Multivariate reconstruction techniques over finite fields for scattering amplitudes

Tiziano Peraro

Higgs Centre for Theoretical Physics
University of Edinburgh

July 11, 2017
Amplitudes, Edinburgh

Outline

1. Introduction and motivation
2. Finite fields and multivariate reconstruction
3. Applications to scattering amplitudes
4. Summary & Outlook

Introduction and motivation

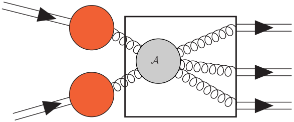
Loop amplitudes at high multiplicity

Phenomenological predictions

- Experiments at LHC
  - high-accuracy (up to % level in Run II)
  - large SM background
  - high c.o.m. energy ⇒ multi-particle states
- We need scattering amplitudes with
  - high accuracy ⇒ loops
  - multi-particle ⇒ high multiplicity

Theoretical studies of amplitudes

- infer general structures in QFT and gauge theories
- exploit them in computational techniques



Loop amplitudes at high multiplicity

- Loop amplitudes can be written as linear combinations of integrals

$$\mathcal{A}^{(\ell)} = \sum_i c_i \, I_i$$

- the integrals I_i are special functions of the kinematic invariants
  - at one-loop only logarithms and dilogarithms
  - at higher loops multiple polylogarithms, elliptic functions, etc. . .
- the coefficients c_i are rational functions of kinematic invariants
  - they are often a bottleneck at high multiplicity

In this talk I will mostly focus on the coefficients

Computing amplitudes: analytic vs numerical

QCD and SM amplitudes:

- Tree-level/One loop $\to$ mostly numerical
  - many automated codes and toolchains
  - essentially a solved problem for any process/theory/multiplicity
  - focus is on performance and stability
- Higher loops $\to$ mostly analytic
  - more efficient/stable numerical evaluation
  - more convenient for some techniques (e.g. IBPs, diff. eqs.)
  - allows many checks/manipulations/studies (singularities, limits, ...)
  - can be used to infer general analytic/algebraic properties

  $\Rightarrow$ more control

- note that numerical algorithms (e.g. at 1 loop) often rely on good understanding of analytic/algebraic properties of the result

Analytic calculation of scattering amplitudes

- Main bottleneck: large intermediate expressions
  - they can be orders of magnitude larger than the final result
  - not constrained by properties and symmetries of the result
- Tools for mitigating the problem
  - computer algebra systems specialised in handling large expressions (e.g. FORM [Vermaseren et al.])
  - generalized unitarity $\Rightarrow$ intermediate steps are gauge invariant

The main idea of this talk

- reconstruct analytic result from “numerical” evaluations
- no large intermediate expression (just numbers!)

Reconstruction of rational functions

- Which kind of "numerical" evaluation is good?
  - floating-point evaluation
    - ✓ very fast
    - ✗ affected by numerical instabilities
  - evaluation over the rational field $\mathcal{Q}$
    - ✓ exact
    - ✗ intermediate results have large numerators/denominators
      $\Rightarrow$ requires slow multi-precision arithmetic
  - evaluation over finite-fields
    - ✓ a finite-number of elements, which can be represented by machine-size integers
    - ✓ fast
    - ✓ exact
    - ✗ some information is lost and must be recovered by repeating the reconstruction over several finite fields

Functional reconstruction over finite fields

- Finite fields
  - used under-the-hood by computer algebra systems (e.g. in polynomial factorization/GCD)
  - used for IBPs (univariate applications)
    [von Manteuffel, Schabinger (2014–2017)]

- Efficient algorithm for functional reconstruction [T.P. (2016)]
  - works on (dense) multivariate polynomials and rational functions
  - implemented in C++ code (proof of concept)
  - the input is a numerical procedure computing a function
  - the output is its analytic expression
- Applications
  - linear systems of equations and composite functions
  - spinor-helicity and tree-level recursion
  - multi-loop integrand reduction and generalized unitarity

