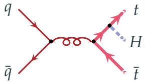
Two-loop virtual contributions to $q\bar{q} \to t\bar{t}H$ production, numerically

Vitaly Magerya (ITP KIT / CERN)
With B. Agarwal, G. Heinrich, S.P. Jones, M. Kerner, S.Y. Klein, J. Lang, A. Olsson

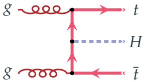
QCD@LHC 2024,
Freiburg, October 8

$t\bar{t}H$ production at the LHC

At the tree level:



quark channel



gluon channel

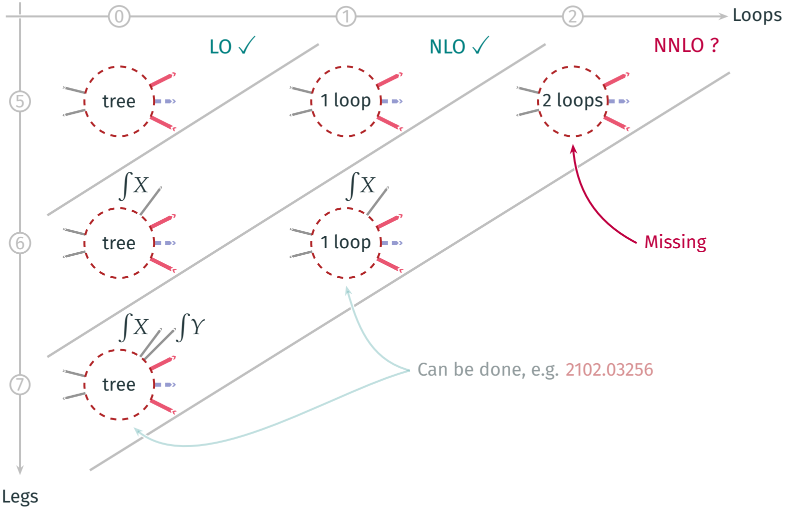
First observation at LHC reported in 2018. [ATLAS ’17, ’17, ’18, ’20, ’23; CMS ’18, ’18, ’20, ’20, ’22]
Measurements based on data from LHC Run 2 (2015–2018):

	$\sigma_{t\bar{t}H}/\sigma_{t\bar{t}H,\mathrm{SM}}$			$\mathscr{L}$	H decay channels
ATLAS ’18	1.32	$^{+0.18}_{-0.18}$(stat)	$^{+0.21}_{-0.19}$(syst)	$79.8\,\mathrm{fb}^{-1}$	$\gamma\gamma, bb, WW, ZZ$
ATLAS ’20	1.43	$^{+0.33}_{-0.31}$(stat)	$^{+0.21}_{-0.15}$(syst)	$139\,\mathrm{fb}^{-1}$	$\gamma\gamma$
CMS ’20	1.38	$^{+0.29}_{-0.27}$(stat)	$^{+0.21}_{-0.11}$(syst)	$137\,\mathrm{fb}^{-1}$	$\gamma\gamma$
CMS ’20	0.92	$^{+0.19}_{-0.19}$(stat)	$^{+0.17}_{-0.13}$(syst)	$137\,\mathrm{fb}^{-1}$	$WW, \tau\tau, ZZ$

HL-LHC will have $\mathscr{L} \sim 3000\,\mathrm{fb}^{-1}$, reducing *statistical* uncertainty by 4-5x.
To reduce *systematic* uncertainty: *NNLO calculation is needed*.

[HL-LHC ’19; Les Houches ’21; Snowmass ’22]

Parts of an NNLO calculation



Big missing part for NNLO: *two-loop virtual amplitudes*.

