New Features in pySecDec

Stephan Jahn[1]

[1]Max-Planck-Institut für Physik, München

March 20, 2018

[1]sjahn@mpp.mpg.de

pySecDec

successor of SecDec-3

S. Borowka, G. Heinrich, S. P. Jones, M. Kerner, J. Schlenk, T. Zirke [1502.06595]

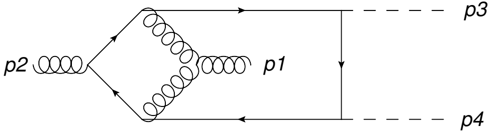
Numerically computes regulated parameter integrals of the form

$$\mathcal{I} \equiv \int_0^1 \mathrm{d}x_1 \ldots \int_0^1 \mathrm{d}x_N \prod_{i=1}^{m} f_i\left(\vec{x}, \vec{a}\right)^{b_i + \sum_k c_{ik}\epsilon_k}$$

where the f_i are polynomials. Typically: $\mathcal{I}|_{\epsilon_k=0} = \infty$.

Example:
Loop integrals

after Feynman parametrization



The SecDec collaboration

Sophia Borowka

Gudrun Heinrich

Stephan Jahn

Stephen Jones

Matthias Kerner

Johannes Schlenk

former members

Thomas Binoth

Jonathon Carter

Tom Zirke

Paper

[1703.09692] published in CPC

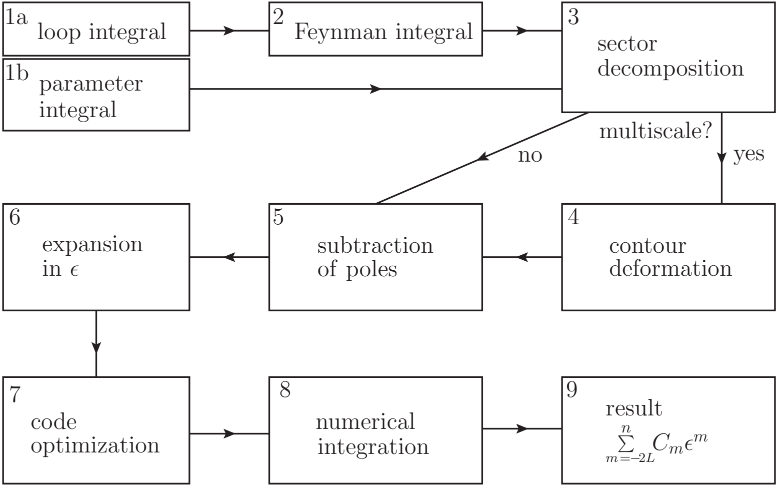
Homepage

`http://secdec.hepforge.org/`

Other Public Implementations

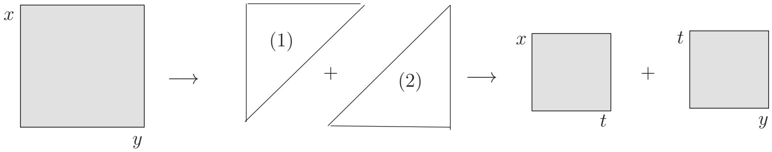
- *C. Bogner, S. Weinzierl*
  Sector decomposition
  [0709.4092]
- *A.V. Smirnov*
  FIESTA 4
  [1511.03614]

Flowchart


1a loop integral
2 Feynman integral
3 sector decomposition
1b parameter integral
multiscale?
no
yes
6 expansion in ϵ
5 subtraction of poles
4 contour deformation
7 code optimization
8 numerical integration
9 result $\sum_{m=-2L}^{n} C_m \epsilon^m$

Sector Decomposition

or: Resolution of Overlapping Singularities



$$\int_0^1 dx \int_0^1 dy \; (x+y)^{a+b\epsilon} \, f(x,y)$$

$$= \int_0^1 dx \int_0^1 dy \; (x+y)^{a+b\epsilon} \, f(x,y) [\underbrace{\Theta(x-y)}_{(1)} + \underbrace{\Theta(y-x)}_{(2)}]$$

$$= \int_0^1 dx \int_0^1 dt \, x \, x^{a+b\epsilon} \, (1+t)^{a+b\epsilon} \, f(x,xt) + \int_0^1 dt \int_0^1 dy \, y \, y^{a+b\epsilon} \, (t+1)^{a+b\epsilon} \, f(yt,y)$$

