



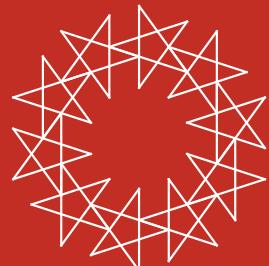
A Tale of Two Packages

pfd-parallel and *NeatIBP*

for multi-loop computations
for multi-loop computations

Yang Zhang

University of Science and Technology of China



Feb 14, 2023

Mathematical Structures in Feynman Integrals

Apologize

I could not attend this exciting conference ...

Although the Covid travel restriction in China was lift last December, I could not get a German visa on time ...

Apologize for giving an online talk,

and thank the conference organizers a lot for giving me this opportunity!

A tale of two packages



photo from USPS website

pfd-parallel

simplifies rational functions
in amplitudes computations,
by multivariate partial fraction

NeatIBP

generates small size
IBP systems based on
syzygy/module intersection

pfd-parallel

A package simplifies rational functions
in amplitudes computations
by **multivariate partial fraction**

Bendle, Boehm, Heymann, Ma, Rahn, Ristau, Wittmann, Wu, Xu and YZ

<https://github.com/singular-gpispace/pfd-parallel/>

arXiv: 2104.06866
package finished on Oct. 2022
to be submitted to Comp. Phys. Comm.

Why do we need multivariate partial fraction?

It is well known that sometimes in amplitudes computations,
multivariate partial fraction can greatly simplify **rational functions**.

- simplify coefficients in Amplitudes
 - simplify IBP reduction coefficients
 - boost canonical DE construction
- Meyer, CPC 222 (2018)
 Abreu, Dormans, Febres Cordero, Ita, Page, Sotnikov, JHEP 05 (2019) 08
 Agarwal, Buccioni, von Manteuffel, Tancred, PRL 127 (26) (2021) 26200