Theory results for $t\bar{t}H$ production

NLO:

- NLO QCD [Beenakker, Dittmaier, Krämer, Plümper, Spira, Zerwas '01]
  [Reina, Dawson '01]
  [Reina, Dawson, Wackeroth '01]
  [Beenakker, Dittmaier, Krämer, Plümper, Spira, Zerwas '02]
  [Dawson, Orr, Reina, Wackeroth '02]
  [Dawson, Jackson, Orr, Reina, Wackeroth '03]
- NLO QCD, parton shower [Frederix, Frixione, Hirschi, Maltoni, Pittau, Torrielli '11]
  [Garzelli, Kardos, Papadopoulos, Trocsanyi '11]
  [Hartanto, Jager, Reina, Wackeroth '15]
- NLO EW [Frixione, Hirschi, Pagani, Shao, and Zaro '14]
- NLO QCD+EW, NWA [Zhang, Ma, Zhang, Chen, Guo '14]
  [Frixione, Hirschi, Pagani, Shao, and Zaro '15]
- NLO QCD, off-shell [Denner, Feger '15]
  [Stremmer, Worek '21]
  [Denner, Lang, Pellen '20]
  [Bevilacqua, Bi, Hartanto, Kraus, Lupattelli, Worek '22]

Theory results for $t\bar{t}H$ production, II

NLO, contd.:

- NLO+NLL QCD [Kulesza, Motyka, Stebel, Theeuwes '15] [Ju, Yang '19]
- NLO+NNLL QCD [Broggio, Ferroglia, Pecjak, Signer, Yang '15] [Broggio, Ferroglia, Pecjak, Yang '16] [Kulesza, Motyka, Stebel, Theeuwes '17] [Kulesza, Motyka, Schwartländer, Stebel, Theeuwes '20]
- NLO QCD+SMEFT [Maltoni, Vryonidou, Zhang '16]
- NLO QCD+EW, off-shell [Denner, Lang, Pellen, Uccirati '16]
- NLO+NNLL QCD+EW [Broggio, Ferroglia, Frederix, Pagani, Pecjak, Tsinikos '19]
- NLO QCD to $\mathcal{O}(\varepsilon^2)$ [Buccioni, Kreer, Liu, Tancredi '23]
- $t \to H$ fragmentation functions at $\mathcal{O}(y_t^2 \alpha_s)$ [Brancaccio, Czakon, Generet, Krämer '21]

Theory results for $t\bar{t}H$ production, III

NNLO:

- NNLO QCD, flavour off-diagonal, q_T subtraction [Catani, Fabre, Grazzini, Kallweit '21]
- NNLO QCD total cross-section, soft Higgs

  [Catani, Devoto, Grazzini, Kallweit, Mazzitelli, Savoini '22]
- Two-loop QCD virtual amplitude, IR poles [Chen, Ma, Wang, Yang, Ye '22]
- Leading N_c two-loop QCD master integrals, n_l-part

  [Cordero, Figueiredo, Kraus, Page, Reina '23]
- Two-loop QCD virtual amplitude, high-energy boosted limit

  [Wang, Xia, Yang, Ye '24]
- *Two-loop QCD virtual amplitude, $q\bar{q}$ channel, n_l- and n_h-parts*

  [Agarwal, Heinrich, Jones, Kerner, Klein, Lang, V.M., Olsson '24; this talk!]

The amplitude

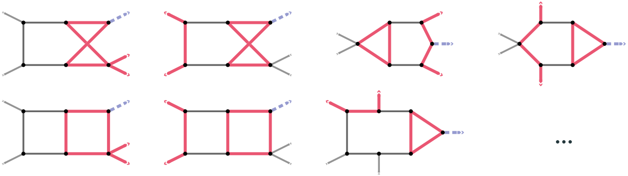
Model: QCD with a scalar H, n_l light (massless) quarks, n_h heavy (top) quarks.
Amplitude of $q\bar{q} \to t\bar{t}H$ projected onto Born, and decomposed in α_s as

$$\langle \text{AMP} \,|\, \text{AMP}_{\text{tree}} \rangle = \mathscr{A} + \left(\frac{\alpha_s}{2\pi}\right) \mathscr{B} + \left(\frac{\alpha_s}{2\pi}\right)^2 \mathscr{C} \,.$$

As a proof-of-concept: only parts proportional to *n_l or n_h* in $\mathscr{C}$ for now.

Why is the calculation complicated?

