INTEGRAND REDUCTION RELOADED; ALGEBRAIC GEOMETRY AND FINITE FIELDS

Ray D. Sameshima
New York City College of Technology
CUNY Graduate Center

Ph. D. Advisors: Andrea Ferroglia, Giovanni Ossola

Work done in collaboration with:

P. Mastrolia, A. Primo, W. J. T. Bobadilla(University of Padova),
T. Peraro (University of Edinburgh)
and
A. Broggio (Technische Universität München)
B. D. Pecjak (Durham University)

Supported in part by NSF Grant PHY-1417354

AN OUTLINE

- Scattering amplitudes, integral and integrand decompositions
- Integrand level decomposition with algebraic geometry
- Analytic expression from numerical evaluation by means of finite fields
- First steps towards implementation
- Top quark physics

WHAT IS A SCATTERING AMPLITUDE?

(LHC, for example)

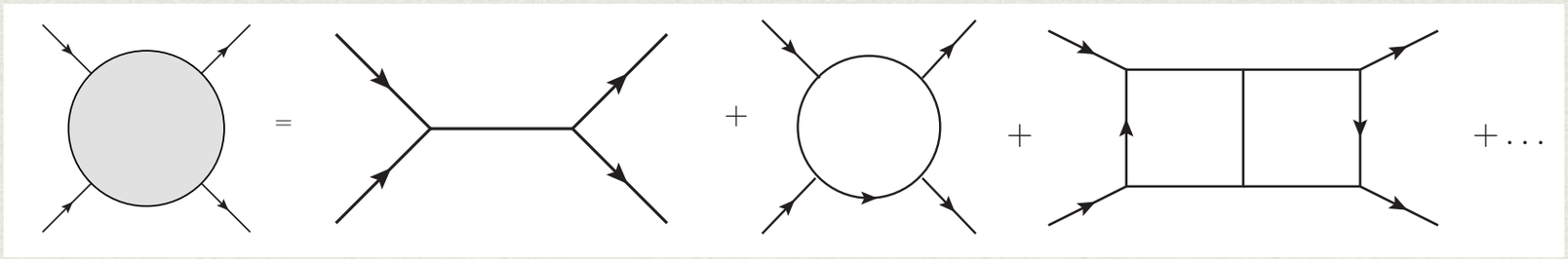
- A connection between collider experiments and theoretical models.

- A scattering amplitude is a probability amplitude, and the cross section is derived from it.

In fact, it (the cross section) is the effective area of a chunk taken out of one beam, …
(Peskin and Schroeder)

LEADING ORDER, NEXT LEADING ORDER, AND HIGHER

A scattering amplitude is computed using a perturbation series:



To understand interactions more deeply and to compare with experiments more accurately, it is required to reduce the theoretical uncertainty.

This will be achieved by taking higher loop amplitudes.

THE FLOW OF AUTOMATION

- Generation
- Reduction
- Evaluation

THE FLOW OF AUTOMATION

- Generation
- Reduction
- Evaluation

"Draw Feynman diagrams"
Given an event, e.g., 2-2 scattering,
generate Feynman diagrams up to certain order.
This is achieved recursively, i.e., add vertices,
virtual lines, and real emissions
on the lower order diagrams.

THE FLOW OF AUTOMATION

- Generation
- Reduction
- Evaluation

Reduce each diagram into the sum of Master Integrals.
OPP method as a standard algorithm.

"partial fraction expansion"

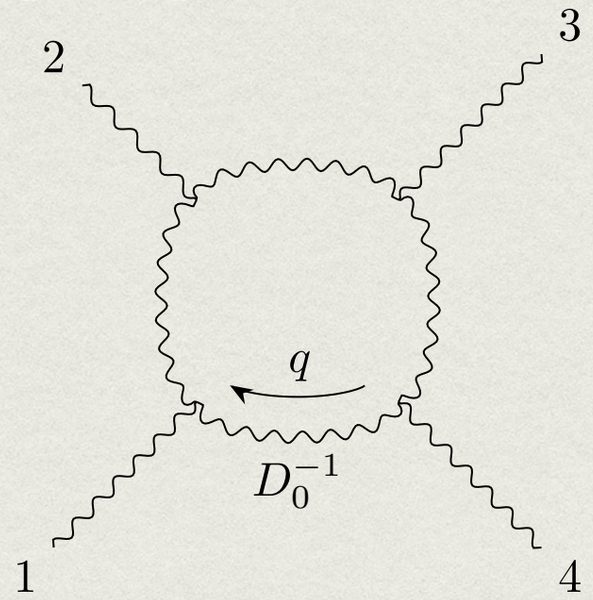
THE FLOW OF AUTOMATION

- Generation
- Reduction
- Evaluation

Integration: brute force computation, Integral By Parts technique, differential equation etc.

AN EXAMPLE

A one-loop $2 \to 2$ scattering diagram is depicted as



We translate it into an integral form using Feynman rules given by the theory we consider:

$$\int d^d q \mathcal{A}(q) = \int d^d q \frac{\mathcal{N}(q)}{D_0 D_1 D_2 D_3}$$

Here the numerator function depends on the theory we consider, but once we draw a diagram then we can write its expression.

INTEGRAL-LEVEL REDUCTION

Consider 1-loop Feynman integral with n denominators

$$\int d^d q \frac{\mathcal{N}(q)}{D_0 \cdots D_{n-1}}$$

The integral-level reduction leads to an expression in terms of Master Integrals which are easier to evaluate than the original integral above:

$$\int d^d q \frac{\mathcal{N}(q)}{D_0 \cdots D_{n-1}} = \sum_i c_i \mathcal{I}_i$$

If Master Integrals are known, all we need is to extract the coefficients (c's) that are given as rational functions of kinematics.