$$\begin{aligned} & -64x_1x_2^2x_3^2x_4^2 - 36x_1x_2^2x_3^2x_4^3 - 64x_1x_2^2x_3^2x_4^4 - 48x_1x_2^2x_3^2x_4^5 - 12x_1x_2^2x_3^2x_4^6 \\ & - 32x_1x_2^2x_3^2x_4^7 - 100x_1x_2^2x_3^2x_4^8 - 48x_1x_2^2x_3^2x_4^9 - 12x_1x_2^2x_3^2x_4^{10} - 48x_1x_2^2x_3^2x_4^{11} \\ & + 12x_1x_2^2x_3^2x_4^{12} - 120x_1x_2^2x_3^2x_4^{13} - 48x_1x_2^2x_3^2x_4^{14} - 12x_1x_2^2x_3^2x_4^{15} - 48x_1x_2^2x_3^2x_4^{16} \\ & - 12x_1x_2^2x_3^2x_4^{17} - 100x_1x_2^2x_3^2x_4^{18} - 48x_1x_2^2x_3^2x_4^{19} - 12x_1x_2^2x_3^2x_4^{20} - 48x_1x_2^2x_3^2x_4^{21} \\ & + 24x_1x_2^2x_3^2x_4^{22} - 64x_1x_2^2x_3^2x_4^{23} - 16x_1x_2^2x_3^2x_4^{24} + 24x_1x_2^2x_3^2x_4^{25} - 24x_1x_2^2x_3^2x_4^{26} \\ & - 24x_1x_2^2x_3^2x_4^{27} - 64x_1x_2^2x_3^2x_4^{28} - 16x_1x_2^2x_3^2x_4^{29} + 24x_1x_2^2x_3^2x_4^{30} - 24x_1x_2^2x_3^2x_4^{31} \\ & + 24x_1x_2^2x_3^2x_4^{32} - 64x_1x_2^2x_3^2x_4^{33} - 16x_1x_2^2x_3^2x_4^{34} + 24x_1x_2^2x_3^2x_4^{35} - 24x_1x_2^2x_3^2x_4^{36} \\ & - 24x_1x_2^2x_3^2x_4^{37} - 64x_1x_2^2x_3^2x_4^{38} - 16x_1x_2^2x_3^2x_4^{39} + 24x_1x_2^2x_3^2x_4^{40} - 24x_1x_2^2x_3^2x_4^{41} \\ & + 24x_1x_2^2x_3^2x_4^{42} - 64x_1x_2^2x_3^2x_4^{43} - 16x_1x_2^2x_3^2x_4^{44} + 24x_1x_2^2x_3^2x_4^{45} - 24x_1x_2^2x_3^2x_4^{46} \\ & - 24x_1x_2^2x_3^2x_4^{47} - 64x_1x_2^2x_3^2x_4^{48} - 16x_1x_2^2x_3^2x_4^{49} + 24x_1x_2^2x_3^2x_4^{50} - 24x_1x_2^2x_3^2x_4^{51} \\ & + 24x_1x_2^2x_3^2x_4^{52} - 64x_1x_2^2x_3^2x_4^{53} - 16x_1x_2^2x_3^2x_4^{54} + 24x_1x_2^2x_3^2x_4^{55} - 24x_1x_2^2x_3^2x_4^{56} \\ & - 24x_1x_2^2x_3^2x_4^{57} - 64x_1x_2^2x_3^2x_4^{58} - 16x_1x_2^2x_3^2x_4^{59} + 24x_1x_2^2x_3^2x_4^{60} - 24x_1x_2^2x_3^2x_4^{61} \\ & + 24x_1x_2^2x_3^2x_4^{62} - 64x_1x_2^2x_3^2x_4^{63} - 16x_1x_2^2x_3^2x_4^{64} + 24x_1x_2^2x_3^2x_4^{65} - 24x_1x_2^2x_3^2x_4^{66} \\ & - 24x_1x_2^2x_3^2x_4^{67} - 64x_1x_2^2x_3^2x_4^{68} - 16x_1x_2^2x_3^2x_4^{69} + 24x_1x_2^2x_3^2x_4^{70} - 24x_1x_2^2x_3^2x_4^{71} \\ & + 24x_1x_2^2x_3^2x_4^{72} - 64x_1x_2^2x_3^2x_4^{73} - 16x_1x_2^2x_3^2x_4^{74} + 24x_1x_2^2x_3^2x_4^{75} - 24x_1x_2^2x_3^2x_4^{76} \\ & - 24x_1x_2^2x_3^2x_4^{77} - 64x_1x_2^2x_3^2x_4^{78} - 16x_1x_2^2x_3^2x_4^{79} + 24x_1x_2^2x_3^2x_4^{80} - 24x_1x_2^2x_3^2x_4^{81} \\ & + 24x_1x_2^2x_3^2x_4^{82} - 64x_1x_2^2x_3^2x_4^{83} - 16x_1x_2^2x_3^2x_4^{84} + 24x_1x_2^2x_3^2x_4^{85} - 24x_1x_2^2x_3^2x_4^{86} \\ & - 24x_1x_2^2x_3^2x_4^{87} - 64x_1x_2^2x_3^2x_4^{88} - 16x_1x_2^2x_3^2x_4^{89} + 24x_1x_2^2x_3^2x_4^{90} - 24x_1x_2^2x_3^2x_4^{91} \\ & + 24x_1x_2^2x_3^2x_4^{92} - 64x_1x_2^2x_3^2x_4^{93} - 16x_1x_2^2x_3^2x_4^{94} + 24x_1x_2^2x_3^2x_4^{95} - 24x_1x_2^2x_3^2x_4^{96} \\ & - 24x_1x_2^2x_3^2x_4^{97} - 64x_1x_2^2x_3^2x_4^{98} - 16x_1x_2^2x_3^2x_4^{99} + 24x_1x_2^2x_3^2x_4^{100} - 24x_1x_2^2x_3^2x_4^{101} \\ & + 24x_1x_2^2x_3^2x_4^{102} - 64x_1x_2^2x_3^2x_4^{103} - 16x_1x_2^2x_3^2x_4^{104} + 24x_1x_2^2x_3^2x_4^{105} - 24x_1x_2^2x_3^2x_4^{106} \\ & - 24x_1x_2^2x_3^2x_4^{107} - 64x_1x_2^2x_3^2x_4^{108} - 16x_1x_2^2x_3^2x_4^{109} + 24x_1x_2^2x_3^2x_4^{110} - 24x_1x_2^2x_3^2x_4^{111} \\ & + 24x_1x_2^2x_3^2x_4^{112} - 64x_1x_2^2x_3^2x_4^{113} - 16x_1x_2^2x_3^2x_4^{114} + 24x_1x_2^2x_3^2x_4^{115} - 24x_1x_2^2x_3^2x_4^{116} \\ & - 24x_1x_2^2x_3^2x_4^{117} - 64x_1x_2^2x_3^2x_4^{118} - 16x_1x_2^2x_3^2x_4^{119} + 24x_1x_2^2x_3^2x_4^{120} - 24x_1x_2^2x_3^2x_4^{121} \\ & + 24x_1x_2^2x_3^2x_4^{122} - 64x_1x_2^2x_3^2x_4^{123} - 16x_1x_2^2x_3^2x_4^{124} + 24x_1x_2^2x_3^2x_4^{125} - 24x_1x_2^2x_3^2x_4^{126} \\ & - 24x_1x_2^2x_3^2x_4^{127} - 64x_1x_2^2x_3^2x_4^{128} - 16x_1x_2^2x_3^2x_4^{129} + 24x_1x_2^2x_3^2x_4^{130} - 24x_1x_2^2x_3^2x_4^{131} \\ & + 24x_1x_2^2x_3^2x_4^{132} - 64x_1x_2^2x_3^2x_4^{133} - 16x_1x_2^2x_3^2x_4^{134} + 24x_1x_2^2x_3^2x_4^{135} - 24x_1x_2^2x_3^2x_4^{136} \\ & - 24x_1x_2^2x_3^2x_4^{137} - 64x_1x_2^2x_3^2x_4^{138} - 16x_1x_2^2x_3^2x_4^{139} + 24x_1x_2^2x_3^2x_4^{140} - 24x_1x_2^2x_3^2x_4^{141} \\ & + 24x_1x_2^2x_3^2x_4^{142} - 64x_1x_2^2x_3^2x_4^{143} - 16x_1x_2^2x_3^2x_4^{144} + 24x_1x_2^2x_3^2x_4^{145} - 24x_1x_2^2x_3^2x_4^{146} \\ & - 24x_1x_2^2x_3^2x_4^{147} - 64x_1x_2^2x_3^2x_4^{148} - 16x_1x_2^2x_3^2x_4^{149} + 24x_1x_2^2x_3^2x_4^{150} - 24x_1x_2^2x_3^2x_4^{151} \\ & + 24x_1x_2^2x_3^2x_4^{152} - 64x_1x_2^2x_3^2x_4^{153} - 16x_1x_2^2x_3^2x_4^{154} + 24x_1x_2^2x_3^2x_4^{155} - 24x_1x_2^2x_3^2x_4^{156} \\ & - 24x_1x_2^2x_3^2x_4^{157} - 64x_1x_2^2x_3^2x_4^{158} - 16x_1x_2^2x_3^2x_4^{159} + 24x_1x_2^2x_3^2x_4^{160} - 24x_1x_2^2x_3^2x_4^{161} \\ & + 24x_1x_2^2x_3^2x_4^{162} - 64x_1x_2^2x_3^2x_4^{163} - 16x_1x_2^2x_3^2x_4^{164} + 24x_1x_2^2x_3^2x_4^{165} - 24x_1x_2^2x_3^2x_4^{166} \\ & - 24x_1x_2^2x_3^2x_4^{167} - 64x_1x_2^2x_3^2x_4^{168} - 16x_1x_2^2x_3^2x_4^{169} + 24x_1x_2^2x_3^2x_4^{170} - 24x_1x_2^2x_3^2x_4^{171} \\ & + 24x_1x_2^2x_3^2x_4^{172} - 64x_1x_2^2x_3^2x_4^{173} - 16x_1x_2^2x_3^2x_4^{174} + 24x_1x_2^2x_3^2x_4^{175} - 24x_1x_2^2x_3^2x_4^{176} \\ & - 24x_1x_2^2x_3^2x_4^{177} - 64x_1x_2^2x_3^2x_4^{178} - 16x_1x_2^2x_3^2x_4^{179} + 24x_1x_2^2x_3^2x_4^{180} - 24x_1x_2^2x_3^2x_4^{181} \\ & + 24x_1x_2^2x_3^2x_4^{182} - 64x_1x_2^2x_3^2x_4^{183} - 16x_1x_2^2x_3^2x_4^{184} + 24x_1x_2^2x_3^2x_4^{185} - 24x_1x_2^2x_3^2x_4^{186} \\ & - 24x_1x_2^2x_3^2x_4^{187} - 64x_1x_2^2x_3^2x_4^{188} - 16x_1x_2^2x_3^2x_4^{189} + 24x_1x_2^2x_3^2x_4^{190} - 24x_1x_2^2x_3^2x_4^{191} \\ & + 24x_1x_2^2x_3^2x_4^{192} - 64x_1x_2^2x_3^2x_4^{193} - 16x_1x_2^2x_3^2x_4^{194} + 24x_1x_2^2x_3^2x_4^{195} - 24x_1x_2^2x_3^2x_4^{196} \\ & - 24x_1x_2^2x_3^2x_4^{197} - 64x_1x_2^2x_3^2x_4^{198} - 16x_1x_2^2x_3^2x_4^{199} + 24x_1x_2^2x_3^2x_4^{200} - 24x_1x_2^2x_3^2x_4^{201} \\ & + 24x_1x_2^2x_3^2x_4^{202} - 64x_1x_2^2x_3^2x_4^{203} - 16x_1x_2^2x_3^2x_4^{204} + 24x_1x_2^2x_3^2x_4^{205} - 24x_1x_2^2x_3^2x_4^{206} \\ & - 24x_1x_2^2x_3^2x_4^{207} - 64x_1x_2^2x_3^2x_4^{208} - 16x_1x_2^2x_3^2x_4^{209} + 24x_1x_2^2x_3^2x_4^{210} - 24x_1x_2^2x_3^2x_4^{211} \\ & + 24x_1x_2^2x_3^2x_4^{212} - 64x_1x_2^2x_3^2x_4^{213} - 16x_1x_2^2x_3^2x_4^{214} + 24x_1x_2^2x_3^2x_4^{215} - 24x_1x_2^2x_3^2x_4^{216} \\ & - 24x_1x_2^2x_3^2x_4^{217} - 64x_1x_2^2x_3^2x_4^{218} - 16x_1x_2^2x_3^2x_4^{219} + 24x_1x_2^2x_3^2x_4^{220} - 24x_1x_2^2x_3^2x_4^{221} \\ & + 24x_1x_2^2x_3^2x_4^{222} - 64x_1x_2^2x_3^2x_4^{223} - 16x_1x_2^2x_3^2x_4^{224} + 24x_1x_2^2x_3^2x_4^{225} - 24x_1x_2^2x_3^2x_4^{226} \\ & - 24x_1x_2^2x_3^2x_4^{227} - 64x_1x_2^2x_3^2x_4^{228} - 16x_1x_2^2x_3^2x_4^{229} + 24x_1x_2^2x_3^2x_4^{230} - 24x_1x_2^2x_3^2x_4^{231} \\ & + 24x_1x_2^2x_3^2x_4^{232} - 64x_1x_2^2x_3^2x_4^{233} - 16x_1x_2^2x_3^2x_4^{234} + 24x_1x_2^2x_3^2x_4^{235} - 24x_1x_2^2x_3^2x_4^{236} \\ & - 24x_1x_2^2x_3^2x_4^{237} - 64x_1x_2^2x_3^2x_4^{238} - 16x_1x_2^2x_3^2x_4^{239} + 24x_1x_2^2x_3^2x_4^{240} - 24x_1x_2^2x_3^2x_4^{241} \\ & + 24x_1x_2^2x_3^2x_4^{242} - 64x_1x_2^2x_3^2x_4^{243} - 16x_1x_2^2x_3^2x_4^{244} + 24x_1x_2^2x_3^2x_4^{245} - 24x_1x_2^2x_3^2x_4^{246} \\ & - 24x_1x_2^2x_3^2x_4^{247} - 64x_1x_2^2x_3^2x_4^{248} - 16x_1x_2^2x_3^2x_4^{249} + 24x_1x_2^2x_3^2x_4^{250} - 24x_1x_2^2x_3^2x_4^{251} \\ & + 24x_1x_2^2x_3^2x_4^{252} - 64x_1x_2^2x_3^2x_4^{253} - 16x_1x_2^2x_3^2x_4^{254} + 24x_1x_2^2x_3^2x_4^{255} - 24x_1x_2^2x_3^2x_4^{256} \\ & - 24x_1x_2^2x_3^2x_4^{257} - 64x_1x_2^2x_3^2x_4^{258} - 16x_1x_2^2x_3^2x_4^{259} + 24x_1x_2^2x_3^2x_4^{260} - 24x_1x_2^2x_3^2x_4^{261} \\ & + 24x_1x_2^2x_3^2x_4^{262} - 64x_1x_2^2x_3^2x_4^{263} - 16x_1x_2^2x_3^2x_4^{264} + 24x_1x_2^2x_3^2x_4^{265} - 24x_1x_2^2x_3^2x_4^{266} \\ & - 24x_1x_2^2x_3^2x_4^{267} - 64x_1x_2^2x_3^2x_4^{268} - 16x_1x_2^2x_3^2x_4^{269} + 24x_1x_2^2x_3^2x_4^{270} - 24x_1x_2^2x_3^2x_4^{271} \\ & + 24x_1x_2^2x_3^2x_4^{272} - 64x_1x_2^2x_3^2x_4^{273} - 16x_1x_2^2x_3^2x_4^{274} + 24x_1x_2^2x_3^2x_4^{275} - 24x_1x_2^2x_3^2x_4^{276} \\ & - 24x_1x_2^2x_3^2x_4^{277} - 64x_1x_2^2x_3^2x_4^{278} - 16x_1x_2^2x_3^2x_4^{279} + 24x_1x_2^2x_3^2x_4^{280} - 24x_1x_2^2x_3^2x_4^{281} \\ & + 24x_1x_2^2x_3^2x_4^{282} - 64x_1x_2^2x_3^2x_4^{283} - 16x_1x_2^2x_3^2x_4^{284} + 24x_1x_2^2x_3^2x_4^{285} - 24x_1x_2^2x_3^2x_4^{286} \\ & - 24x_1x_2^2x_3^2x_4^{287} - 64x_1x_2^2x_3^2x_4^{288} - 16x_1x_2^2x_3^2x_4^{289} + 24x_1x_2^2x_3^2x_4^{290} - 24x_1x_2^2x_3^2x_4^{291} \\ & + 24x_1x_2^2x_3^2x$$