1. IBP reduction of the amplitude to master integrals is too complicated to be computed symbolically (at the moment).
   * 5 legs and 2 masses $(m_t, m_H) \Rightarrow$ 7 scales (6 scaleless variables).
2. Massive two-loop integrals contributing to $\mathscr{C}$ are not known analytically.



Calculation method

1. Generate all Feynman diagrams for $q\bar{q} \to t\bar{t}H$ at two loops. [QGRAF]
   - $\Rightarrow$ 249 non-zero diagrams (of 702 for the full $q\bar{q}$ channel).
2. Insert Feynman rules, apply the projector $|\,\mathrm{AMP}_{\mathrm{tree}}\rangle$. [ALIBRARY]
3. Sum over the spinor and color tensors. [FORM; COLOR.H]
   - $\Rightarrow$ ~20000 scalar integrals (of ~90000);
   - $\Rightarrow$ 9 structures: $\{n_h|n_l\}\,C_A C_F N_c$, $\{n_h|n_l\}\,C_F^2 N_c$, $\{n_h|n_l\}\,d_{33}$, $\{n_h|n_l\}^2\,C_F N_c$;
     - * 6 structures not included: $C_A^2 C_F N_c$, $C_A C_F^2 N_c$, $C_F^3 N_c$, $C_A d_{33}$, $C_F d_{33}$, d_{44}.
4. Resolve integal symmetries, construct integral families. [FEYNSON; ALIBRARY]
   - $\Rightarrow$ 44 families, 28 up to external leg permutation (of 89 and 39).
5. Figure out master integral count in each sector. [KIRA]
   - $\Rightarrow$ 831 master integrals in total (of 3005 for the full $q\bar{q}$ channel);
   - $\Rightarrow$ up to 8 integrals per sector (up to 13 for the full $q\bar{q}$ channel).

...

Calculation method, II

6. Choose a *good master integral basis*, allowing raised denominator powers and dimensional shifts.
7. Generate IBP relations, dimensional recurrence relations. [Kira; Alibrary]
8. *Precompute* ("trace") the *IBP solution* for each family with Rational Tracer. [Ratracer]
9. *Precompile* the pySecDec *integration library* for the amplitude pieces. [pySecDec]
   - Each color structure as a separate weighted sum of the master integrals.
10. *For each point* in the phase space:
    - 10.1 *Solve IBP relations* using the precomputed trace (with Ratracer).
      - Each Mandelstam variable set to a rational number.
    - 10.2 *Evaluate the amplitudes* as weighted sums of masters (with pySecDec).
      - The weights are taken from the IBP solution.
    - 10.3 Apply renormalization and pole subtraction. [Ferroglia, Neubert, Pecjak, Yang '09; Bärnreuther, Czakon, Fiedler '13]
    - 10.4 Save the result.

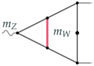
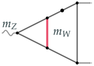
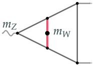
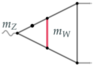
Choosing the master integrals

A good basis of master integrals optimizes the IBP solution time and the pySecDec evaluation time. Our choice:

- ⋆ is quasi-finite, [von Manteuffel, Schabinger ’14]
- ⋆ is d-factorizing, [Smirnov, Smirnov ’20; Usovitsch ’20]
- ⋆ results in IBP coefficients with small denominators,
- ⋆ avoids ε poles in the coefficients of top-level sectors,
- ⋆ avoids ε poles in the differential equation matrix,
- ⋆ is fast to evaluate with pySecDec.

$\Longrightarrow$ Need to consider denominator powers raised up to 6, and dimensional shifts to $d = 6 - 2\varepsilon$ and $d = 8 - 2\varepsilon$.

To illustrate, pySecDec integration time to 10^{-3} precision:[1]

	$\varepsilon^{-2} \ldots \varepsilon^{0}$	>2h		$\varepsilon^{-2} \ldots \varepsilon^{0}$	20m
	$\varepsilon^{-2} \ldots \varepsilon^{0}$	1m		$\varepsilon^{-3} \ldots \varepsilon^{0}$	27s

[1] pySecDec 1.5.3, NVidia A100 GPU.