Finite fields and multivariate reconstruction

Finite fields

- In this talk we consider finite fields $\mathcal{Z}_p$, with p prime
- We define

$$\mathcal{Z}_n = \{0, \ldots, n-1\}$$

  - addition, subtraction, and multiplication via modular arithmetic

$$4 + 5\Big|_{\mathcal{Z}_7} = (4+5) \text{ mod } 7 = 2$$

  - if $a \in \mathcal{Z}_n$ and a, n are coprime, we can define an inverse

$$b = a^{-1} \in \mathcal{Z}_n, \qquad a \times b \text{ mod } n = 1$$

  - if $n = p$ prime, an inverse exists for every $a \in \mathcal{Z}_p \Rightarrow \mathcal{Z}_p$ is a field
  - every rational operation is well defined in $\mathcal{Z}_p$

Polynomials and rational functions

- multi-index notation: variables $\mathbf{z} = (z_1, \ldots, z_n)$ and integer list $\alpha = (\alpha_1, \ldots, \alpha_n)$

$$\mathbf{z}^\alpha \equiv \prod_{i=1}^{n} z_i^{\alpha_i}, \qquad |\alpha| = \sum_i \alpha_i$$

- Given a generic field $\mathcal{F}$
  - $\mathcal{F}[\mathbf{z}]$ is the ring of polynomials in $\mathbf{z}$ with coefficients in $\mathcal{F}$

$$f(\mathbf{z}) = \sum_\alpha c_\alpha \, \mathbf{z}^\alpha.$$

  - $\mathcal{F}(\mathbf{z})$ is the field of rational functions in $\mathbf{z}$ with coefficients in $\mathcal{F}$

$$f(\mathbf{z}) = \frac{p(\mathbf{z})}{q(\mathbf{z})} = \frac{\sum_\alpha n_\alpha \, \mathbf{z}^\alpha}{\sum_\beta d_\beta \, \mathbf{z}^\beta},$$

    - technicality: set $d_{\min\beta} = 1$ to make the representation unique.

Rational reconstruction

Functional reconstruction

Reconstruct the monomials z^α and their coefficients from numerical evaluations of the function (over finite fields)

- from $\mathcal{Q}$ to $\mathcal{Z}_p$

$$q = a/b \in \mathcal{Q} \quad \longrightarrow \quad q \bmod p \equiv a \times (b^{-1} \bmod p) \bmod p$$

- how to go back from $\mathcal{Z}_p$ to $\mathcal{Q}$?
- rational reconstruction algorithm: given $c \in \mathcal{Z}_n$ find its pre-image $q = a/b \in \mathcal{Q}$ with “small” a, b [Wang (1981)]
  - it’s correct when $a, b \lesssim \sqrt{n}$
- make n large enough using Chinese reminder theorem
  - solution in $\mathcal{Z}_{p_1}, \mathcal{Z}_{p_2} \ldots \Rightarrow$ solution in $\mathcal{Z}_{p_1 p_2 \ldots}$

The black-box interpolation problem

Given a polynomial or rational function f in the variables $z = (z_1, \ldots, z_n)$

- reconstruct analytic form of f, given a numerical procedure

$$z \longrightarrow \boxed{f} \longrightarrow f(z),$$

- modified black-box interpolation problem, for usage with finite fields

$$(z, p) \longrightarrow \boxed{f} \longrightarrow f(z) \text{ mod } p.$$

  - the two are equivalent because of Chinese reminder theorem
- no further assumptions on f

Univariate polynomials

- Newton' interpolation formula, form a sequence $\{y_0, y_1, \ldots\}$

$$f(z) = \sum_{r=0}^{R} a_r \prod_{i=0}^{r-1} (z - y_i)$$
$$= a_0 + (z - y_0)\Bigg(a_1 + (z - y_1)\Big(a_2 + (z - y_2)\big(\cdots + (z - y_{r-1})\, a_r\big)\Big)\Bigg)$$

- each coefficient a_i can be determined by evaluations $f(y_j)$ with $j \le i$
  - good when degree is not known
- conversion into canonical form

$$f(z) = \sum_{r=0}^{R} c_r\, z^r.$$

  - addition of univariate polynomials,
  - multiplication of a univ. polynomial by a linear univ. polynomial

