Efficient supercomputer-scale IBP reduction

Mao Zeng, University of Edinburgh

Scattering Amplitudes @ Liverpool, 26 Mar 2025

A.V. Smirnov, MZ, arXiv:2311.02370 (FIRE 6.5)

A.V. Smirnov, MZ, arXiv:2409.19099 (Balanced Zippel Reconstruction)

Bern, Herrmann, Roiban, Ruf, Smirnov, Smirnov, MZ, arXiv:2406.01554 (SUSY black hole scattering @ 4 loops)

Outline

- Background
- Faster analytic IBP in FIRE 6.5
- Parallel finite field runs with FIRE MPI
- How to choose seed integrals

Background

Background

- Loop calculations extract precision predictions from QFT, with many applications.
  - Collider physics (QCD, electroweak)
  - Gravitational wave physics (post-Newtonian, post-Minkowskian expansions)
  - Cosmological correlators
  - Statistical physics

Background

- Integration-by-parts (IBP) reduction [Chetyrkin, Tkachov, '81] is ubiquitous in modern Feynman integral calculations.
- Integrals of total derivatives vanish in dim. reg. ⇒ Linear relations between integrals with different propagator / numerator powers

$$0 = \int d^d \ell \frac{\partial}{\partial \ell^\mu} \frac{k^\mu}{\rho_1^{a_1} \rho_2^{a_2} \dots \rho_n^{a_n}}$$

Seed integral

Background

- Integration-by-parts (IBP) reduction [Chetyrkin, Tkachov, '81] is ubiquitous in modern Feynman integral calculations.
- Integrals of total derivatives vanish in dim. reg. ⇒ Linear relations between integrals with different propagator / numerator powers

IBP operator $\frac{\partial}{\partial \ell^\mu} k^\mu$

$$0 = \int d^d \ell \, \frac{\partial}{\partial \ell^\mu} \frac{k^\mu}{\rho_1^{a_1} \rho_2^{a_2} \ldots \rho_n^{a_n}}$$

Laporta algorithm

- Solves large linear system to express complicated integrals in terms of simple integrals, under some ordering. [Laporta, '01]
- **Codes:** AIR, Reduze, LiteRed, FIRE, Kira, FiniteFlow, Blade, NeatIBP…
- Alternatives: symbolic reduction rules, intersection theory, Groebner bases, D modules… Or not doing IBP at all (SecDec, LTD, FeynTrop…)
- **Optimizations & Variations**: trimming IBP equations by Lie algebra, syzygy equations, numerical unitairty, finite fields & function reconstruction, improved master basis, block triangular form…

FIRE

- "**F**eynman **I**ntegral **Re**duction". IBP code developed over many years [A.V. Smirnov, '08. A.V. Smirnov, V.A. Smirnov, '13. A.V, Smirnov, 14. A.V. Smirnov, F.S. Chukharev, '19. A.V. Smirnov, MZ, '23]
- Implements Laporta algorithm. Can use symmetry & reduction rules from LiteRed [R.N. Lee, '12]. Initially written in Mathematica, available in C++ since version 5. Supports modular arithmetic, MPI in version 6.
- Trims IBP equations by Lie algebra. [R.N. Lee, '08] Forward reduction w/ tail masking [Anastasiou, Lazapoulos, "04], then backward substitution.

Faster analytic IBP: FIRE 6.5 / FLINT

Faster analytic IBP: FIRE 6.5 / FLINT

- During IBP calculation, FIRE (C++ version) needs external help in simplifying expressions of the form