INTEGRAND-LEVEL REDUCTION

Alternatively, we can reduce the integral kernel before integration.

$$\frac{\mathcal{N}(q)}{D_0 \cdots D_{n-1}}$$

Integrand reduction allows us to fit the numerators as the following polynomial form

$$N(q) = \Delta(q) + \sum_j \Delta_j(q) D_j + \sum_{j<k} \Delta_{jk}(q) D_j D_k + \cdots$$

This expression allows us to compute all coefficients in a straightforward manner.

The multivariate polynomial division is a systematic way of the integrand reduction.

G. Ossola, C. G. Papadopoulos and R. Pittau, Nucl. Phys B 763, 147 (2007).
P. Mastrolia and G. Ossola, JHEP 1111, 014 (2011).
P. Mastrolia, E. Mirabella, G. Ossola and T. Peraro, Phys. Lett. B718 (2012).

An algorithmic recipe:

For a given Feynman integral

$$\frac{N}{D_0 \cdots D_{n-1}}$$

take the ideal generated by the denominators

$$\langle D_0, \cdots, D_{n-1} \rangle .$$

The ideal generated by denominators is defined by

$$\langle D_0 \cdots D_{n-1} \rangle := \left\{ \sum_i f_i * D_i \middle| f_i \text{ is a polynomial} \right\}$$

Construct a Gröbner basis of this ideal

$$g_1, \cdots, g_t \in \langle D_0 \cdots D_{n-1} \rangle .$$

Then the reminder Δ is unique, and the polynomial division becomes

$$\begin{aligned} N &= \Delta + \Gamma \\ \Gamma &= \sum_i Q_i g_i . \end{aligned}$$

By construction, $\forall g_i \in \langle D_1 \cdots D_n \rangle$, we have

$$g_i = \sum_j R_{ij} D_j$$

and

$$\begin{aligned} N &= \Delta + \sum_j N_j D_j \\ N_j &= \sum_j Q_i R_{ij} . \end{aligned}$$

Since each element of Gröbner basis is also given by a finite sum of the denominators, and

$$\Gamma = \sum_i Q_i \sum_j R_{ij} D_j$$

Now we can recursively apply the above algorithm with N_j until we reach a fully reduced form:

$$\frac{N_j D_j}{D_0 \cdots D_{n-1}} = \frac{N_j}{D_0 \cdots D_{j-1} D_{j+1} \cdots D_{n-1}}$$

$$\frac{N}{D_0 \cdots D_{n-1}} = \frac{\Delta}{D_0 \cdots D_{n-1}} + \sum_j \frac{N_j D_j}{D_0 \cdots D_{n-1}}$$

Recursive algorithm

P. Mastrolia, E. Mirabella, G. Ossola and T. Peraro, Phys. Lett. B718 (2012).

FACTS

For one loop case, in the dimensionally regulated space ($d = 4 - 2\epsilon$), introducing a new variable μ^2, any diagram with 6 or more denominators can be reduced to diagrams with 5 or less denominators.

$$\frac{N(q)}{D_1 \cdots D_6} = \sum_i \frac{\Delta_i^5}{D_1 \cdots D_{i-1} D_{i+1} \cdots D_6} + \cdots + \sum_i \frac{\Delta_i^1}{D_i} + R.$$

This equality holds integrand level, so as integral level.

Algebraic geometry also provides a systematic way of the proof.

P. Mastrolia, E. Mirabella, G. Ossola and T. Peraro, Phys. Lett. B718 (2012).

OUTLINE OF THE PROOF

Hilbert's weak Nullstellensatz: over algebraic closed field (e.g. complex numbers), an ideal contains 1 iff the polynomials in the ideal share no zeros.

In regularized dimension $d = 4 - 2\epsilon$, we have $4 + 1$ integration variables. In general, 6 equations in 5 variables have no solution:

$$\nexists x \text{ s.t. } D_1(x) = 0, \cdots, D_6(x) = 0.$$ Over constraints !

Then from Nullstellensatz, there exist $c_1 \cdots c_6$ with

$$\forall x, 1 = c_1(x) * D_1(x) + \cdots + c_6(x) * D_6(x).$$

Thus, for an arbitrary numerator, we have

$$\frac{N}{D_1 \cdots D_6} = \sum_i \frac{N c_i}{D_1 \cdots D_{i-1} D_{i+1} \cdots D_6}$$

where the right hand sides have only 5 denominators.

TOWARD NEW EFFICIENT ALGORITHMS

- Integrand reduction method makes problem and its solution clear, but it requires to generate a Gröbner basis for each diagram.

- A new algorithm (adaptive integrand decomposition) has been published recently, which enable to reduce the numerator using clever choice of momentum basis without generating Gröbner basis. This also can reduce the number of parameters for denominators.

$$q = q_{||} + q_{\perp}$$

P. Mastrolia, T. Peraro and A. Primo, JHEP 1680, 164(2016)

FUNCTIONAL RECONSTRUCTION

- There are a list of well-tested numerical implementations for the evaluation of one-loop Feynman Integrals.

```
Samurai, Ninja, Golem,…
```

- Can we extract analytic expression from them?

Given sufficient input-output data, we can reconstruct the generic, analytic expressions for these integrands.