Univariate rational functions

- Thiele's (1838–1910) interpolation formula

$$f(z) = a_0 + \cfrac{z - y_0}{a_1 + \cfrac{z - y_1}{a_2 + \cfrac{z - y_3}{\cdots + \cfrac{z - y_{r-1}}{a_N}}}}$$

$$= a_0 + (z - y_0)\left(a_1 + (z - y_1)\left(a_2 + (z - y_2)\left(\cdots + \frac{z - y_{N-1}}{a_N}\right)^{-1}\right)^{-1}\right)^{-1},$$

- analogous to Newton's for rational functions
  - good when degrees of numerator/denominator are not known
- if degrees are known and $d_0 = 1$ (see later), just solve the system

$$f(z) = \frac{\sum_{r=0}^{R} n_r\, z^r}{\sum_{r'=0}^{R'} d_{r'}\, z^{r'}} \quad \Rightarrow \quad \sum_{r=0}^{R} n_r\, y_i^r - \sum_{r'=1}^{R'} d_{r'}\, y_i^{r'} f(y_i) = f(y_i)$$

Multivariate polynomials

- recursive Newton's formula

$$f(z_1,\dots,z_n) = \sum_{r=0}^{R} a_r(z_2,\dots,z_n) \prod_{i=0}^{r-1} (z_1 - y_i),$$

- like univariate with

$$f(y_j) \longrightarrow f(y_j, z_2, \dots, z_n), \qquad a_j \longrightarrow a_j(z_2,\dots,z_n).$$

- convert it back to canonical representation using
    - addition of multivariate polynomials,
    - multiplication of a multiv. polynomial by a linear univ. polynomial.
- very efficient, even for large polynomials

Multivariate rational functions

- dense algorithm, adapted from sparse one by A. Cuyt, W. Lee (2011)
- overall normalization
  - assume non-vanishing constant term in denominator ($d_{(0,\ldots,0)} = 1$)
  - if not the case, shift args. by appropriate vector $\boldsymbol{s}$, using $f_{\boldsymbol{s}} = f(\boldsymbol{z}+\boldsymbol{s})$
- define new function $h \in \mathcal{F}(t,\boldsymbol{z})$ as

$$h(t,\boldsymbol{z}) \equiv f(t\,\boldsymbol{z}) = f(t\,z_1,\ldots,t\,z_n) = \frac{\sum_{r=0}^{R} p_r(\boldsymbol{z})\,t^r}{1+\sum_{r'=1}^{R'} q_{r'}(\boldsymbol{z})\,t^{r'}}$$

where

$$p_r(\boldsymbol{z}) \equiv \sum_{|\alpha|=r} n_\alpha\,\boldsymbol{z}^\alpha, \qquad q_{r'}(\boldsymbol{z}) \equiv \sum_{|\beta|=r'} d_\beta\,\boldsymbol{z}^\beta.$$

$\Rightarrow$ univ. rational fun. in t with (homogeneous) multiv. polynomial coefficients

Multivariate functional reconstruction (summary)

T.P. (2016)

- Univariate polynomials
  - based on Newton's interpolation formula
- Univariate rational functions
  - based on Thiele's (1838–1910) interpolation formula
- Multivariate polynomials
  - recursive application of Newton's interpolation
- Multivariate rational functions
  - use ideas proposed for sparse interpolation [A. Cuyt, W. Lee (2011)]
  - combined with Newton and Thiele's interpolation for dense case

- Notes:
  - all implemented in C++
  - results automatically come out GCD-simplified
  - can be used from a MATHEMATICA interface

Finite-fields and functional reconstruction

- Any algorithm which can be implemented via a sequence of rational operations allows a numerical implementation over $\mathcal{Z}_p$
- Given a numerical procedure computing a rational function f over finite fields $\mathcal{Z}_p$, we can reconstruct the analytic expression of f

$\Rightarrow$ We can perform analytic calculations by implementing equivalent numerical algorithms over finite fields

Example: linear solver

- A $n \times m$ linear system with parametric rational entries

$$\sum_{j=1}^{m} A_{ij}\, x_j = b_i, \quad (j = 1, \ldots, n), \qquad A_{ij} = A_{ij}(z), \quad b_i = b_i(z)$$

- solution $\Rightarrow$ find coefficients $c_{ij} = c_{ij}(z)$ such that

$$x_i = c_{i0} + \sum_{j \in \text{indep}} c_{ij}\, x_j \qquad (i \notin \text{indep})$$