Subtraction of Poles

$$\int_0^1 dt\, t^{-1+b\epsilon} g(t)$$

$$= \int_0^1 dt\, t^{-1+b\epsilon} \left(g(0) + g(t) - g(0)\right)$$

$$= \underbrace{\int_0^1 dt\, t^{-1+b\epsilon} g(0)}_{=\frac{1}{b\epsilon} g(0)} + \underbrace{\int_0^1 dt\, t^{-1+b\epsilon} \left(g(t) - g(0)\right)}_{\text{finite for } \epsilon \to 0 \text{, expand integrand in } \epsilon}$$

New Features in pySecDec

Basic Usage

$$\int_0^1 dx \int_0^1 dy (x+y)^{-2+\epsilon} = \frac{1}{\epsilon} + (1 - \log(2)) + O(\epsilon) \approx \frac{1}{\epsilon} + 0.306853 + O(\epsilon)$$

Step 1: write input files

generate_easy.py

```
from pySecDec import make_package

make_package(

name = 'easy',
integration_variables = ['x','y'],
regulators = ['eps'],

requested_orders = [0],
polynomials_to_decompose = ['(x+y)^(-2+eps)'],

)
```

integrate_easy.py

```
from pySecDec.integral_interface \
    import IntegralLibrary

# load c++ library
easy_integral = \
    IntegralLibrary('easy/easy_pylink.so')

# integrate
_, _, result = easy_integral()

# print result
print('Numerical Result:')
print(result)
```

Step 2: run pySecDec

```
$ python generate_easy.py && make -C easy && python integrate_easy.py
<skipped some output>
Numerical Result:
 + (1.00015897181235158e+00 +/- 4.03392522752491021e-03)*eps^-1 + (3.06903035514056399e-01 +/-
  ↪     2.82319349818329918e-03) + O(eps)
```

Symmetry Finder

$$\int_0^1 dx \int_0^1 dy \; (x+y)^{a+b\epsilon}$$

$$= \int_0^1 dx \int_0^1 dt \, x \, x^{a+b\epsilon} \, (1+t)^{a+b\epsilon} + \int_0^1 dt \int_0^1 dy \, y \, y^{a+b\epsilon} \, (t+1)^{a+b\epsilon}$$

same up to renaming $x \leftrightarrow y$

Symmetry Finder

represent polynomials as integer arrays

$$x + 2\,y = 1 \cdot x^1 \cdot y^0 + 2 \cdot x^0 \cdot y^1 \equiv \begin{array}{ccc} \text{coefficient} & x & y \\ \end{array} \left[\begin{array}{ccc} 1 & 1 & 0 \\ 2 & 0 & 1 \end{array}\right]$$

$$2\,x + y = 2 \cdot x^1 \cdot y^0 + 1 \cdot x^0 \cdot y^1 \equiv \begin{array}{ccc} \text{coefficient} & x & y \\ \end{array} \left[\begin{array}{ccc} 2 & 1 & 0 \\ 1 & 0 & 1 \end{array}\right]$$

Symmetry Finder

compare arrays allowing for row- and columnwise permutations

$$\begin{array}{c} \\ \text{term1} \\ \text{term2} \end{array} \begin{array}{c} \begin{array}{ccc} \text{coefficient} & x & y \end{array} \\ \left[\begin{array}{ccc} 2 & 1 & 0 \\ 1 & 0 & 1 \end{array}\right] \end{array} \qquad \begin{array}{c} \\ \text{term2} \\ \text{term1} \end{array} \begin{array}{c} \begin{array}{ccc} \text{coefficient} & y & x \end{array} \\ \left[\begin{array}{ccc} 1 & 1 & 0 \\ 2 & 0 & 1 \end{array}\right] \end{array}$$

$$2\,x + y \quad \xleftrightarrow{x \leftrightarrow y} \quad x + 2\,y$$

Symmetry Finder

- $n!$ possible column permutations with n variables
- two optimized algorithms implemented by Ben Ruijl and Stephen Jones:

  B. Ruijl, S. P. Jones [to appear in the proceedings of ACAT 2017]

  - based on permutations as suggested by Alexey Pak

    A. Pak [1111.0868]
  - based on finding graph isomorphisms with `dreadnaut`

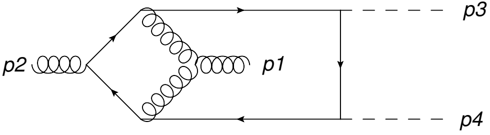
    B. D. McKay, A. Piperno [1301.1493]

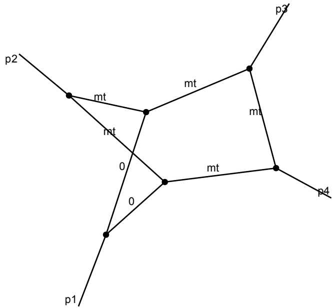
- outlook: identify matroid symmetries in Loopedia

  C. Bogner, S. Borowka, T. Hahn, G. Heinrich, S. P. Jones, M. Kerner, A. von Manteuffel, M. Michel, E. Panzer, V. Papara [1709.01266]

Higgs Boson Pair Production

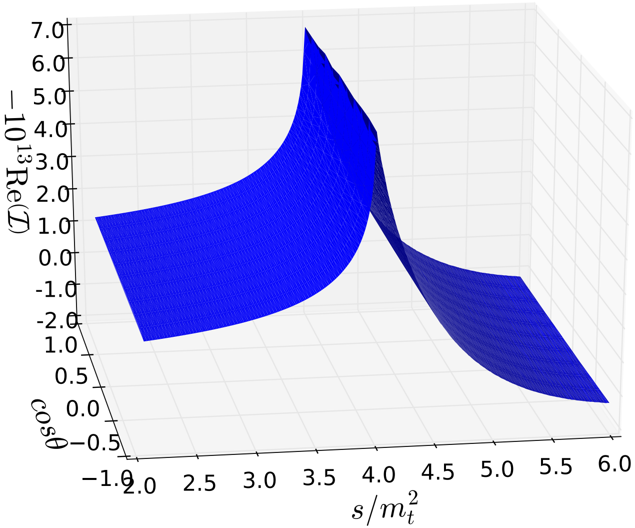
S. Borowka, N. Greiner, G. Heinrich, S.P. Jones, M. Kerner, J. Schlenk, U. Schubert, T. Zirke [1604.06447]