pfd-parallel

Massively-Parallel
Partial Fraction Decomposition

a massively parallel framework for partial fraction decomposition of rational functions
based on the **Singular/GPI-Space framework**

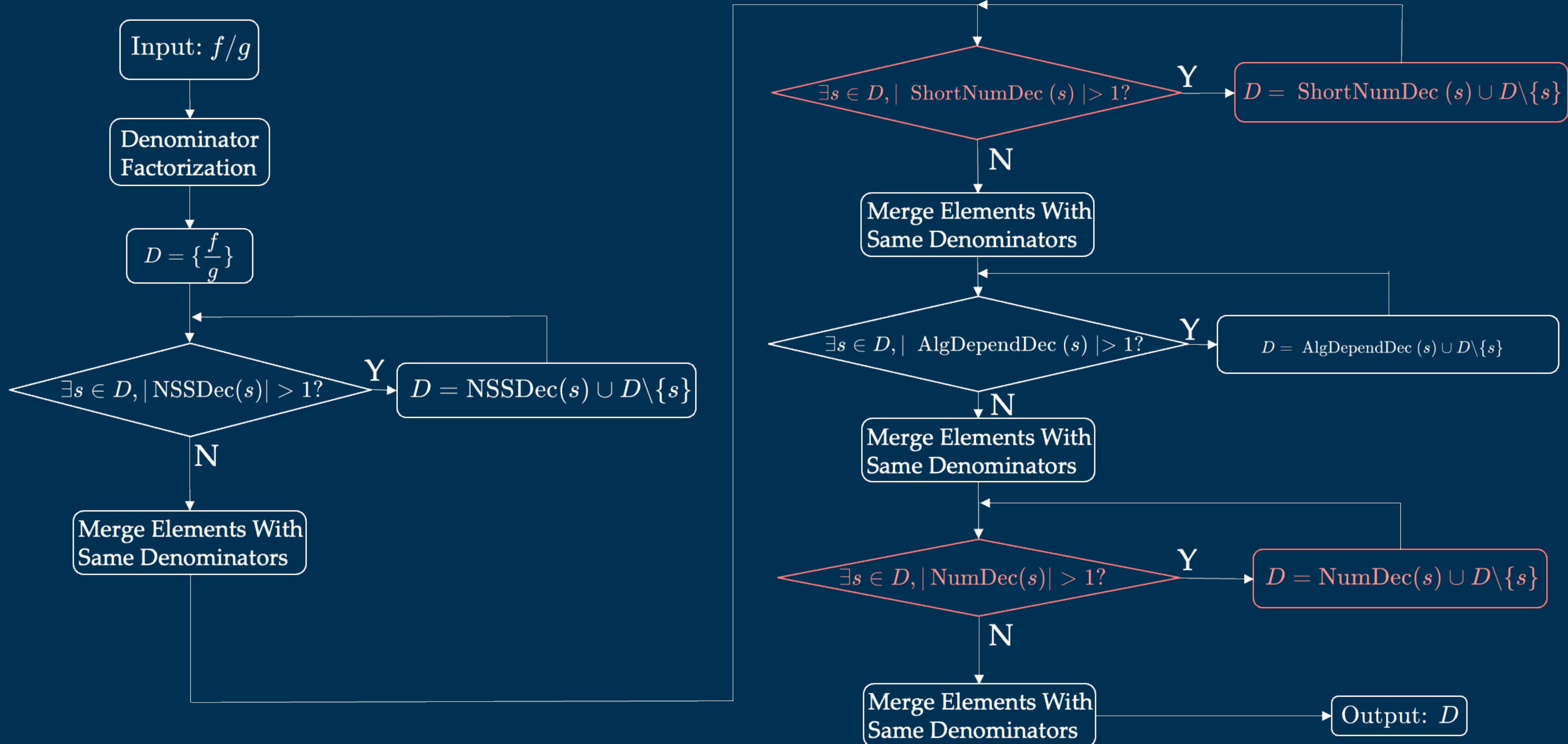
Improved Leintars algorithm

Boehm, Wittmann, Wu, Xu, and YZ, JHEP 12 (2020) 054

Heller and von Manteuffel: CPC. 271 (2022) 108174.