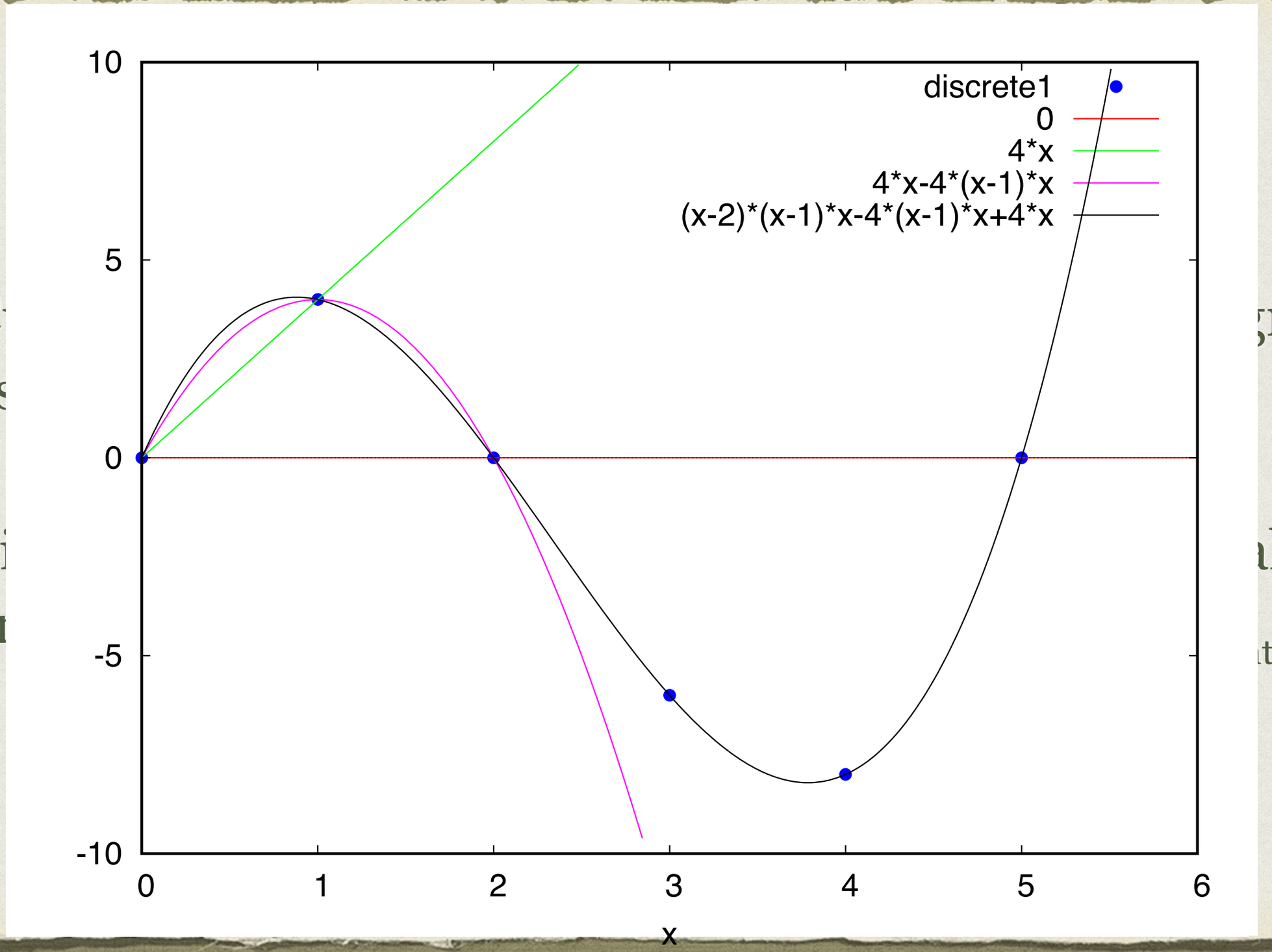
- Newton interpolation for polynomials

- Thiele interpolation for rational functions

Introduction to Numerical Analysis (J. Storer, R. Bulirsch)

FUNCTIONAL RECONSTRUCTION

- C… …gree as…
- Fi… …al ar… …t



WHY?

both space and time in computers

- Cheaper numerical evaluation (everything is numeric, not symbolic)

- Existing platforms (blackbox interpolation)

- Hidden redundancy

  Reconstructed functions are fully reduced form.

FUNCTIONAL RECONSTRUCTION NEWTON'S ALGORITHM

The first differences are constructed from the functional values.

```
Prelude Data.Ratio> let f x = x^3
Prelude Data.Ratio> fs = map f [0..10]
Prelude Data.Ratio> let diff xs = zipWith (-) (tail xs) xs
Prelude Data.Ratio> fs
[0,1,8,27,64,125,216,343,512,729,1000]
Prelude Data.Ratio> diff fs
[1,7,19,37,61,91,127,169,217,271]
Prelude Data.Ratio> diff it
[6,12,18,24,30,36,42,48,54]
Prelude Data.Ratio> diff it
[6,6,6,6,6,6,6,6]
```

```
>    -> [a]
> firstDifs xs = reverse . map head . difLists $ [xs]
```

Haskell Road to Logic, Maths and Programming, Kees Doets and Jan van Eijck, 2004.

FUNCTIONAL RECONSTRUCTION THIELE'S ALGORITHM

Thiele interpolation for integer sampling:

Continuous fraction form

$$r(x) = a_0 + \cfrac{z}{a_1 + \cfrac{z-1}{a_2 + \cfrac{z-2}{\ddots + \cfrac{z-(R-1)}{a_R}}}}$$

$$a_0 = r(0),\ a_r = \cfrac{1}{\cfrac{2}{\cfrac{3}{\ddots - a_{n-2}} - a_{n-1}} - a_2} - a_1$$

Inverse differences are also given by the functional values.