$\mu^{-6-4\epsilon}\mathcal{I} \equiv \text{finite}$ [] $, \ \mu = 1\text{GeV}$

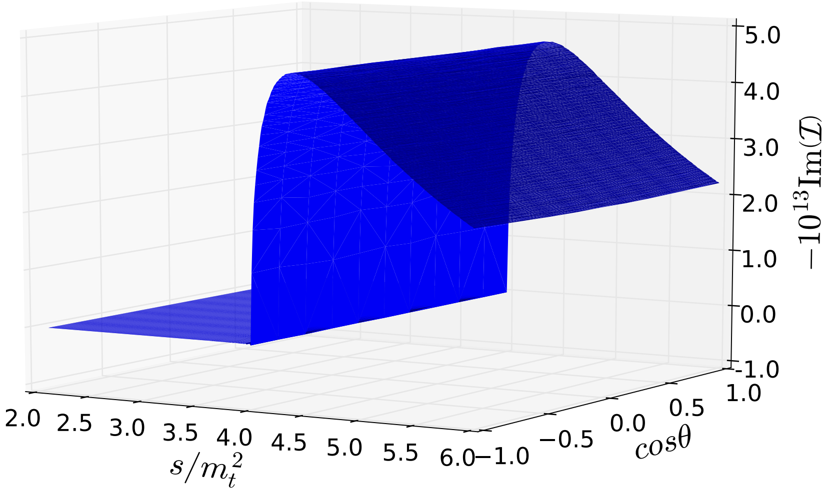
Higgs Boson Pair Production

S. Borowka, N. Greiner, G. Heinrich, S.P. Jones, M. Kerner, J. Schlenk, U. Schubert, T. Zirke [1604.06447]



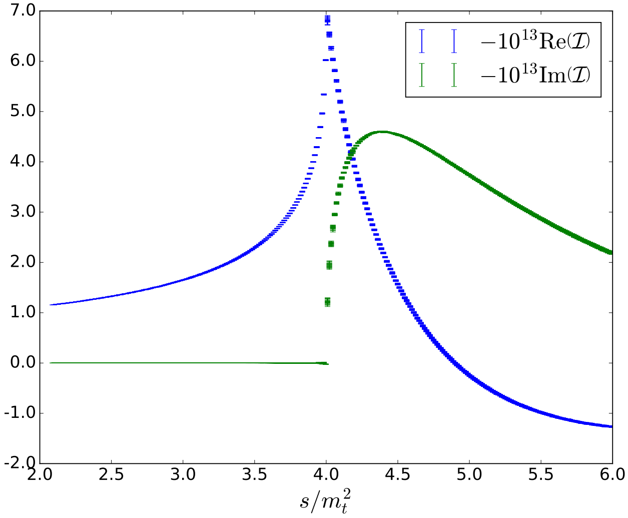
Higgs Boson Pair Production

S. Borowka, N. Greiner, G. Heinrich, S.P. Jones, M. Kerner, J. Schlenk, U. Schubert, T. Zirke [1604.06447]



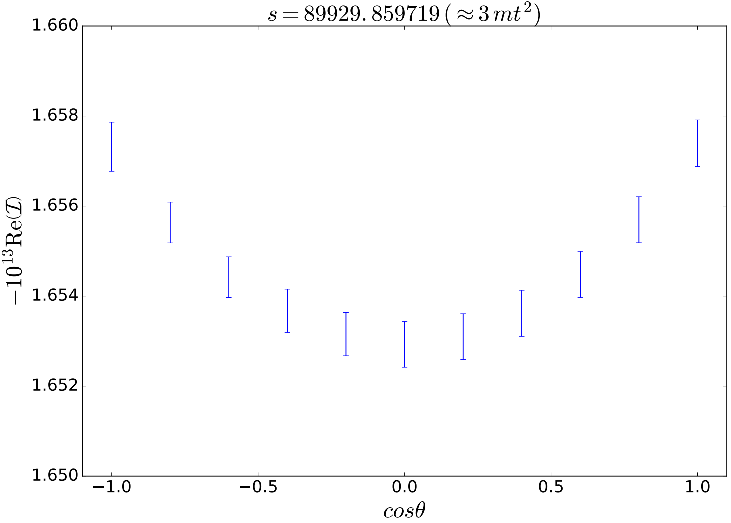
Higgs Boson Pair Production

S. Borowka, N. Greiner, G. Heinrich, S.P. Jones, M. Kerner, J. Schlenk, U. Schubert, T. Zirke [1604.06447]



Higgs Boson Pair Production

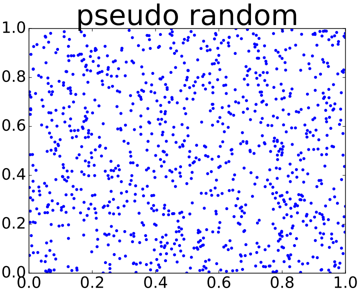
S. Borowka, N. Greiner, G. Heinrich, S.P. Jones, M. Kerner, J. Schlenk, U. Schubert, T. Zirke [1604.06447]