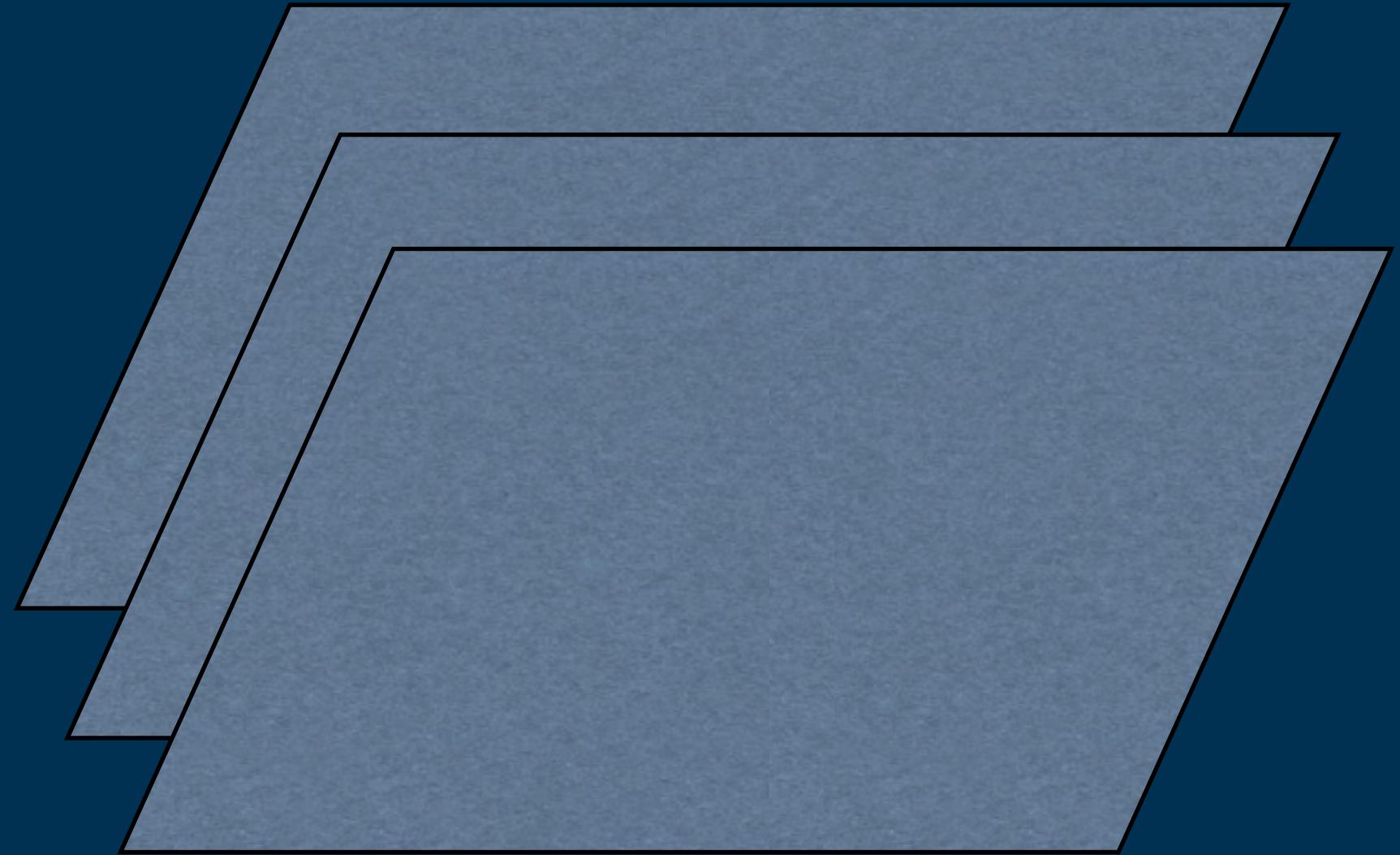
MultivariateApart algorithm

Improved Leintars' algorithm



pfd-parallel

Massively-Parallel
Partial Fraction Decomposition



parallelization between several rational functions
parallelization between terms in one rational function

Partial Fraction Decomposition can be a heavy computation,
so parallelization is important

We implement **several layers** of parallelization

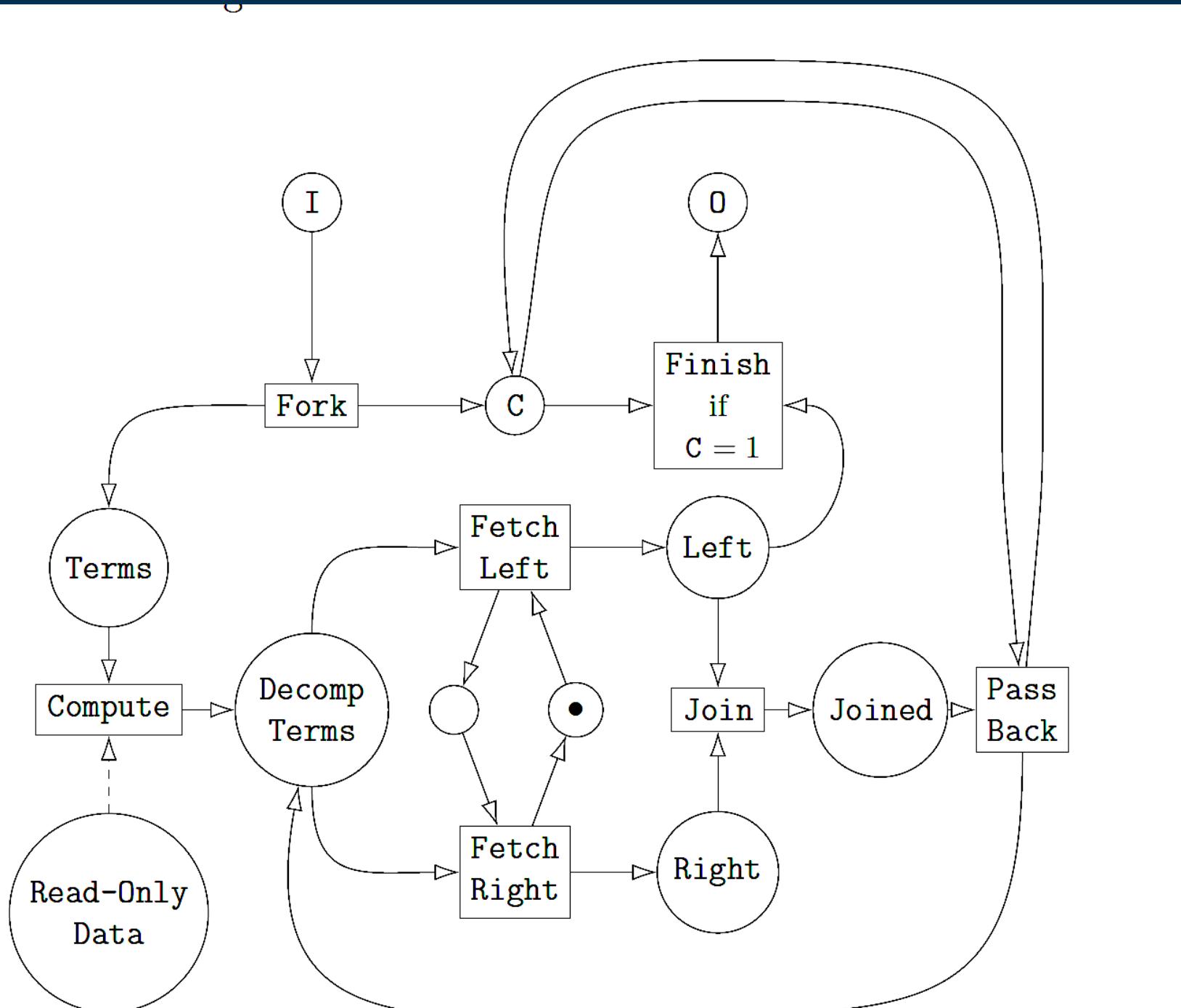
complicated workflow
so we implement the
workflow manager
GPI-space

pfd-parallel

Massively-Parallel
Partial Fraction Decomposition

GPI-Space, a workflow management system,
created by the Competence Center High Performance Computing of Fraunhofer ITWM.

Combine **Singular** with GPI-Space
to create a way of performing massively parallel computations in computer algebra.



<https://www.mathematik.uni-kl.de/~boehm/singulargpispace/>

For details,
refer to Janko Boehm's talk and his website

Petri-Net for the parallelization over
the decomposition of terms

pfd-parallel Example

Simplifying coefficients in multi-loop scattering amplitudes

Two-loop leading color $pp \rightarrow W^\pm + \gamma + j$ from Badger, Hartanto, Krys, Zoia
JHEP 05 (2022) 035

6 kinematic variables



with MultivariateApart algorithm
implemented in GPI-space/Singular

Running on a node with 48 CPUs, ~ 5.5 hours

NeatIBP

A package generates small size
IBP systems based on
syzygy/module intersection

Wu, Boehm, Ma, Xu and YZ

<https://github.com/yzhphy/NeatIBP>

arXiv: 2303. ***
to be submitted to Comp. Phys. Comm.