RECIPROCAL DIFFERENCE

Introducing the reciprocal differences which are closely related to the inverse difference:

$$
\begin{aligned}
\rho_{n,i} &:= 0, n < 0 \\
\rho_{0,i} &:= f(i), i = 0, 1, 2, \cdots \\
\rho_{n,i} &:= \frac{n}{\rho_{n-1,i+1} - \rho_{n-1,i}} + \rho_{n-2,i+1} \\
a_0 &:= f(0) \\
a_n &:= \rho_{n,0} - \rho_{n-2,0}
\end{aligned}
$$

Mathematical definition

Haskell definition

```
> rho :: [Ratio Int] -> Int -> Int -> Ratio Int
> rho fs 0 i = fs !! i
> rho fs n _
>   | n < 0 = 0
> rho fs n i = (n*den)%num + rho fs (n-2) (i+1)
>   where
>     num  = numerator next
>     den  = denominator next
>     next = (rho fs (n-1) (i+1)) - (rho fs (n-1) i)

> a fs 0 = fs !! 0
> a fs n = rho fs n 0 - rho fs (n-2) 0
```

RATIONAL FUNCTION RECONSTRUCTION

Consider this function, and pretend we do not know its canonical form.

The 1st row is the outputs of integers.

```
*Univariate> map (\p -> map (rho (map (\t -> 1%(1+t^2)) [0..]) p) [0..3]) [0..5]
[[Just (1 % 1)   ,Just (1 % 2)    ,Just (1 % 5)    ,Just (1 % 10)]
,[Just ((-2) % 1),Just ((-10) % 3),Just ((-10) % 1),Just ((-170) % 7)]
,[Just ((-1) % 1),Just ((-1) % 10),Just ((-1) % 25),Just ((-1) % 46)]
,[Just (0 % 1)   ,Just (40 % 1)   ,Just (140 % 1)  ,Just (324 % 1)]
,[Just (0 % 1)   ,Just (0 % 1)    ,Just (0 % 1)    ,Just (0 % 1)]
,[Nothing,Nothing,Nothing,Nothing]
]
```

4th row is constant sequence, and 5th is infinity

```
*Univariate> let f t = 1/(1+t^2)
*Univariate> let fs = map f [0..]
*Univariate> take 10 fs :: [Ratio Int]
[1 % 1,1 % 2,1 % 5,1 % 10,1 % 17,1 % 26,1 % 37,1 % 50,1 % 65,1 % 82]
*Univariate> list2rat' it
Just ([1 % 1,0 % 1,0 % 1],[1 % 1,0 % 1,1 % 1])
```

Take first 10 out puts as the accessible data.

This outcomes are the list of coefficients, numerator and denominator.

FINITE FIELDS; AN EFFICIENT WAY OF RECONSTRUCTION

Some numerical algorithms can be solved efficiently over finite fields, i.e., prime fields.

We map our problems and solve on finite fields, then we remap them to rational field with high probability.

Let p be an arbitrary prime, then

$$\mathbb{Z}_p := \mathbb{Z}/p\mathbb{Z} \cong \{0, 1, \cdots, (p-1)\}$$

becomes a field, where

$$\begin{aligned} + &: (\mathbb{Z}_p, \mathbb{Z}_p) \to \mathbb{Z}_p; (a, b) \mapsto a + b \mod p \\ * &: (\mathbb{Z}_p, \mathbb{Z}_p) \to \mathbb{Z}_p; (a, b) \mapsto a * b \mod p \end{aligned}$$

P. S . Wang, SYMSAC '81, ACM, 1981

T. Peraro, JHEP 1612 (2016) 030

FUNCTIONAL RECONSTRUCTION OVER A FINITE FILED

Keep away from infinity!

- Treatment of singular and fake-singular

  1/p is the other form of singularity on Z_p.

- Reconstruct functions (formally) on finite fields

  Each coefficient is Z_p value.

- Re-map them into rational fields using Chinese Remainder Theorem

  Rational number reconstruction for coefficients.

P. S . Wang, SYMSAC '81, ACM, 1981

T. Peraro, JHEP 1612 (2016) 030

EXAMPLE

```
*GUniFin> let f x = x^3/ (1+x)^4
*GUniFin> let fs func2graph f [0..20]
*GUniFin> fs
[(0 % 1,0 % 1),(1 % 1,1 % 16),(2 % 1,8 % 81),(3 % 1,27 % 256), ...
```

1 variable case

A finite subgraph of given function

```
*GUniFin> uniRatCoeffm fs
(Just [0 % 1,0 % 1,0 % 1,1 % 1,0 % 1]
,Just [1 % 1,4 % 1,6 % 1,4 % 1,1 % 1]
)
*GUniFin> let fs' = func2graph f [1,3..31]
```

Different set of inputs

```
*GUniFin> uniRatCoeffm fs'
(Just [0 % 1,0 % 1,0 % 1,1 % 1,0 % 1]
,Just [1 % 1,4 % 1,6 % 1,4 % 1,1 % 1])
```

MULTIVARIATE FUNCTIONS

2 variable case

Introducing an auxiliary variable t,

$$h(x, y, t) := f(t * x, t * y)$$

and choosing representative (x, y), we can apply univariate functional reconstruction technique:

$$h(x, y; t) = \frac{\sum_{r=0}^{R} p_r(x, y)t^r}{1 + \sum_{r'=1}^{R'} q_{r'}(x, y)t^{r'}}$$

where each "coefficient" is a homogeneous polynomial in (x, y).

Term by term application of reconstruction

Recursively!