IBP relations with RATRACER

Solving IBP with Rational Tracer

Basic finite field method: [von Manteuffel, Schabinger '14; Peraro '16]

1) solve IBP equations many times using modular arithmetic with variables set to integers modulo a 63-bit prime;
   - same sequence of operations, many times, with different numbers;
2) reconstruct the coefficients as rational functions from the many samples.

Observation: modular arithmetic is so fast, that typical *solvers waste 90% of the time* managing their data structures (not performing the arithmetic).

Improvement: cut the waste, and abandon the data structures:

- solve the system once using modular artihmetics, and *record every arithmetic operation* into a file (a *"trace"*);
- instead of re-solving the system from scratch, just *replay the trace*.

Implementation: *Rational Tracer* (RATRACER). [V.M. '22]

- github.com/magv/ratracer
- Around 10x faster black-box evaluation than KIRA.

[github.com/magv/ibp-benchmark]

Solving IBP with Rational Tracer, II

Additional trick with traces:

- A trace is a stand-in for a rational expression, and can be *expanded in ε*, producing a new trace that
  - outputs the ε expansion of the IBP coefficients directly,
  - drops ε from the list of considered variables.
  - $\Longrightarrow$ 3x-4x performance gain for this calculation.

For our case of 2-loop $q\bar{q} \to t\bar{t}H$ (n_l- and n_h-parts):

- Reduction is done for each phase-space point separately.
  - $\Longrightarrow$ Mandelstam variables are set to rational numbers.
- Coefficients are expanded into a series in ε.
  - $\Longrightarrow$ No need to reconstruct in ε.
- $\Longrightarrow$ RATRACER outputs *rational numbers* (no need for function reconstruction).
  - Traces of size 0.4–90MB per family, 500MB in total (compressed).
- $\Longrightarrow$ IBP reduction in under *2 CPU minutes per phase-space point*.
  - Down from $\sim$1 hour on 16 cores with KIRA 2.3+FIREFLY!
  - Fast enough that we don't need symbolic IBP solution.

Feynman integrals with pySecDec

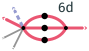
Amplitude evaluation with pySecDec

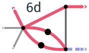
pySecDec: library for numerically evaluating Feynman integrals via *sector decomposition* and *(Quasi-) Monte Carlo integration*. [Heinrich et al '23, '21, '18, '17]

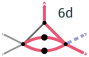
- github.com/gudrunhe/secdec
- Takes a specification for *weighted sum of integrals* (i.e. amplitudes), decomposes integrals into sectors, produces an integration library.
  - Integration via Randomized Quasi-Monte Carlo on *rank-1 lattice rules* constructed via *median QMC lattice* construction (new in v1.6), applied to *Korobov-transformed integrands*. [Heinrich et al '23; Goda, L'Ecuyer '22]
  - We use one sum per color structure.
  - Integrals sampled adaptively to reach the requested precision of the sums.
  - The 831 masters decompose into $\sim$18000 sectors ($\sim$28000 integrals).
- Integration time to get 0.3% precision for this calculation on a GPU:
  - from *5 minutes in the bulk* of the phase-space,
  - to ∞ near boundaries (e.g. high-energy region) due to growing cancellations and spiky integrals (capped at 1 day).

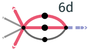
Dealing with large cancellations

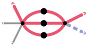
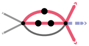
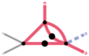
Large cancellations in parts of the high-energy region, e.g.:

$\mathscr{C} = 10^{29}$  $+\,10^{29}$ 

$+\,10^{24}$ 

$+\,10^{24}$ 

$+\,10^{24}$ 

$+\,10^{19}$  $+\,10^{19}$  $+\,10^{18}$ 

$+ \cdots \approx 10^{-3}$

- ⋆ Knowing the integrals at full double precision (16 digits) is not enough!
- ⋆ The cancelling integrals converge well with QMC.
  - ⋆ The precision is limited by the use of double floats more than convergence.