- Functional reconstruction
  - solve system numerically (over finite fields) to evaluate the coefficients $c_{ij}(z)$ of the solution
  - independent equations/variables and vanishing coefficients can be determined quickly and simplify further evaluations
- Very good efficiency compared to traditional computer algebra systems

Applications to scattering amplitudes

Choice of kinematic variables: momentum twistors

Hodges (2009), Badger, Frellesvig, Zhang (2013), Badger (2016)

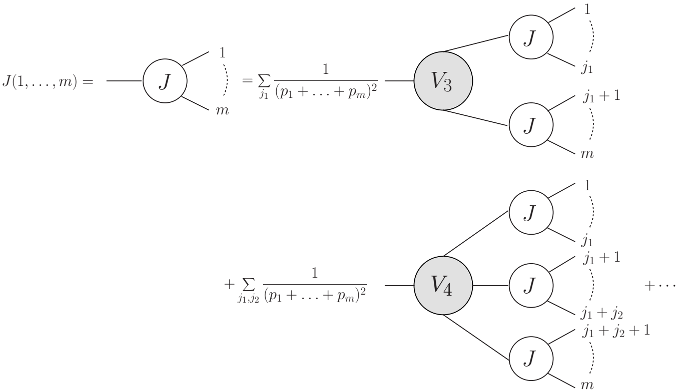
- rational parametrization of the n-point phase-space and the spinor components using $3n - 10$ momentum-twistor variables
- the components of spinors, external momenta and polarization vectors are rational functions of momentum twistor variables

$$|1\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \qquad |2\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix}, \qquad |3\rangle = \begin{pmatrix} \frac{1}{x_1} \\ 1 \end{pmatrix}, \qquad \ldots$$

$$|1] = \begin{pmatrix} 1 \\ \frac{x_4 - x_5}{x_4} \end{pmatrix}, \qquad |2] = \begin{pmatrix} 0 \\ x_1 \end{pmatrix}, \qquad |3] = \begin{pmatrix} x_1\, x_4 \\ -x_1 \end{pmatrix}, \qquad \ldots$$

Both analytic and numerical calculations can be performed operating directly on the components of spinors and momenta

Tree-level amplitudes via Berends-Giele recursion



- very efficient for numerical calculations
- functional reconstruction techniques can exploit this for obtaining analytic results

Integrand reduction

Ossola, Papadopoulos, Pittau (2007)

- generic contribution to a loop amplitude

$$\int_{-\infty}^{\infty} \left(\prod_{i=1}^{\ell} d^d k_i \right) \frac{\mathcal{N}(k_i)}{\prod_j D_j(k_i)},$$

- integrand reduction (integrand as sum of irreducible contributions)

$$\frac{\mathcal{N}(k_i)}{\prod_j D_j(k_i)} = \sum_{T \in \text{topologies}} \frac{\Delta_T(k_i)}{\prod_{j \in T} D_j(k_i)}, \qquad \Delta_T(k_i) = \sum_\alpha c_{T,\alpha}\ (\boldsymbol{m}_T(k_i))^\alpha$$

- the on-shell integrands or residues Δ_T
  - $\{\boldsymbol{m}_T^\alpha\}$ forms a complete integrand basis (see below)
- fit unknown $c_{T,\alpha}$ on multiple cuts $\{D_j = 0\}_{j \in T}$
  - solutions of a linear system

Finding an integrand basis

1. use monomials in a complete set of irreducible scalar products between loop momenta k_i^μ, external momenta p_i^μ and orthogonal vectors ω_i^μ

$$\{\boldsymbol{m}_T\} = \{\boldsymbol{m}_T\}_{\text{complete}} = \{k_i \cdot k_j,\ k_i \cdot p_j\}_{\text{irreducible}} \ \cup \ \{k_i \cdot \omega_j\}_{\omega_i \perp p_j}$$

   - irreducible $\equiv$ not a combination of denominators $D_i \in T$
   - all scalar products $k_i \cdot \omega_j$ are irreducible but they can be integrated out and do not appear in the final result P. Mastrolia, A. Primo, T.P. (2016)