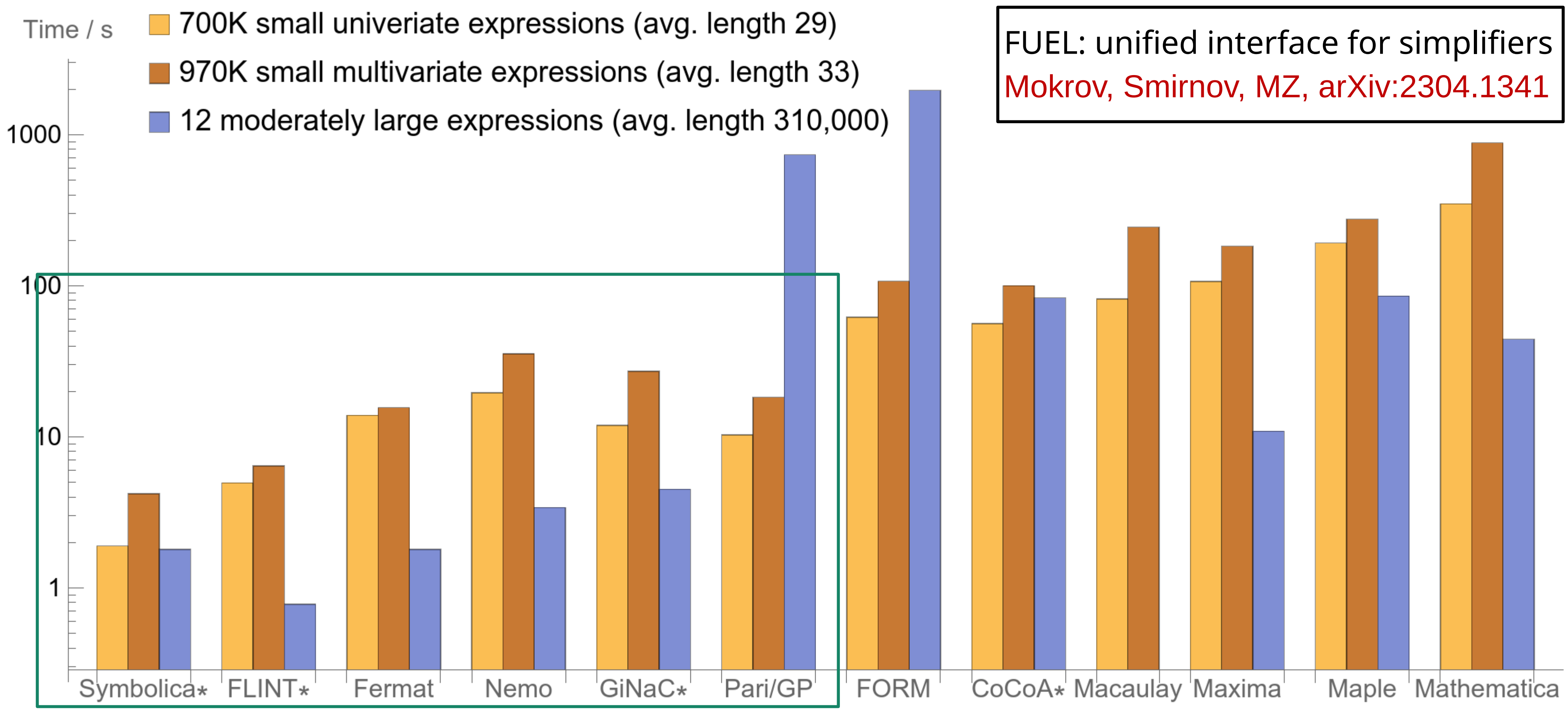
$$\sum_k \frac{\frac{\mathrm{Poly}_{k,1}}{\mathrm{Poly}_{k,2}} \cdot \frac{\mathrm{Poly}_{k,3}}{\mathrm{Poly}_{k,4}}}{\frac{\mathrm{Poly}_{k,5}}{\mathrm{Poly}_{k,6}}}$$

→ *Polynomial in Horner form $a+x(b+x(c+dx))$*

or expanded form $a+bx+cx^2+dx^3$

- FIRE assembles the expression into a string and send to an external simplifier (Mathematica, Maple, Fermat, FLINT...)
- The simplifier simplifies the expression and sends back a string - *overhead* in string parsing/printing besides *actual simplification*.

Benchmark: small to moderate complexity



FUEL: unified interface for simplifiers
Mokrov, Smirnov, MZ, arXiv:2304.1341

Benchmark: huge expression

$$x = \frac{(a+b+c+d+f+g)^{14}+3}{(2a+b+c+d+f+g)^{14}+4} - \frac{(3a+b+c+d+f+g)^{14}+5}{(4a+b+c+d+f+g)^{14}+6}$$

Simplifier	Time taken via FUEL (seconds)
FLINT	**5.2**
Symbolica	**5.2**
Nemo	6.9
Maple	7.9
Fermat	**98.3**
Maxima	112.8
Mathematica	169

FLINT is recommended simplifier

- Symbolica by Ben Ruijl (commerical but free for single-core use) also an excellent option; can be fastest in factorized mode.
- Fermat by Robert Lewis served HEP community well for many years (also e.g. Fermatica), but better options exist.
- Use option `--calc=flint` when running FIRE to select the simplifer. ~10 times faster for five-scale problems.
- Internal development versions exclusively use FLINT.