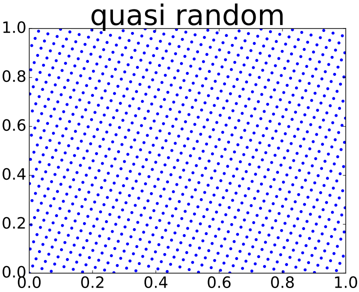
Quasi Monte Carlo (QMC) on GPUs

D. Nuyens et al. (2006), *J. Dick et al.* (2013), *Z. Li et al.* [1508.02512]

will be part of the next major pySecDec release



integral error $\sim \frac{1}{\sqrt{N}}$



integral error between $\sim \frac{1}{N}$ and $\sim \frac{1}{\sqrt{N}}$

with N: number of integrand evaluations

Quasi Monte Carlo (QMC) on GPUs

D. Nuyens et al. (2006), *J. Dick et al.* (2013), *Z. Li et al.* [1508.02512]

will be part of the next major pySecDec release

number of samples	algorithm	CPU	GPU	rel. error
$32 \times 16{,}777{,}213$	QMC	19m	2m	3×10^{-4}
28,754,314	Divonne	14m	-	2×10^{-4}
$32 \times 134{,}217{,}689$	QMC	2h 30m	15m	2×10^{-5}
84,536,102	Divonne	45m	-	4×10^{-5}

- CPU: 4 × Intel Xeon Gold 6140
- GPU: 1 × Tesla V 100
- comparison against Divonne integrator from CUBA library

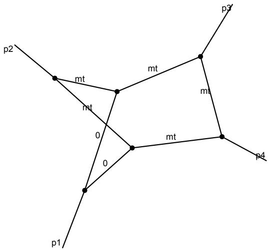
  T. Hahn [hep-ph/0404043]

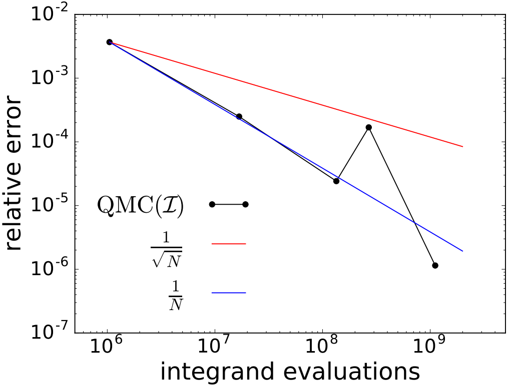
Quasi Monte Carlo (QMC) on GPUs

D. Nuyens et al. (2006), *J. Dick et al.* (2013), *Z. Li et al.* [1508.02512]

will be part of the next major pySecDec release

- scaling can fluctuate





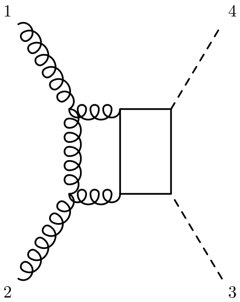
Quasi Monte Carlo (QMC) on GPUs

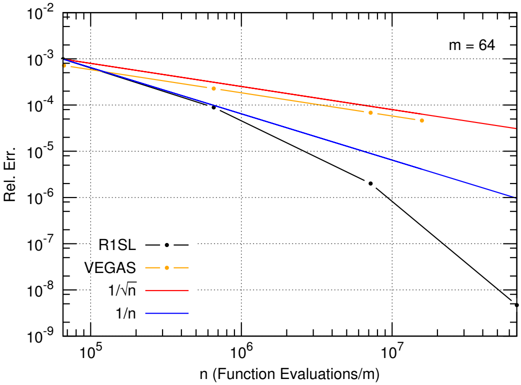
D. Nuyens et al. (2006), *J. Dick et al.* (2013), *Z. Li et al.* [1508.02512]

will be part of the next major pySecDec release

- scaling can fluctuate

plot by Stephen Jones





Summary

introduction to the Sector Decomposition approach
as implemented in pySecDec (http://secdec.hepforge.org/)

- sketch of the method
- application in $gg \to HH$
- reducing the number of integrands using sector symmetries
- improved numerical integration using QMC on GPUs