EXAMPLE

```
*GMulFin> let f x y = x*y^2/(1+x+2*y)^3
*GMulFin> twoVariableRational f [1,3..21]
(Just [[0 % 1],[0 % 1],[0 % 1],[0 % 1,0 % 1,1 % 1]]
,Just [[1 % 1],[3 % 1,6 % 1],[3 % 1,12 % 1,12 % 1],[1 % 1,6 % 1,12 % 1,8 % 1]]
)
```

The coefficient for x and y. The coefficient for x^2, x*y and y^2.

Each sub-list corresponds to the homogeneous order.

```
(%i1) f(x,y) := x*y^2/(1+x+2*y)^3;
                                                  2
                                               x y
(%o1)                      f(x, y) := --------------
                                                      3
                                      (1 + x + 2 y)
(%i2) denom(expand(f(x,y)));
         3         2       2       2                      3      2
(%o2) 8 y  + 12 x y  + 12 y  + 6 x  y + 12 x y + 6 y + x  + 3 x  + 3 x + 1
```

Maxima

PROBLEMS

- Expressions for functions on computer systems

  List (recursive data structure), borrow the existing plat forms

- Treatments for singularities

  Interfaces (Error handling, IO, etc)

- Heavy, multi layered recursion (nested loop)

  Better data structure, linear solver (Gaussian elimination)

- Modification of the problem

Functions on machine size integers v.s. coefficients of machine size integers

TOWARD HIGHER LOOPS AND IMPLEMENTATIONS

- We aim to build a fully automation for the evaluation of Feynman integrals with 2 or more loops, with adaptive integrand decomposition algorithm, using combinations of numerical evaluations and reconstructions over finite fields.

- The reconstruction codes can also provide analytic expression for generic processes.

Work in progress with

P. Mastrolia, A. Primo, W. J. T. Bobadilla(University of Padova),
T. Peraro (University of Edinburgh)

BACKGROUND WORKS

- LO calculations for ttW and ttZ with FORM
- Numerical verification with Maxima
- NLO calculations with MG_5 on the cluster system of City Tech

Generation and Evaluation

BACKGROUND WORKS

Analytic computation for LO ttW and ttZ with FORM

"Draw" Feynman diagrams with FORM syntax,
calculate cross section analytically

Numerical verification on Maxima

On Maxima, substitute several phase space points to analytic expression from FORM

```
(%i10) ((1/aem)*ev(qqttWtreelevel(p1,p2,p3,p4,mt,mw,N,aem,sw,%pi),N = 3,
                   numer))
 /(0.2282444653585411*10^-2)
(%o10)                              [1.000000000000008]
```

For a randomly chosen phase point, I compare my own calculation with that of Professor Ferroglia's calculation.

The ratio is 1 plus machine epsilon.

TOP PAIR PHYSICS

JHEP 1704 (2017) 105

Associated production of a top pair and a Z boson at the LHC to NNLL accuracy

(A. Broggio, A. Ferroglia, G. Ossola, B.D. Pecjak and RDS)

Partonic process

$$\left\{ \begin{array}{c} q \\ g \end{array} \right\}(p_1) + \left\{ \begin{array}{c} q \\ g \end{array} \right\}(p_2) \to t(p_3) + \bar{t}(p_4) + \left\{ \begin{array}{c} W^{\pm} \\ Z \\ H \end{array} \right\}(p_5) + X$$

$$\begin{array}{rcl} \hat{s} & := & (p_1 + p_2)^2 \\ M^2 & := & (p_3 + p_4 + p_5)^2 \\ z & := & \dfrac{M^2}{\hat{s}} \overset{\text{"soft"}}{\to} 1 \end{array}$$

$$\hat{O} = \overbrace{\overbrace{1 + \alpha_s(L^2 + L + 1)}^{\text{NLO}} + \alpha_s^2(L^4 + L^3 + L^2 + L + 1) + \mathcal{O}(\alpha_s^3)}^{\text{NNLO}}$$

$$= \exp\Big(\underbrace{\underbrace{Lg_1(\alpha_s L)}_{\text{LL}} + g_2(\alpha_s L)}_{\text{NLL}} + \alpha_s g_3(\alpha_s L) + \cdots\Big) \underbrace{C(\alpha_s)}_{\text{constants}}$$

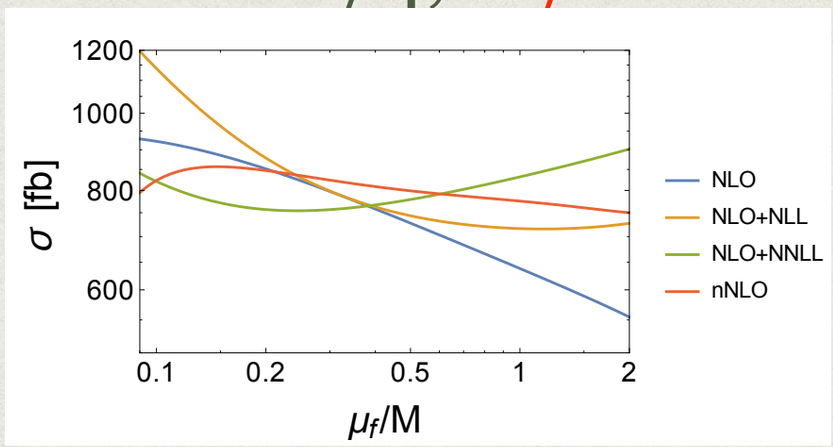
So we need to expand an observable with respect to α_s and L of certain combinations.

TOP PAIR PHYSICS

To see how q-g ch contributes to the scale uncertainty

What I did: LO, NLO calculation with/without quark-gluon channel for W, Z production in 8 TeV and 13TeV with 3 different scales.