It is well know that IBP reduction could be
the slowest step for an amplitude computation

$$\int \frac{d^D l_1}{i\pi^{D/2}} \cdots \int \frac{d^D l_L}{i\pi^{D/2}} \frac{\partial}{\partial l_i^\mu} \frac{v_i^\mu}{D_1^{\alpha_1} \dots D_k^{\alpha_k}} = 0 \quad \text{Chetyrkin, Tkachov (1981)}$$

Standard Laporta algorithm may provide a system containing
too many integrals other than target or master integrals.

Gaussian elimination with symbolic parameter is
extremely difficult

Many way to improve the situation ...

other than the standard Laporta algorithm

Non-commutative

algebra

to avoid Gaussian elimination by
a reduction over the integral index operators

Smirnov, Smirnov , JHEP01(2006)001

...

Syzgyy

to decrease the number of integrals
in an IBP system

Gluza, Kajda, Kosower, PRD. 83.045012

Schabinger, JHEP 01 (2012) 077

Larsen YZ, PRD 93 (2016) 4, 041701 ...

For more details about syzygy IBP,
refer to Bakul's talk

Intersection

theory

to map the space of Feynman integrals
to a twisted cohomology group

Mastrolia, Mizera JHEP 02 (2019) 139

Frellesvig, Gasparotto, Laporta, Mandal, Mastrolia, Mizera JHEP 05 (2019) 153

Frellesvig, Gasparotto, Mandal, Mastrolia, Mattiazzi, Mizera PRL. 123.201602

...

Auxiliary mass

flow

to find integrals relations
based on the eta expansion

Liu, Ma Phys. Rev. D 99 (2019), no 7 071501

Guan, Liu, Ma, Chin. Phys. C 44(2020) 9, 093106

...

other than the standard Laporta algorithm

Non-commutative

algebra

to avoid Gaussian elimination by
a reduction over the integral index operators

Smirnov, Smirnov , JHEP01(2006)001

...

Syzygy

to decrease the number of integrals
in an IBP system

Gluza, Kajda, Kosower, PRD. 83.045012

Schabinger, JHEP 01 (2012) 077

Larsen YZ, PRD 93 (2016) 4, 041701 ...

For more details about syzygy IBP,
refer to Bakul's talk

Intersection

theory

to map the space of Feynman integrals
to a twisted cohomology group

Mastrolia, Mizera JHEP 02 (2019) 139

Frellesvig, Gasparotto, Laporta, Mandal, Mastrolia, Mizera JHEP 05 (2019) 153

Frellesvig, Gasparotto, Mandal, Mastrolia, Mattiazzi, Mizera PRL. 123.201602

...

Auxiliary mass

flow

to find integrals relations
based on the eta expansion

Liu, Ma Phys. Rev. D 99 (2019), no 7 071501

Guan, Liu, Ma, Chin. Phys. C 44(2020) 9, 093106

...

Syzygy for IBP reduction

smart vector choice

$$\int \frac{d^D l_1}{i\pi^{D/2}} \cdots \int \frac{d^D l_L}{i\pi^{D/2}} \frac{\partial}{\partial l_i^\mu} \frac{v_i^\mu}{D_1^{\alpha_1} \dots D_k^{\alpha_k}} = 0$$

Suppose that we want to forbid the increase of the propagator index α_i , we can require that,

$$\left(\sum_{j=1}^L v_j^\mu \frac{\partial}{\partial l_j^\mu} D_i \right) + g_i D_i = 0$$

Syzygy equation

Gluza, Kajda, Kosower,
PhysRevD. 83.045012

where both v_j^μ and g_i contain polynomials in loop momenta.

ways to apply syzygy method

Syzygy from Linear Algebra, Schabinger, *JHEP* 01 (2012) 077

Module Intersection, Larsen, YZ, *Phys.Rev.D* 93 (2016) 4, 041701

Dual Conformal Symmetry, Bern, Enciso, Ita, Zeng, *Phys. Rev. D* 96, 096017 (2017)

....

Syzygy for IBP reduction

smart vector choice

$$\int \frac{d^D l_1}{i\pi^{D/2}} \cdots \int \frac{d^D l_L}{i\pi^{D/2}} \frac{\partial}{\partial l_i^\mu} \frac{v_i^\mu}{D_1^{\alpha_1} \cdots D_k^{\alpha_k}} = 0$$

Suppose that we want to forbid the increase of the propagator index α_i , we can require that,

$$\left(\sum_{j=1}^L v_j^\mu \frac{\partial}{\partial l_j^\mu} D_i \right) + g_i D_i = 0$$

Syzygy equation

Gluza, Kajda, Kosower,
PhysRevD. 83.045012

where both v_j^μ and g_i contain polynomials in loop momenta.

ways to apply syzygy method

Syzygy from Linear Algebra, Schabinger, *JHEP* 01 (2012) 077

Module Intersection, Larsen, YZ, *Phys.Rev.D* 93 (2016) 4, 041701

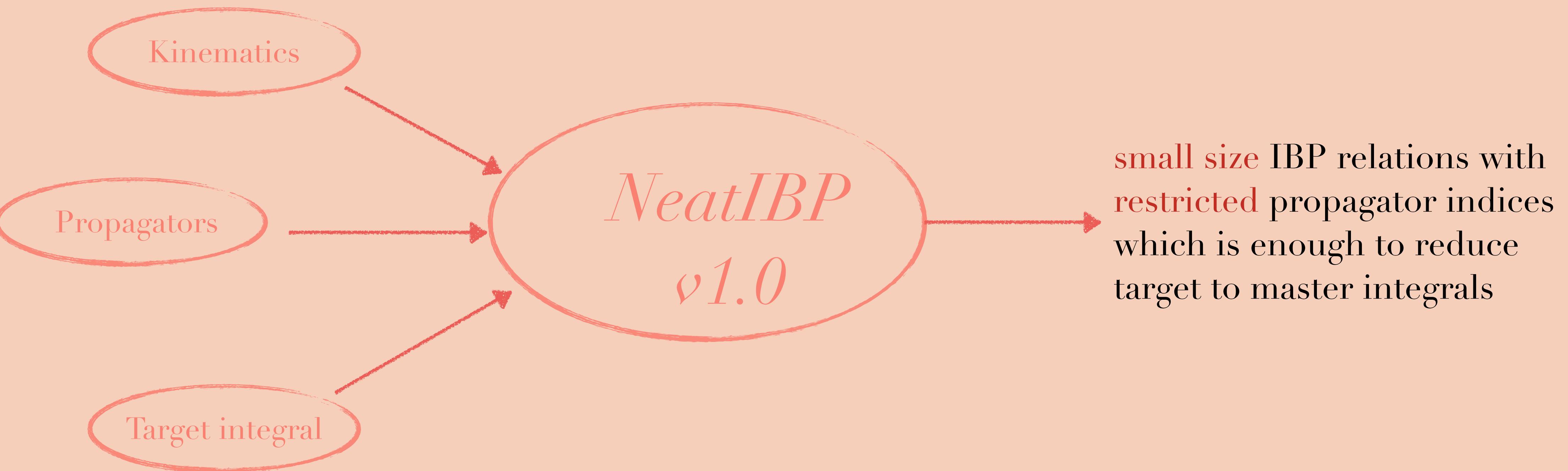
Dual Conformal Symmetry, Bern, Enciso, Ita, Zeng, *Phys. Rev. D* 96, 096017 (2017)

....