BACKUP

pySecDec Timings

Table 5
Comparison of timings (algebraic, numerical) using pySecDec, SecDec 3 and Fiesta 4.1.

	pySecDec time (s)	SecDec 3 time (s)	Fiesta 4.1 time (s)
triangle2L	(40.5, 9.6)	(56.9, 28.5)	(211.4, 10.8)
triangle3L	(110.1, 0.5)	(131.6, 1.5)	(48.9, 2.5)
elliptic2L_euclidean	(8.2, 0.2)	(4.2, 0.1)	(4.9, 0.04)
elliptic2L_physical	(21.5, 1.8)	(26.9, 4.5)	(115.3, 4.4)
box2L_invprop	(345.7, 2.8)	(150.4, 6.3)	(21.5, 8.8)

Basic Usage - Analytical Calculation

$$\int_0^1 dx \int_0^1 dy (x+y)^{-2+\epsilon}$$

$$= 2 \int_0^1 dx\, x^{-1+\epsilon} \int_0^1 dt\ (1+t)^{-2+\epsilon}$$

$$= \frac{2}{\epsilon} \left[\int_0^1 dt\ (1+t)^{-2} + \epsilon \int_0^1 dt\ (1+t)^{-2} \log(1+t) + O(\epsilon^2) \right]$$

$$= \frac{2}{\epsilon} \left[\frac{1}{2} + \epsilon\, \frac{1}{2} (1 - \log(2)) + O(\epsilon^2) \right] = \frac{1}{\epsilon} + (1 - \log(2)) + O(\epsilon)$$

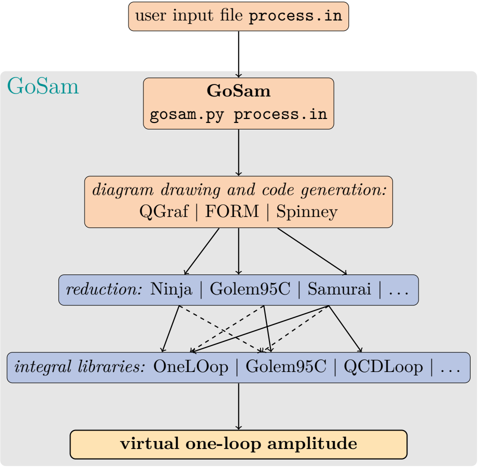
GoSam-1loop

The GoSam collaboration

- Nicolas Greiner
- Gudrun Heinrich
- Stephan Jahn
- Stephen Jones
- Matthias Kerner
- Gionata Luisoni
- Pierpaolo Mastrolia
- Giovanni Ossola
- Tiziano Peraro
- Johannes Schlenk
- Ludovic Scyboz
- Francesco Tramontano

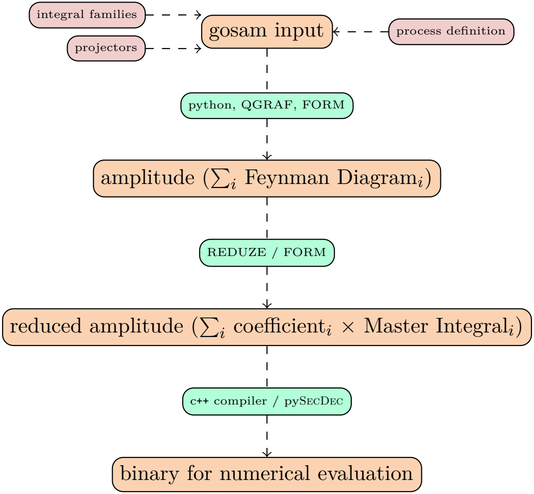
former members

- Gavin Cullen
- Hans van Deurzen
- Edoardo Mirabella
- Joscha Reichel
- Thomas Reiter
- Johann Felix von Soden-Fraunhofen



http://gosam.hepforge.org/

GoSam-Xloop



The GoSam-Xloop collaboration

Nicolas Greiner
Gudrun Heinrich
Stephan Jahn
Stephen Jones
Matthias Kerner
et al.

New Features in pySecDec