M/4, M/2 and M



For fixed order calculation, total cross sections depend on the factorization scale.
We choose this scale so that different perturbative approximations are not so different.

Results:

from NLO to NLO +NNLL accuracy, both total cross sections and differential distributions of ttZ increase, and theoretical uncertainties becomes smaller.

TOP PAIR PHYSICS

ttZ total cross section at 13 TeV

order	PDF order	code	σ [fb]
LO	LO	MG5_aMC	$521.4^{+165.4}_{-116.9}$
app. NLO	NLO	in-house MC	$737.7^{+38.5}_{-64.5}$
NLO no qg	NLO	MG5_aMC	$730.4^{+41.8}_{-64.9}$
NLO	NLO	MG5_aMC	$728.3^{+93.8}_{-90.3}$
NLO+NLL	NLO	in-house MC +MG5_aMC	$742.0^{+90.1}_{-30.3}$
NLO+NNLL	NNLO	in-house MC +MG5_aMC	$777.8^{+61.3}_{-65.2}$
nNLO	NNLO	in-house MC +MG5_aMC	$798.7^{+36.2}_{-23.6}$
$(\text{NLO+NNLL})_{\text{NNLO exp.}}$	NNLO	in-house MC +MG5_aMC	$766.2^{+17.2}_{-50.1}$

Table 2. Total cross section for $t\bar{t}Z$ production at the LHC with $\sqrt{s} = 13$ TeV and MMHT 2014 PDFs. The default value of the factorization scale is $\mu_{f,0} = M/2$, and the uncertainties are estimated through variations of this scale (and of the hard and soft scales μ_s and μ_h when applicable), as explained in the text.

$\sqrt{s}$ and pert. order	process	σ [fb]
8 TeV NLO	$t\bar{t}W^+$	$136.7^{+15.6}_{-15.2}$
8 TeV NLO	$t\bar{t}W^-$	$60.5^{+7.1}_{-6.8}$
8 TeV NLO	$t\bar{t}Z$	$189.8^{+24.5}_{-24.8}$
8 TeV NLO+NNLL	$t\bar{t}W^+$	$130.7^{+6.9}_{-4.9}$
8 TeV NLO+NNLL	$t\bar{t}W^-$	$59.1^{+3.1}_{-2.2}$
8 TeV NLO+NNLL	$t\bar{t}Z$	$203.9^{+13.5}_{-15.8}$
13 TeV NLO	$t\bar{t}W^+$	$356.3^{+43.7}_{-39.5}$
13 TeV NLO	$t\bar{t}W^-$	$182.2^{+23.1}_{-20.4}$
13 TeV NLO	$t\bar{t}Z$	$728.3^{+93.8}_{-90.3}$
13 TeV NLO+NNLL	$t\bar{t}W^+$	$341.0^{+23.1}_{-13.6}$
13 TeV NLO+NNLL	$t\bar{t}W^-$	$177.1^{+12.0}_{-6.9}$
13 TeV NLO+NNLL	$t\bar{t}Z$	$777.8^{+61.3}_{-65.2}$

Table 3. Total cross section for $t\bar{t}Z$ and $t\bar{t}W$ production at the LHC with $\sqrt{s} = 8$ and 13 TeV and MMHT 2014 PDFs. The default value of the factorization scale is $\mu_{f,0} = M/2$, and the uncertainties are estimated through variations of this scale (and of the resummation scales μ_s and μ_h when applicable).

ttW total cross section at 13 TeV

SUMMARY

- Integral-level(integral) and integrand-level(polynomial)
- Algebraic geometry as tools, and a new algorithm for decomposition
- The usage of finite fields in reconstruction
- Toward implementations
- Top quark physics

ADAPTIVE INTEGRAND DECOMPOSITION

- Clever choice of parameter for momentum space reveals, so called, spurious terms which will vanish under integration.

$$q = q_{\parallel} + q_{\perp}$$

- This choice makes the polynomial division problems into a set of liner equation.

P. Mastrolia, T. Peraro and A. Primo, JHEP 1680, 164(2016)

GROEBNER BASES

Properties

- unique remainder
- iff condition for ideal membership
- generating algorithm (Buchberger, F4, etc.)

Ideals, varieties, and algorithms (Cox, Littele, and O'Shea)

RECONSTRUCTION

- Black-box problem, i.e., we can access only input-output data.

- From the table of in-out, i.e., a sub-graph of the function, we build the function itself.

For an arbitrary function $f : A \to B$,

$$G(X \subset A) := \{\, (x, f(x)) |\, x \in X \}\,.$$

From this sub-graph, the reconstruction means to find

$$f' : X \to B \text{ s.t. } \; f'|_X = f|_X\,.$$

OVER FINITE FIELDS

- For efficiency, but it can also prevent from the machine size integer overflow.

Extended Euclidean algorithm as a constructive proof for Bezöut's lemma

Chinese remainder theorem

Rational number reconstruction for coefficients.