NeatIBP v1.0

an automatic package to generate small-size IBP
based on syzygy/module intersection

Boehm, Ma, Wu, Xu and YZ

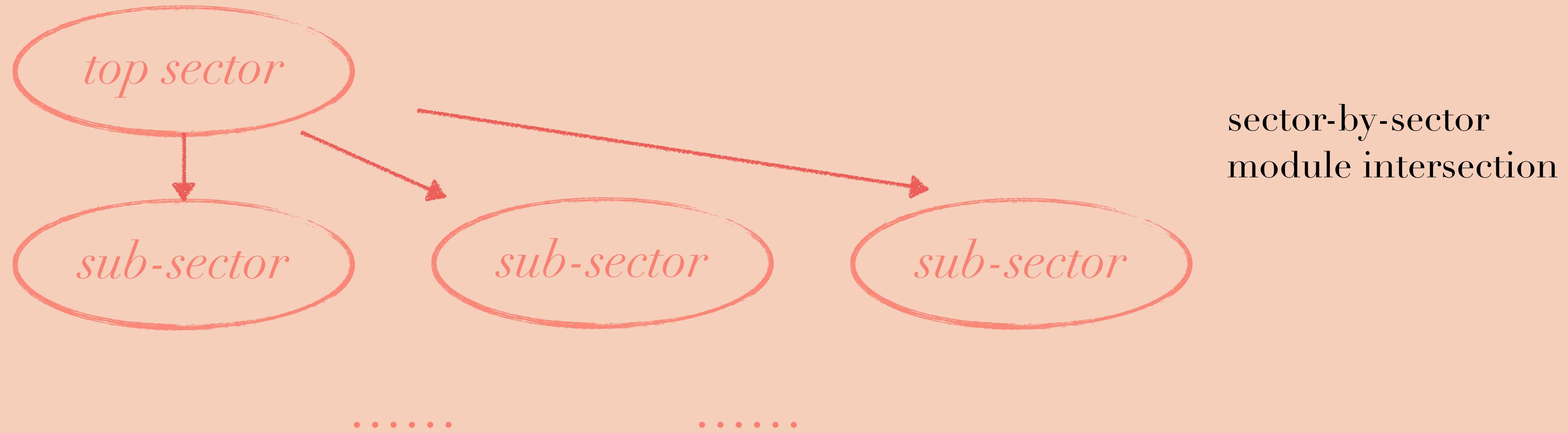


<https://github.com/yzhphy/NeatIBP>

development version
welcome to download and try

NeatIBP v1.0

an automatic package to generate small-size IBP
based on syzygy/module intersection



generate IBPs with propagator indices restricted
finite-field method to select useful and independent IBP relations

Determine the master integrals and accumulate IBP
relations in the same time

NeatIBP v1.0 Implementation notes

Mathematica/Singular codes module intersection is done
with the Schreyer's algorithm in Singular

Parallelization is over different sectors

Finite Field Computations

Unitarity cut applicable

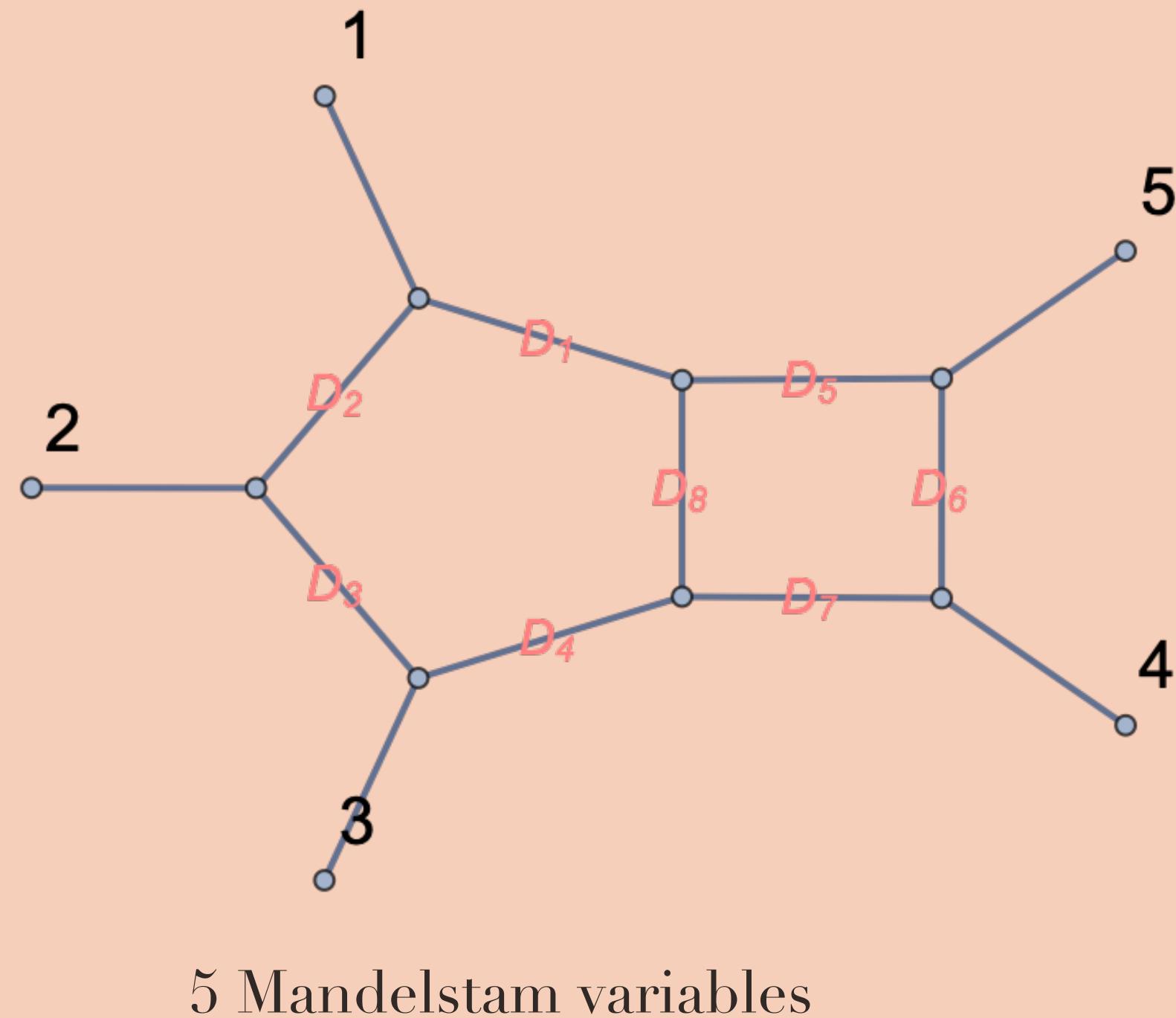
to select useful and independent IBPs
powered by the package SpaSM
<https://github.com/cbouilla/spasm>
by Charles Bouillaguet

Symmetry relation implemented

NeatIBP v1.0

Example 1

Massless pentagon-box



on a workstation with 10 cores
NeatIBP running time ~21 minutes

2503 target integrals without double propagators
(hardest ones with deg-5 numerator)
12694 IBP relations without double propagators
enough

short and sparse IBP system
with integer-valued Mandelstam variables
Finite-field reduction < 1 second