Parallel finite field runs

Analytics from numerics

- Solving linear systems over polynomials / rational functions of parameters is difficult – parallelize by solving at many **numerical** values, then reconstruct analytic form.

- **Finite field** numerics (no roundoff errors) + **reconstructing rational functions** – a mini-revolution in analytic manipulations in loop calculations. *Codes: Finred, FiniteFlow, FireFly, FIRE…* [von Manteuffel, Schabinger, ’14. Peraro, ‘16, ’19. Klappert, Lange, ’19. Klappert, Klein, Lange, ’20. de Laurentis, Page, ’22, Belitsky, Smirnov, Yakovlev, ’23. Chawdhry, ’23. Liu, ’23. Maier, ’24… + many process-specific papers] i

Univariate reconstructions

- Polynomials: Newton reconstruction

$$f(x_i) = a_i,$$

$$f_N(x) = a_1 + (x - x_1)\Big[a_2 + (x - x_2)\big[a_3 + (x - x_3)[a_4 + \dots]\big]\Big]$$

- Rational functions: Thiele reconstruction

$$f_T(x) = b_1 + (x - x_1)\Big[b_2 + (x - x_2)\big[b_3 + (x - x_3)[b_4 + \dots]^{-1}\big]^{-1}\Big]^{-1}$$

where b_i are defined in terms of $f(x_j)$ for j =1, 2, … , i

Challenges in multivariate reconstruction

- Builds upon univariate Newton & Thiele reconstructions.
- One approach for multivarite case: homogeneous scaling [Peraro, '15]:

$$\frac{f(x,y,z)}{g(x,y,z)} \to \frac{f(t\tilde{x},t\tilde{y},t\tilde{z})}{g(t\tilde{x},t\tilde{y},t\tilde{z})}$$

- Thiele reconstruction of rational function in *t*, then Newton reconstruction of coefficients as polynomials in $(\tilde{x},\tilde{y},\tilde{z})$.

Why not homogeneous scaling

$$\frac{f(x,y,z)}{g(x,y,z)} \to \frac{f(t\tilde{x},t\tilde{y},t\tilde{z})}{g(t\tilde{x},t\tilde{y},t\tilde{z})}$$

- Degree in *t* can be higher than degree in each individual variable ⇒ more complicated Thiele reconstruction
- To prevent zero constant term in denominator, need to shift varibles ⇒ may destroy sparsity (if present)

Alternative – balanced reconstruction

[Belitsky, Smirnov, Yakovlev, ’23]

- Preserves the “individuality” of each variable w/o mixing.
- Simple example in [Smirnov, MZ, ’24] extending it to the sparse case:

$$f(x,y) = \frac{xy+2}{xy-2x+4} = \frac{n(x,y)}{d(x,y)}$$

highest deg. Polynomial in denominator normalized to unit coefficient

- “Balance” the reconstruction in *x* with that in *y* to cancel normalization discrepancies, recover *n* and *d* separately.

“Balancing” example

$$f(x,y) = \frac{xy+2}{xy-2x+4} = \frac{n(x,y)}{d(x,y)}$$??

- First, set y to a e.g. y_1=4, y_2=5 … and reconstruct x dependence by Thiele

$$f(x, y_1 = 4) = \frac{4x+2}{2x+4} = \frac{2x+1}{x+2} = \frac{n'(x,y_1)}{d'(x,y_1)}$$ x polynomials for each y_i

highest deg. Polynomial in denominator normalized to unit coefficient

- Next, reconstruct y dependence for an arbitrary base value x=x_0

$$f(x=x_0, y) = \frac{y+2/x_0}{y+4/x_0-2} = \frac{n''(x=x_0,y)}{d''(x=x_0,y)}$$ y polynomials for x_0

$$f(x,y)=\frac{xy+2}{xy-2x+4}=\boxed{\frac{n(x,y)}{d(x,y)}}$$??

$$f(x,y_1=4)=\frac{4x+2}{2x+4}=\boxed{\frac{2x+1}{x+2}=\frac{n'(x,y_1)}{d'(x,y_1)}}$$ *x* polynomials for each y_i

$$f(x=x_0,y)=\boxed{\frac{y+2/x_0}{y+4/x_0-2}=\frac{n''(x=x_0,y)}{d''(x=x_0,y)}}$$ *y* polynomials for x_0