P. S . Wang, SYMSAC ’81, ACM, 1981
T. Peraro, JHEP 1612 (2016) 030

EEA WITH HASKELL CODE

Extended Euclidean Algorithm

The inputs are two integers a, b and the out puts are $\gcd(a, b)$ and four lists $\{q_i\}_i, \{r_i\}_i, \{s_i\}_i, \{t_i\}_i$. The base cases are

$$\begin{aligned}(r_0, s_0, t_0) &:= (a, 1, 0)\\ (r_1, s_1, t_1) &:= (b, 0, 1)\end{aligned}$$

and inductively, for $i \geq 2$,

$$\begin{aligned}q_i &:= \text{quot}(r_{i-2}, r_{i-1})\\ r_i &:= r_{i-2} - q_i * r_{i-1}\\ s_i &:= s_{i-2} - q_i * s_{i-1}\\ t_i &:= t_{i-2} - q_i * t_{i-1}.\end{aligned}$$

The termination condition is

$$r_k = 0$$

for some $k \in \mathbb{N}$ and

$$\begin{aligned}\gcd(a, b) &= r_{k-1}\\ x &= s_{k-1}\\ y &= t_{k-1}.\end{aligned}$$

```
> exGCD' :: (Integral n) => n -> n -> ([n], [n], [n], [n])
> exGCD' a b = (qs, rs, ss, ts)
>   where
>     qs = zipWith quot rs (tail rs)
>     rs = takeUntil (==0) r'
>     r' = steps a b
>     ss = steps 1 0
>     ts = steps 0 1
>     steps a b = rr
>       where
>         rr@(_:rs) = a:b: zipWith (-) rr (zipWith (*) qs rs)
>
> takeUntil :: (a -> Bool) -> [a] -> [a]
> takeUntil p = foldr func []
>   where
>     func x xs
>       | p x = []
>       | otherwise = x : xs
```

Bezöut's identity:

$$\forall a, b \in \mathbb{N}, \exists x, y \in \mathbb{Z} \text{ s.t. } a * x + b * y = \gcd(a, b),$$

where gcd is the greatest common divisor.

CRT WITH HASKELL CODE

Chinese Remainder Theorem

Let $p_1, p_2 \in \mathbb{N}$ be coprime (not necessarily primes), then for arbitrary $a_1, a_2 \in \mathbb{N}$,

$$\begin{aligned} x &= a_1 \mod p_1 \\ x &= a_2 \mod p_2 \end{aligned}$$

have a unique solution modulo $p_1 * p_2$. Indeed, the solution is

$$a := a_1 * p_2 * m_1 + a_2 * p_1 * m_2,$$

where $m_1 := p_1^{-1} \mod p_2, m_2 := p_2^{-1} \mod p_1$

This can merge several prime fields into a quotient ring of big product.

$(a_1, p_1), (a_2, p_2) \mapsto (a, p_1 * p_2)$ on $Z_{p_1 * p_2}$

```
> crtRec'
>   :: Integral a =>
> .    (Maybe a, a) -> (Maybe a, a) -> (Maybe a, a)
> crtRec' (Nothing,p) (_,q)        = (Nothing, p*q)
> crtRec' (_,p)       (Nothing,q) = (Nothing, p*q)
> crtRec' (Just a1,p1) (Just a2,p2) = (Just a,p)
>   where
>     a = (a1*p2*m2 + a2*p1*m1) `mod` p
>     Just m1 = p1 `inversep` p2
>     Just m2 = p2 `inversep` p1
>     p = p1*p2
```

HASKELL LANGUAGE

- A standardized purely functional lazy language for general purpose.

www.haskell.org

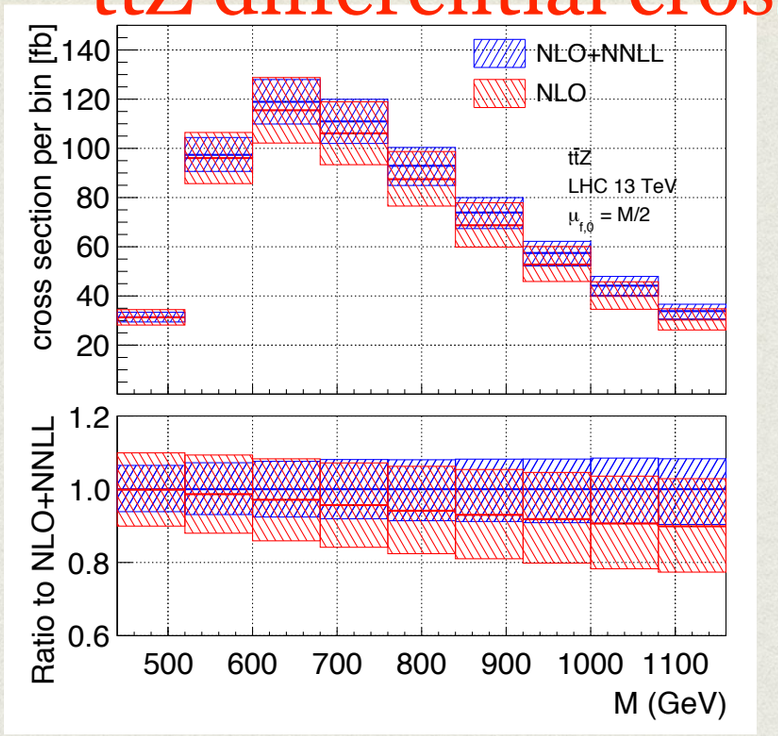
Haskell is a computer programming language.
In particular, it is a polymorphically statically typed, lazy, purely functional language, quite different from most other programming languages.
The language is named for Haskell Brooks Curry, whose work in mathematical logic serves as a foundation for functional languages.
Haskell is based on the lambda calculus, hence the lambda we use as a logo.