Look at the input files

```
(*-----input files-----*)
kinematicsFile = workingPath<>"kinematics.txt";(*workingPath will be set as the current working path*)
targetIntegralsFile = workingPath<>"targetIntegrals.txt";

(*-----dependency settings-----*)
SingularApp = "/usr/bin/Singular";

(*-----directory settings-----*)
ReductionOutputName="pb1";
outputPath=Automatic;(*recommended*)

(*outputPath=workingPath<>"test/";*)(*alternative*)

(*-----math settings-----*)
IntegralOrder = "Global"
OptionSimplification = 12;
NeedSymmetry=True;

(*-----usage settings-----*)
MemoryUsedLimit=100000;(*MB*)
ThreadUsedLimit=20;
```

config.txt

NeatIBP v1.0

Example 1

Massless pentagon-box

Look at the input files

```
LoopMomenta={l1,l2};  
ExternalMomenta={k1,k2,k3,k4};  
Propagators=#^2&/@{l1,l1-k1,l1-k1-k2,l1-k1-k2-k3,l2,l2+k1+k2+k3+k4,l2+k1+k2+k3,l1+l2,l1-k1-k2-k3-k4,l2+k1,l2+k2};  
Kinematics={k1^2 -> 0, k2^2 -> 0, k3^2 -> 0, k4^2 -> 0, k1 k2 -> s12/2,  
k1 k3 -> 1/2 (-s12 - s23 + s45), k1 k4 -> 1/2 (-s15 + s23 - s45),  
k2 k3 -> s23/2, k2 k4 -> 1/2 (s15 - s23 - s34), k3 k4 -> s34/2};  
GenericPoint={s12 -> 13, s23 -> 2, s34 -> 73, s45 -> 7, s15 -> 213};
```

kinematics.txt

```
{G[1, 1, 1, 1, 1, 1, 1, 0, 0, -5], G[1, 1, 1, 1, 1, 1, 1, 0, -1, -2], G[1, 1, 1, 1, 1, 1, 1, 0, -2, -1],  
G[1, 1, 1, 1, 1, 1, 1, 0, -5, 0], G[1, 1, 1, 1, 1, 1, 1, -1, 0, -3], G[1, 1, 1, 1, 1, 1, 1, -1, -1, -4],  
G[1, 1, 1, 1, 1, 1, 1, -1, -2, -1], G[1, 1, 1, 1, 1, 1, 1, -1, -3, 0], G[1, 1, 1, 1, 1, 1, 1, -1, -4, 0],  
G[1, 1, 1, 1, 1, 1, 1, -2, 0, -1], G[1, 1, 1, 1, 1, 1, 1, -2, -1, -1], G[1, 1, 1, 1, 1, 1, 1, -3, 0, 0],  
G[1, 1, 1, 1, 1, 1, 1, -3, 0, -1], G[1, 1, 1, 1, 1, 1, 1, -3, -2, 0], G[1, 1, 1, 1, 1, 1, 1, -4, 0, -1],  
G[1, 1, 1, 1, 1, 1, 1, -4, -1, 0], G[1, 1, 1, 1, 1, 1, 1, -5, 0, 0], G[1, 1, 1, 1, 1, 1, 1, 0, 0, -4],  
G[1, 1, 1, 1, 1, 0, 1, 0, 0, -3], G[1, 1, 1, 1, 1, 1, 0, 1, 0, -2, -1], G[1, 1, 1, 1, 1, 1, 0, 1, -1, -3, 0],  
G[1, 1, 1, 1, 1, 0, 1, -2, 0, -2], G[1, 1, 1, 1, 1, 1, 0, 1, -3, 0, -1], G[1, 1, 1, 1, 1, 1, 0, 1, -4, 0, -1],  
G[1, 1, 1, 1, 1, 0, 1, -4, -1, 0], G[1, 1, 1, 1, 1, 1, 1, -1, 1, -2, 0, 0], G[1, 1, 1, 1, 1, 1, 1, -2, 1, 0, -1, 0],  
G[1, 1, 1, 1, 1, 1, -3, 1, -1, 0, -1], G[1, 1, 1, 1, 1, 0, 1, 1, 0, 0, -1], G[1, 1, 1, 1, 1, 1, 0, 1, 1, 0, 0, -3],  
G[1, 1, 1, 1, 1, 0, 1, 1, 0, 0, -4], G[1, 1, 1, 1, 1, 0, 1, 1, -1, -1, 0], G[1, 1, 1, 1, 1, 0, 1, 1, -1, -2, -2],  
G[1, 1, 1, 1, 1, 0, 1, 1, 1, -2, 0, 0], G[1, 1, 1, 1, 1, 0, 1, 1, -2, 0, -3], G[1, 1, 1, 1, 1, 0, 1, 1, -2, -2, 0],  
G[1, 1, 1, 1, 1, 0, 1, 1, -3, 0, 0], G[1, 1, 1, 1, 1, 0, 1, 1, -3, 0, -2], ... ...}
```