2. use monomials in a overcomplete set of irreducible scalar products

$$\{\boldsymbol{m}_T\} = \{\boldsymbol{m}_T\}_{\text{complete}} \ \cup \ (k_{i,[d-4]} \cdot k_{j,[d-4]}) \ \cup \ \cdots$$

   - the monomials satisfy linear relations which can be inverted (numerically over f.f.) to determine an independent basis
   - by maximizing the presence of $(k_{i,[d-4]} \cdot k_{j,[d-4]})$ we ensure a smooth $d \to 4$ limit, which yields simpler results

Other choices for an integrand basis

- Local integrands for 5- and 6-point 2-loop all-plus amplitudes
  - $\mathcal{N} = 4$ [Arkani-Hamed, Bourjaily, Cachazo, Trnka (2010)]
  - all-plus QCD [Badger, Mogull, T.P. (2016)]
  - free of spurious singularities
  - smooth soft limits to lower-point integrands
  - infrared properties manifest at the integrand level

  $\Rightarrow$ simpler results

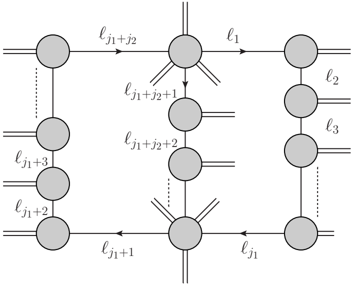
  ✗ ...but no general algorithm for a complete one (yet)
- Other properties worth looking for in the future
  - correspondence with uniform-weight integrals for easier integration (cfr. J. Henn (2013))
- Looking for a good choice using functional reconstruction
  - the functional reconstruction algorithm allows to quickly compute the degree of multivariate functions without a full reconstruction
  - the degree can be used to assess the complexity of the result

Integrand reduction and generalized unitarity

Britto, Cachazo, Feng (2004), Giele, Kunszt, Melnikov (2008), Bern, Dixon, Kosower et al. (2008)

- Generalized unitarity
  - build irreducible integrands from multiple cuts
  - multiple-cuts $\Rightarrow$ loop propagators go on-shell, $\ell_i^2 = 0$
  - integrand factorizes as product of trees (summed over internal helicities)
  - multiple cuts $\Rightarrow$ unitarity cuts

- # unitarity cuts $\ll$ # diagrams
  - lower complexity
- Every intermediate step is gauge invariant
  - no ghosts
  - more compact expressions



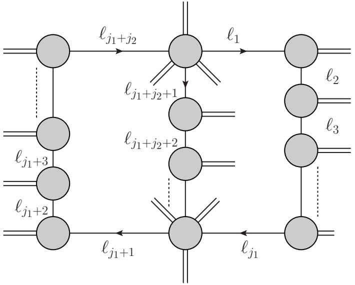
Two-loop unitarity cuts in d dimensions

Bern, Carrasco, Dennen, Huang, Ita (2010), Davies (2011), Badger, Frellesvig, Zhang (2013)

- d-dim. dependence of loops $k_i^\mu \Rightarrow$ embed k_i^μ in $\mathcal{D}$ dimensions $(\mathcal{D} > 4)$
- unitarity cuts $\ell_i^2 = 0 \Rightarrow$ explicit $\mathcal{D}$-dim. representation of loop components
- describe internal on-shell states with $\mathcal{D}$-dim. spinor-helicity formalism
  see e.g. six-dim. formalism by Cheung, O'Connell (2009)
- additional gluon states as $d_s - \mathcal{D}$ scalars ($d_s = 4, d$ in FDH, tHV)

$\mathcal{D} = 6$ sufficient up to two loops

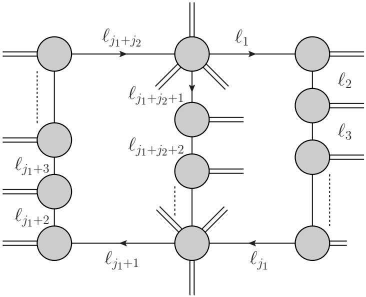
numerical evaluation over finite fields using an explicit (rational) representation of internal states



Generalized unitarity over finite fields

T.P. (2016)

- Amplitudes over finite fields
  - momentum-twistor variables
  - loop states: embed in 6-dim.
  - spinor-helicity in 4 and 6 dim.
  - tree-level recursion
  - two-loop d-dim. unitarity cuts