- **Balancing:** choose an arbitrary base value e.g. x_0 = 5, then form the ratio

$$\frac{n'(x,y_i)\cdot n''(x_0,y_i)}{n'(x_0,y_i)}=n(x,y_i)\cdot c$$

No y_i dependence, only depends on constant x_0 = 5. Now reconstruct n and d, with same extra factor, from a sequence of y_i by ***Newton reconstruction.***

$$f(x,y) = \frac{xy+2}{xy-2x+4} = \boxed{\frac{n(x,y)}{d(x,y)}}$$ **??**

$$f(x, y_1 = 4) = \frac{4x+2}{2x+4} = \boxed{\frac{2x+1}{x+2} = \frac{n'(x,y_1)}{d'(x,y_1)}}$$ x polynomials for each y_i

$$f(x = x_0, y) = \boxed{\frac{y + 2/x_0}{y + 4/x_0 - 2} = \frac{n''(x = x_0, y)}{d''(x = x_0, y)}}$$ y polynomials for x_0

- **Balancing:** choose an arbitrary base value e.g. $x_0 = 5$, then form the ratio

$$\frac{n'(x, y_i = 4) \cdot n''(x_0, y_i = 4)}{n'(x_0, y_i = 4)} = \frac{(2x+1)(4+2/5)}{2 \cdot 5 + 1} = \frac{4x+2}{5} = \frac{n(x,y_i)}{5}$$

No y_i dependence, only depends on constant $x_0 = 5$. Now reconstruct n and d, with same extra factor, from a sequence of y_i by ***Newton reconstruction.***

Balanced Zippel Reconstruction

- Balanced reconstruction extended to sparse rational functions in [Smirnov, MZ, ’24] by combining with Zippel method.
- FIRE MPI orchestrates finite-field reduction and reconstruction processes on clusters and supercomputers.
- In development: **multiple finite fields** run together, significant reduction of overhead.
- In development: **GPU Zippel reconstruction** by solving Vandermonde linear system. [Smirnov, Rozhnov, in progress]

How to choose seed integrals

- Laporta’s golden rule: cutoff at *r* and *s* at the same values of the integrals to reduce.
  - *r*: total number of denominator powers
  - *s*: total number of ISP powers (tensor rank)
- FIRE: reduce to masters + lower sectors, then restart seed selection in lower sectors. Also uses Roman Lee’s Lie algebra methods to trim seeds.

Inspiration from syzygy equations

$$0 = \int d^d\ell \frac{\partial}{\partial \ell^\mu} \frac{k^\mu}{\rho_1^{a_1} \rho_2^{a_2} \cdots \rho_n^{a_n}}$$

IBP operator $\frac{\partial}{\partial \ell^\mu} k^\mu$

Seed integral

- Syzygy equation method [Gluza, Kajda, Kosower, '10. Schabinger '14. Ita '15. Larsen, Zhang, '15…] : Take seed integral with no "dots", but use algebraic geometry to find polynomial-linear combination of IBP operators to generate dot-free IBP linear equations.
- Success implies that you can also use (mostly) **no-dot integrals** in Laporta approach. Seed-operator pairs implicitly span syzygy solutions

Putting it together – 4-loop gravity

[Bern, Herrmann, Roiban, Ruf, Smirnov, Smirnov, MZ, arXiv:2406.01554 + ongoing]

- Classical GR dynamics from scattering amplitudes – LIGO and beyond.
- Low degree & sparse in velocity parameter y, high degree in dimension d - balanced Zippel reconstruction avoids mixing & preserves sparsity
- Mostly no-dot seed integrals favored.
- Improved pre-solving: Gaussian elimination on IBP operators
- Similar seeding improvements in Kira: worldline calculation of binary dynamics [Driesse, Jakobsen, Mogull, Plefka, Sauer, Usovitsch, '24]

Conclusion

- **FIRE** being actively developed – improvements in both analytic and finite-field reduction & reconstruction.
- **FLINT** provides open-source, high-performance polynomial simplifications.
- **Balanced Zippel reconstruction** avoids homogeneous scaling which mixes variables in reconstruction, and exploits sparsity. **MPI enabled.**
- **Seed integral selection** can have dramatic effects on IBP performace. See also recent work using *genetic algorithms & machine learning:*

(1) Symbolic decision tree for inclusion of seed integrals [von Hippel, Wilhelm, '25]

(2) Quadratic priority function for seeding [Song, Yang, Cao, Luo, Zhu, '25]