target.txt

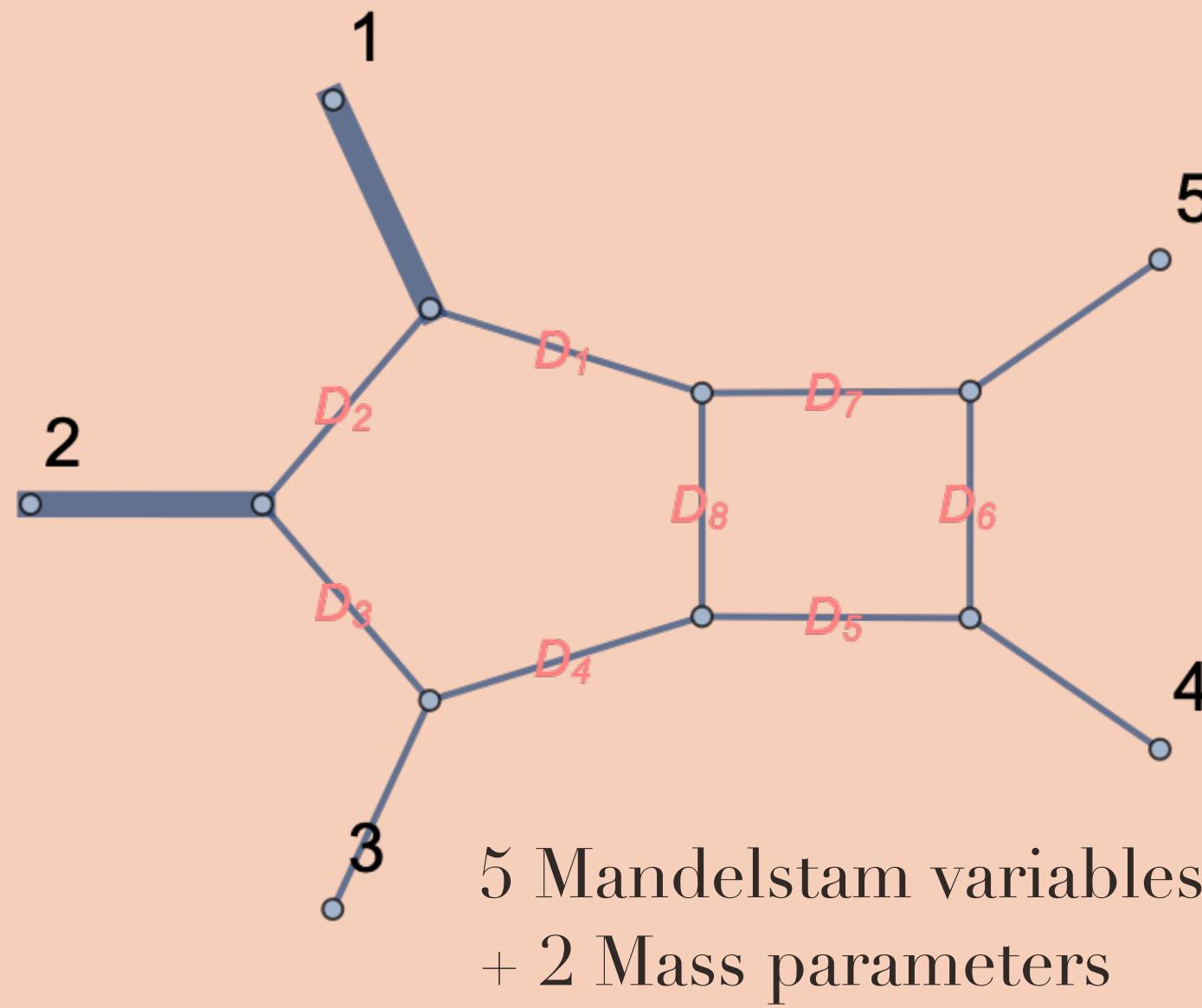
NeatIBP v1.0 Example 1

then in terminal enter

./run.sh

NeatIBP v1.0

Example 2 pentagon-box with two masses



a difficult problem

Impose a spanning cut

$$D_2 = D_5 = D_7 = 0$$

top sector target integrals with numerator degree 5



6142 IBP relations without double propagators
enough

on a workstation with 10 cores
NeatIBP running time ~30 minutes

NeatIBP v1.0

Example 2 pentagon-box with two masses

```
(*-----input files-----*)
kinematicsFile = workingPath<>"kinematics.txt";(*workingPath will be set as the current working path*)
targetIntegralsFile = workingPath<>"targetIntegrals.txt";

(*-----dependency settings-----*)
SingularApp = "/usr/bin/Singular";
SparseRREF`SpaSMLibrary = "/usr/local/lib/libspasm.so"

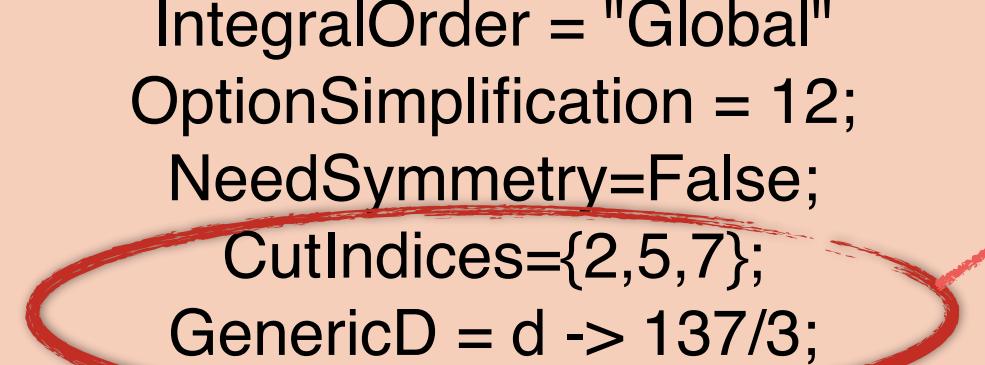
(*-----directory settings-----*)
ReductionOutputName="pb2m257cut";
outputPath=Automatic;(*recommended*)

(*outputPath=workingPath<>"test/";*)(*alternative*)

(*-----math settings-----*)
IntegralOrder = "Global"
OptionSimplification = 12;
NeedSymmetry=False;
CutIndices={2,5,7};
GenericD = d -> 137/3;

(*-----usage settings-----*)
MemoryUsedLimit=100000;(*MB*)
ThreadUsedLimit=18;
```

Apply a
spanning cut



CutIndices={2,5,7};

$$D_2 = D_5 = D_7 = 0$$

config.txt

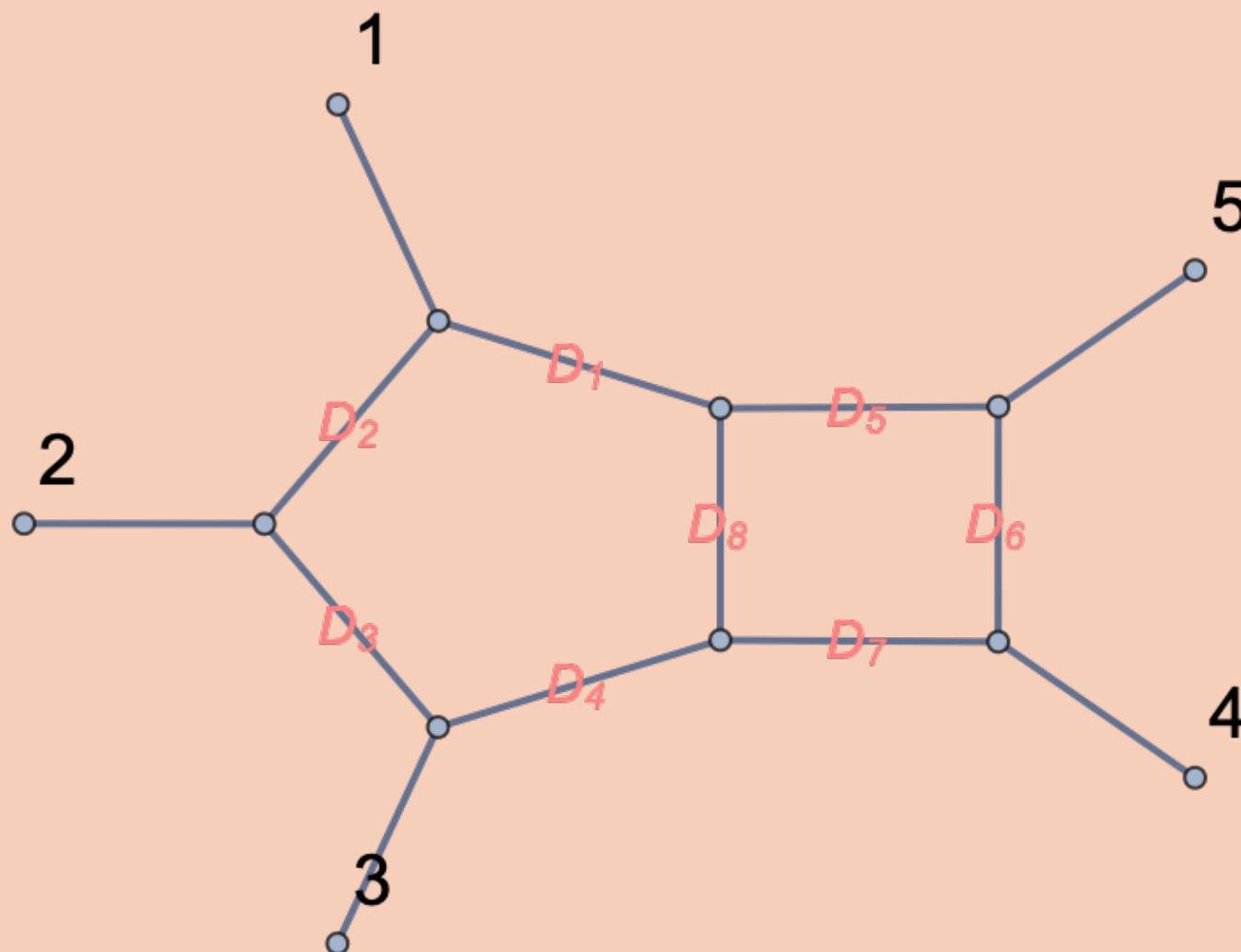
NeatIBP v1.0

Example 3 with double propagators

Slogan said: “syzygy IBP method does not apply for integrals with double propagators”

This is wrong.

syzygy IBP method **does** apply for integrals with double propagators



61 master integrals' derivatives in Mandelstam variables

880 target integrals; most of them with double propagators



3374 IBP relations, enough

on a workstation with 10 cores

NeatIBP running time ~51 minutes

NeatIBP v1.0

Issues and outlook

- NeatIBP v1.0 uses Mathematica quite a lot. In future, we will focus on open-source Languages.
- NeatIBP v1.0 uses Singular's Schreyer algorithm. We have been testing linear algebra syzygy method for a long time, but not used in this version ... *looking forward to Bakul's talk*
- The IBP seeding for the syzygy method has many possibilities . It seems that there are a lot of room for tuning ...
- We will apply the massive parallelization framework **GPI-space** for the future NeatIBP versions

Summary

pfd-parallel and *NeatIBP*

Based on algebraic geometry

aims at cutting-edge computation of Feynman integrals

various updates of the two packages will appear in this summer

Looking forward to meet you in Europe this summer!

IBP in Baikov representation with constraints

Require

1. no shifted exponent:

$$\sum_{j=1}^k a_j(z) \frac{\partial F}{\partial z_j} + \beta(z)F = 0$$

2. no propagator degree increase:

$$a_i(z) \in \langle z_i \rangle, \quad 1 \leq i \leq m$$

polynomials

These $(a_1(z), \dots, a_k(z))$ form a module $M_1 \subset R^k$.

These $(a_1(z), \dots, a_k(z))$ form a module $M_2 \subset R^k$.

Both M_1 and M_2 are pretty simple ...

Larsen YZ

Phys.Rev.D 93 (2016) 4, 041701

$$M_1 \cap M_2$$

Intersection of two modules