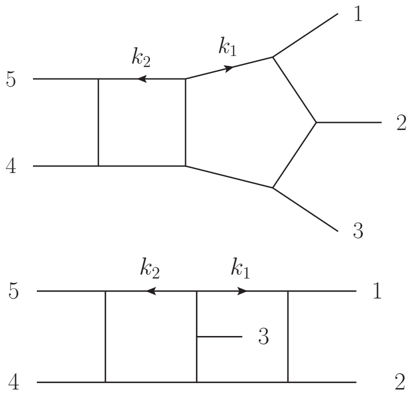


Finite-field implementation

- explicit six-dim. representation of loop states
- efficient numerical techniques for analytic calculations
- two-loop unitarity cuts by sewing Berends-Giele currents
  - sum over helicities only for 2 internal lines
  - the others replaced by contraction of currents

Finite fields and functional reconstruction: examples

- five-gluon on-shell integrands of maximal cuts ($\equiv$ top-level topology) for



(for a complete set of helicities)

Finite fields and functional reconstruction

- penta-box

Helicity	Non-vanishing coeff.	Max. terms	Max. degree	Avg. non-zero terms
$(1^+, 2^+, 3^+, 4^+, 5^+)$	14	19	8	15.00
$(1^-, 2^+, 3^+, 4^+, 5^+)$	27	443	19	152.96
$(1^+, 2^-, 3^+, 4^+, 5^+)$	37	1977	24	674.97
$(1^+, 2^+, 3^+, 4^-, 5^+)$	61	474	18	184.05
$(1^-, 2^-, 3^+, 4^+, 5^+)$	35	1511	24	278.77
$(1^-, 2^+, 3^+, 4^+, 5^-)$	79	7027	34	1112.82
$(1^+, 2^+, 3^+, 4^-, 5^-)$	18	19	8	15.00
$(1^-, 2^+, 3^-, 4^+, 5^+)$	41	2412	22	368.41
$(1^+, 2^-, 3^+, 4^-, 5^+)$	85	18960	42	3934.96
$(1^-, 2^+, 3^+, 4^-, 5^+)$	85	10386	37	1803.52

- double-pentagon

Helicity	Non-vanishing coeff.	Max. terms	Max. degree	Avg. non-zero terms
$(1^+, 2^+, 3^+, 4^+, 5^+)$	104	1937	26	626.39
$(1^-, 2^+, 3^+, 4^+, 5^+)$	104	1449	27	601.43
$(1^+, 2^+, 3^-, 4^+, 5^+)$	104	1554	23	642.90
$(1^-, 2^-, 3^+, 4^+, 5^+)$	99	1751	26	739.05
$(1^+, 2^-, 3^-, 4^+, 5^+)$	104	2524	24	923.71
$(1^-, 2^+, 3^+, 4^+, 5^-)$	104	1838	27	823.00
$(1^-, 2^+, 3^+, 4^-, 5^+)$	104	1307	24	630.48

Summary & Outlook

Summary

- Finite-fields and functional reconstruction techniques
  - can be use to solve complex algebraic problems
  - any function which can be implemented as a sequence of rational operations is suited for these algorithms
- Applications to scattering amplitudes
  - spinor-helicity in four and six dimensions
  - tree-level calculations
  - multi-loop integrand reduction via generalized unitarity

Outlook

- complete five-point two-loop calculations
- apply the algorithm to other techniques (e.g. diagrammatic techniques, tensor reduction, IBPs,. . .)

THANKS!

BACKUP SLIDES

Extended euclidean algorithm

- given integers a, b, find s, t such that

$$a\,s + b\,t = \gcd(a,b).$$

- algorithm: generate sequences of integers $\{r_i\}$, $\{s_i\}$, $\{t_i\}$ and the integer quotients $\{q_i\}$ as follows

$$\begin{aligned} r_0 &= a & \cdots &= \cdots \\ s_0 &= 1 & q_i &= \lfloor r_{i-2}/r_{i-1} \rfloor \\ t_0 &= 0 & r_i &= r_{i-2} - q_i\, r_{i-1} \\ r_1 &= b & s_i &= s_{i-2} - q_i\, s_{i-1} \\ s_1 &= 0 & t_i &= t_{i-2} - q_i\, t_{i-1} \\ t_1 &= 1 & & \end{aligned}$$

- stop when $r_k = 1 \Rightarrow t = t_{k-1}$, $s = s_{k-1}$, $\gcd(a,b) = r_{k-1}$
- multiplicative inverse: if $b = n$ and $\gcd(a,n) = 1 \Rightarrow s = a^{-1}$.