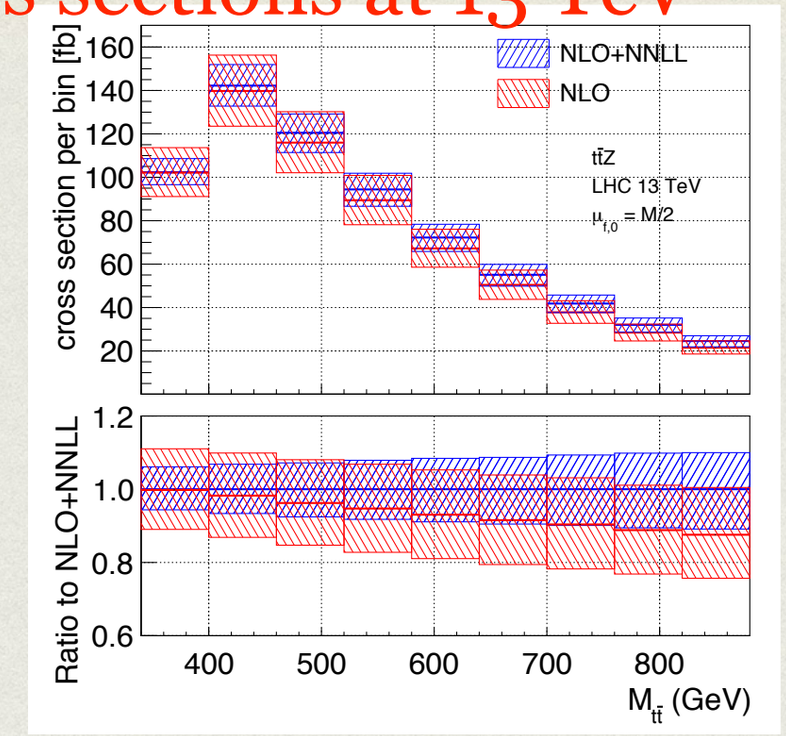
https://wiki.haskell.org/Introduction

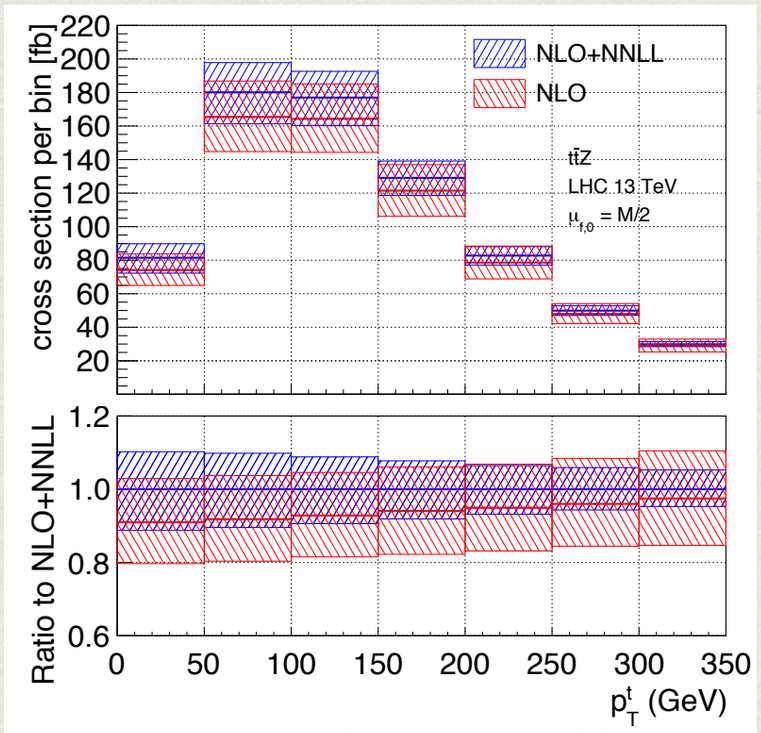
HASKELL

- What (definition) v.s. How (procedure)
- lambda calculus v.s. Turing machine
- immutable and lazy

ttZ differential cross sections at 13 TeV







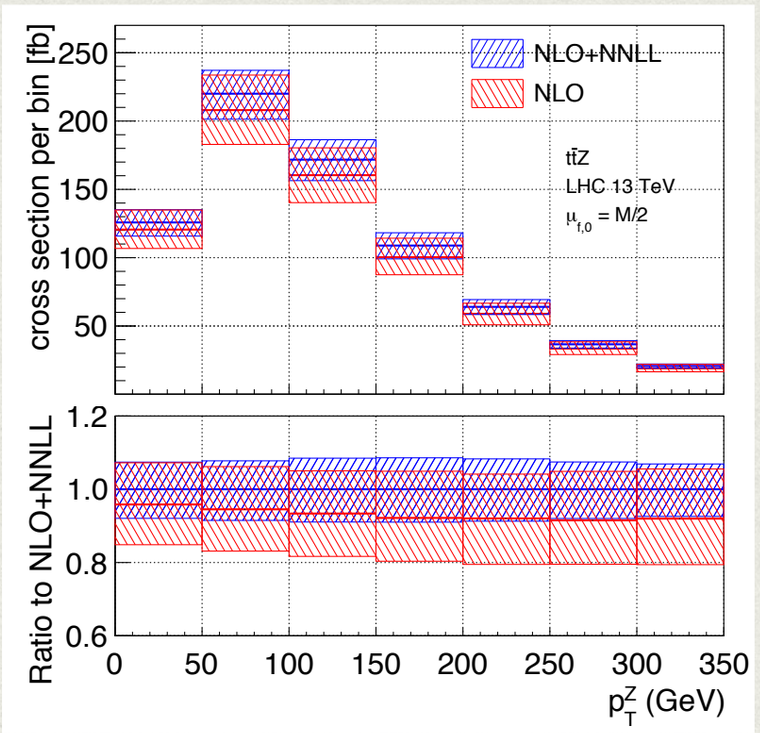


Figure 5. Differential distributions with $\mu_{f,0} = M/2$ at NLO+NNLL (blue band) compared to the NLO calculation (red band). The uncertainty bands are generated through scale variations of μ_f, μ_s and μ_h as explained in the text.