced word in each case may be different. Thus, we need to find the sign difference between two reduced words. Any two reduced words can be mutated into each other using just the braid and swap moves defined in (B.1). By decorating these rules with ± 1, we can find the desired difference between the words. In fact, we already determined those signs in appendix A because the generators act as adjacent transpositions.

The braid move is simply a special case of (iv), so it should be decorated with a $+1$, and the swap move is a special case of (i), so it should be decorated with a -1. Hence, the relative sign between two words is the number of swaps required to transform one into the other.

The number of swaps can be determined directly from the *inversion list* of each word. A reduced word creates a unique ordering on the set of inversions in the permutation labeling the cell (not all orderings are possible). The number of swap moves needed to transform one word into another is the number of pairs of inversions (i, j) and (k, l), with all i, j, k, l distinct, in different order in the two inversion lists.

References

[1] N. Arkani-Hamed et al., *Scattering amplitudes and the positive Grassmannian*, arXiv:1212.5605 [INSPIRE].

[2] S. Franco, D. Galloni and A. Mariotti, *The geometry of on-shell diagrams*, *JHEP* **08** (2014) 038 [arXiv:1310.3820] [INSPIRE].

[3] Y. Bai and S. He, *The amplituhedron from momentum twistor diagrams*, *JHEP* **02** (2015) 065 [arXiv:1408.2459] [INSPIRE].

[4] N. Arkani-Hamed and J. Trnka, *The amplituhedron*, *JHEP* **10** (2014) 030 [arXiv:1312.2007] [INSPIRE].

[5] N. Arkani-Hamed and J. Trnka, *Into the amplituhedron*, *JHEP* **12** (2014) 182 [arXiv:1312.7878] [INSPIRE].

[6] S. Franco, D. Galloni, A. Mariotti and J. Trnka, *Anatomy of the amplituhedron*, *JHEP* **03** (2015) 128 [arXiv:1408.3410] [INSPIRE].

[7] O. Aharony, O. Bergman, D.L. Jafferis and J. Maldacena, *$N = 6$ superconformal Chern-Simons-matter theories, $M2$-branes and their gravity duals*, *JHEP* **10** (2008) 091 [arXiv:0806.1218] [INSPIRE].

[8] K. Hosomichi, K.-M. Lee, S. Lee, S. Lee and J. Park, *$N = 5, 6$ superconformal Chern-Simons theories and $M2$-branes on orbifolds*, *JHEP* **09** (2008) 002 [arXiv:0806.4977] [INSPIRE].

[9] Y.-T. Huang and C. Wen, *ABJM amplitudes and the positive orthogonal Grassmannian*, *JHEP* **02** (2014) 104 [arXiv:1309.3252] [INSPIRE].

[10] Y.-T. Huang, C. Wen and D. Xie, *The positive orthogonal Grassmannian and loop amplitudes of ABJM*, *J. Phys.* **A 47** (2014) 474008 [arXiv:1402.1479] [INSPIRE].

[11] H. Elvang et al., *Grassmannians for scattering amplitudes in 4d $N = 4$ SYM and 3d ABJM*, *JHEP* **12** (2014) 181 [arXiv:1410.0621] [INSPIRE].

[12] T. Lam and D. Speyer, private communication.

[13] R. Karpman, private communication.

[14] J. Bourjaily, private communication.

[15] J.L. Bourjaily, *Positroids, plabic graphs and scattering amplitudes in mathematica*, arXiv:1212.6974 [INSPIRE].

[16] A. Knutson, T. Lam and D. Speyer, *Positroid varieties: juggling and geometry*, arXiv:1111.3660.

[17] T. Lam, *Notes on the totally nonnegative Grassmannian*, unpublished notes, (2013).

[18] A. Postnikov, *Total positivity, Grassmannians and networks*, math/0609764 [INSPIRE].

[19] R. Karpman, *Bridge graphs and Deodhar parametrizations for positroid varieties*, arXiv:1411.2997.

[20] R. Britto, F. Cachazo and B. Feng, *New recursion relations for tree amplitudes of gluons*, *Nucl. Phys.* **B 715** (2005) 499 [hep-th/0412308] [INSPIRE].

[21] R. Britto, F. Cachazo, B. Feng and E. Witten, *Direct proof of tree-level recursion relation in Yang-Mills theory*, *Phys. Rev. Lett.* **94** (2005) 181602 [hep-th/0501052] [INSPIRE].

[22] A. Brandhuber, P. Heslop and G. Travaglini, *A note on dual superconformal symmetry of the $N = 4$ super Yang-Mills S-matrix*, *Phys. Rev.* **D 78** (2008) 125005 [arXiv:0807.4097] [INSPIRE].

[23] A. Björner and F. Brenti, *Combinatorics of Coxeter groups*, Graduate Texts in Mathematics **231**, Springer, Germany (2005) [ISBN:9783540442387].