Chinese reminder theorem

- given $a_1 \in \mathcal{Z}_{n_1}$, $a_2 \in \mathcal{Z}_{n_2}$ (n_1, n_2 co-prime) find $a \in \mathcal{Z}_{n_1 n_2}$ such that

$$a \bmod n_1 = a_1, \qquad a \bmod n_2 = a_2.$$

- rational reconstruction over $\mathcal{Q}$
  - reconstruct a function f over several finite fields $\mathcal{Z}_{p_1}, \mathcal{Z}_{p_2}, \ldots$
  - recursively combine the result in $\mathcal{Z}_{p_1 p_2 \cdots}$ using the Chinese reminder
  - use the rational reconstruction algorithm on the combined result over $\mathcal{Z}_{p_1 p_2 \cdots}$ to obtain a guess over $\mathcal{Q}$
  - when $\prod_i p_i$ is large enough the reconstruction is successful
  - the termination criterion is consistency over several finite fields
- we can choose the primes p_i small enough to use machine-size integers
- multi-precision arithmetic only required for Chinese reminder
- 1, 2 or 3 primes are often sufficient

Rational reconstruction: example

- Reconstruct $q = -611520/341$ from its images over finite fields
- $\mathcal{Z}_{p_1}$, with $p_1 = 897473$

$$a_1 = q \text{ mod } p_1 = 13998,$$

$$\text{first guess: } a_1 \xrightarrow{\text{rational rec. over } \mathcal{Z}_{p_1}} g_1 = -411/577$$

- $\mathcal{Z}_{p_2}$, with $p_2 = 909683$

$$a_2 = q \text{ mod } p_2 = 835862$$

$$g_1 \text{ mod } p_2 = 807205 \quad \Rightarrow \quad \text{guess } g_1 \text{ is wrong}$$

- Chinese reminder: $a_1, a_2 \longrightarrow a_{12} \in \mathcal{Z}_{p_1 p_2}$, with $p_1 p_2 = 816415931059$

$$a_{12} \equiv q \text{ mod } p_1 p_2 = 629669763217 \xrightarrow{\text{rational rec. over } \mathcal{Z}_{p_1 p_2}} g_2 = -611520/341$$

- calculation over other fields $\mathcal{Z}_{p_3}, \ldots$ confirm the guess g_2

Choice of variables: spinor-helicity formalism

Mangano, Parke

- tree-level amplitudes and coefficients of loop integrals are rational functions of spinor variables $|p\rangle$ and $|p]$
- satisfying the Dirac equation (in Weyl components)

$$p^\mu \, \sigma_\mu |p\rangle = p^\mu \, \sigma_\mu |p] = 0$$

- momenta and polarization vectors

$$p^\mu = \frac{1}{2} \langle p| \, \sigma^\mu \, |p], \quad \epsilon_+^\mu(p) = \frac{\langle \eta | \sigma^\mu | p]}{\sqrt{2}\, \langle \eta \, p \rangle}, \quad \epsilon_-^\mu(p) = \frac{\langle p | \sigma^\mu | \eta]}{\sqrt{2}\, [p \, \eta]}$$

- helicity amplitudes are combinations of spinor products, e.g.

$$\mathcal{A}_{5g}(1^+, 2^-, 3^+, 4^-, 5^+) = i \, g_s^3 \, \frac{\langle 2\,4 \rangle^4}{\langle 1\,2 \rangle \langle 2\,3 \rangle \langle 3\,4 \rangle \langle 4\,5 \rangle \langle 5\,1 \rangle}$$

- redundancy: spinor components are not all independent

A brief digression on spinor phases

- under a little group tranformation (complex redefinition of phase)

$$|i\rangle \to t_i\, |i\rangle, \qquad |i] \to \frac{1}{t_i}\, |i],$$

  an n-point amplitude $\mathcal{A}(1,\ldots,n)$ transforms as

$$\mathcal{A}(1,\ldots,n) \to \left(\prod_{i=1}^{n} t_i^{-2\,h_i}\right) \mathcal{A}(1,\ldots,n),$$

  where h_i is the helicity of the i-th particle (e.g. $\pm 1/2$ for fermions and ± 1 for gluons)

- extract from the amplitude an overall factor $\mathcal{A}^{(\text{phase})}(1,\ldots,n)$ which transform as the amplitude
- consider $\tilde{A}$ such that

$$\mathcal{A} = \underbrace{\mathcal{A}^{(\text{phase})}}_{\text{only depends on helicities}} \times \underbrace{\tilde{\mathcal{A}}(x_i)}_{\text{phase-free} \,\to\, \text{mom. twist.}}$$

A brief digression on spinor phases

Examples (loop independent):

- possible choices for 5-gluon amplitudes

$$\mathcal{A}^{(\text{phase})}(1^+,2^+,3^+,4^+,5^+) = \frac{1}{\langle 1\,2\rangle\,\langle 2\,3\rangle\,\langle 3\,4\rangle\,\langle 4\,5\rangle\,\langle 5\,1\rangle}$$

$$\mathcal{A}^{(\text{phase})}(1^-,2^+,3^+,4^+,5^+) = \frac{\left(\langle 1\,2\rangle\,[23]\langle 3\,1\rangle]\right)^2}{\langle 1\,2\rangle\,\langle 2\,3\rangle\,\langle 3\,4\rangle\,\langle 4\,5\rangle\,\langle 5\,1\rangle}$$

$$\mathcal{A}^{(\text{phase})}(1^-,2^-,3^+,4^+,5^+) = \frac{\langle 1\,2\rangle^4}{\langle 1\,2\rangle\,\langle 2\,3\rangle\,\langle 3\,4\rangle\,\langle 4\,5\rangle\,\langle 5\,1\rangle},$$

- a choice n-gluon amplitudes

S. Badger (2016)

$$\mathcal{A}^{(\text{phase})}(1^{h_1},\ldots,n^{h_n}) = \left(\frac{\langle 3\,2\,1]}{\langle 3\,1\rangle}\right)^{(h_1-\sum_{i=2}^n h_i)} \prod_{i=2}^{n} \langle i\,1\rangle^{-2h_i}$$

Choice of kinematic variables (phase-free part)

Hodges (2009), Badger, Frellesvig, Zhang (2013), Badger (2016)

- $3n - 10$ momentum-twistor variables
- 5-point example → 5 variables $\{x_1, \ldots, x_5\}$

$$|1\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \qquad |1] = \begin{pmatrix} 1 \\ \frac{x_4 - x_5}{x_4} \end{pmatrix}, \qquad x_k = x_k(s_{ij}, \mathrm{tr}(\sigma_5\, 1\, 2\, 3\, 4))$$

$$|2\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix}, \qquad |2] = \begin{pmatrix} 0 \\ x_1 \end{pmatrix}, \qquad p_i^\mu = \frac{\langle i|\, \sigma^\mu \,|\, i]}{2}$$

$$|3\rangle = \begin{pmatrix} \frac{1}{x_1} \\ 1 \end{pmatrix}, \qquad |3] = \begin{pmatrix} x_1\, x_4 \\ -x_1 \end{pmatrix},$$

$$|4\rangle = \begin{pmatrix} \frac{1}{x_1} + \frac{1}{x_1\, x_2} \\ 1 \end{pmatrix}, \qquad |4] = \begin{pmatrix} x_1 (x_2\, x_3 - x_3\, x_4 - x_4) \\ -\frac{x_1\, x_2\, x_3\, x_5}{x_4} \end{pmatrix},$$

$$|5\rangle = \begin{pmatrix} \frac{1}{x_1} + \frac{1}{x_1\, x_2} + \frac{1}{x_1\, x_2\, x_3} \\ 1 \end{pmatrix}, \qquad |5] = \begin{pmatrix} x_1\, x_3 (x_4 - x_2) \\ \frac{x_1\, x_2\, x_3\, x_5}{x_4} \end{pmatrix}.$$