⟹ Make pySecDec use *double-double* (32 digits) for integrals that need it:

[Bailey, Li, Hida ’03; Shewchuk ’97]

- ⋆ *20+ digits of precision* for 4-propagator integrals reachable;
- ⋆ custom implementation for CPUs and GPUs;
- ⋆ around 20x performance hit compared to doubles.

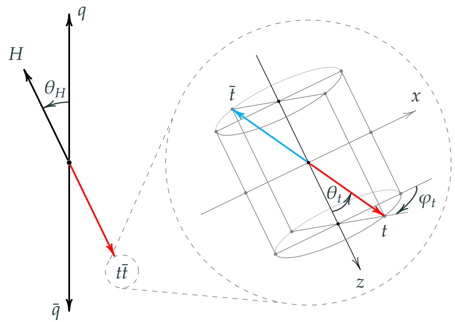
Results for $q\bar{q}\to t\bar{t}H$

Phase-space parameters

We parameterize the $q\bar{q} \to t\bar{t}H$ phase space as chained decay, and instead of

$$s = \left(p_q + p_{\bar{q}}\right)^2 \in \left[\,(2m_t + m_H)^2\,;\,\infty\right],$$
$$s_{t\bar{t}} = \left(p_t + p_{\bar{t}}\right)^2 \in \left[\,(2m_t)^2\,;\,\left(\sqrt{s} - m_H\right)^2 - (2m_t)^2\,\right],$$

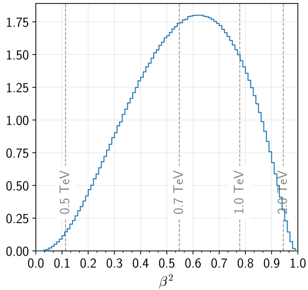
introduce:

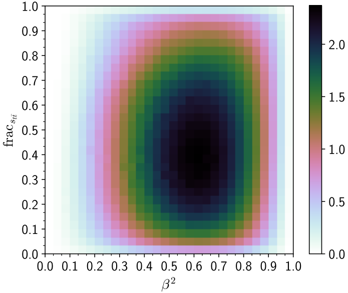


$$\beta^2 \equiv 1 - \frac{s_{min}}{s} \in [0;1],$$
$$\text{frac}_{s_{t\bar{t}}} \equiv \frac{s_{t\bar{t}} - s_{t\bar{t},min}}{s_{t\bar{t},max} - s_{t\bar{t},min}} \in [0;1],$$
$$\theta_H \in [0;\pi],$$
$$\theta_t \in [0;\pi],$$
$$\varphi_t \in [0;2\pi].$$

Which parts of the phase-space are relevant?

Event density at the LHC according to the tree-level amplitude:





To cover 90% of events: $\beta^2 \in [0.24, 0.88]$, that is $\sqrt{s} \in [540 \text{ GeV}, 1.4 \text{ TeV}]$.

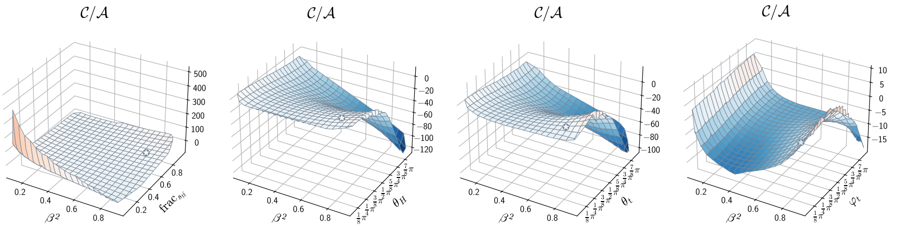
* * *

Example results as two-dimensional slices around the center point of:

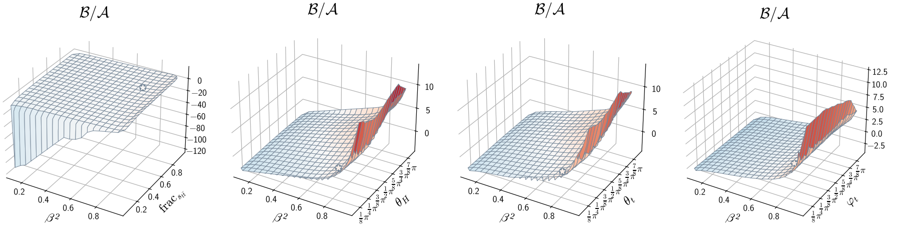
$$\beta^2 = 0.8, \qquad \text{frac}_{s_{t\bar{t}}} = 0.7,$$
$$\cos\theta_H = 0.8, \qquad \cos\theta_t = 0.9, \qquad \cos\varphi_t = 0.7,$$
$$m_H^2 = 12/23\, m_t^2, \qquad \mu = s/2.$$

Resulting slices in β^2 and $\mathrm{frac}_{s_{t\bar{t}}}$, θ_H, θ_t, φ_t

N_f part of the two-loop amplitude (*our result*):

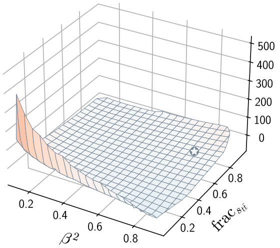


One-loop amplitude (*already known*):

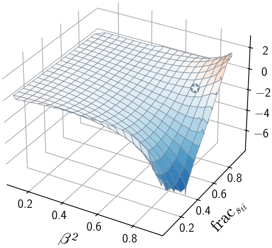


Resulting slices in β^2 and $\mathrm{frac}_{s_{t\bar{t}}}$

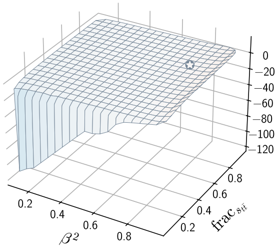
$\mathcal{C}/\mathcal{A}$



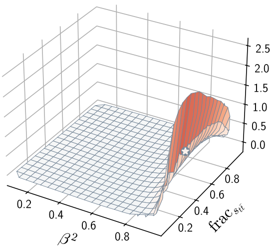
$\mathcal{C} \times$ (phase-space density) $\times 10^3$



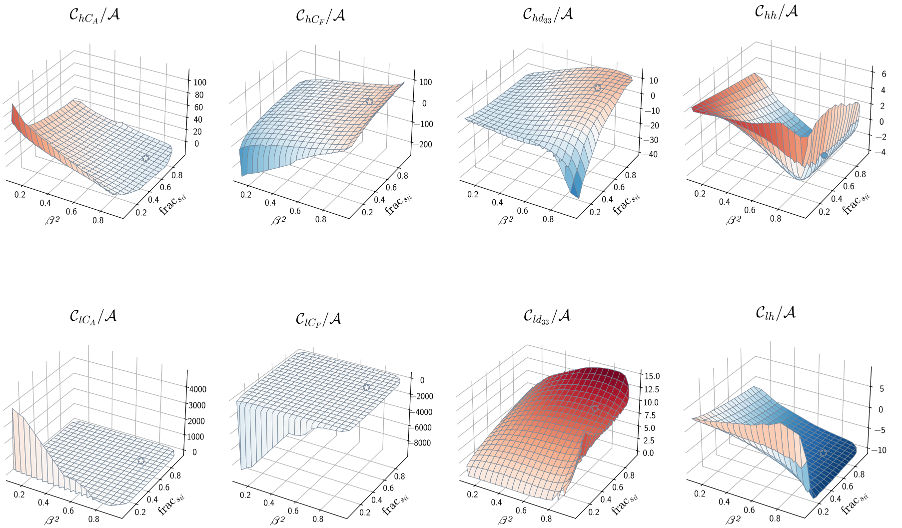
$\mathcal{B}/\mathcal{A}$



$\mathcal{B} \times$ (phase-space density) $\times 10^3$



Resulting slices in β^2 and $\mathrm{frac}_{s_{t\bar{t}}}$ by color factor

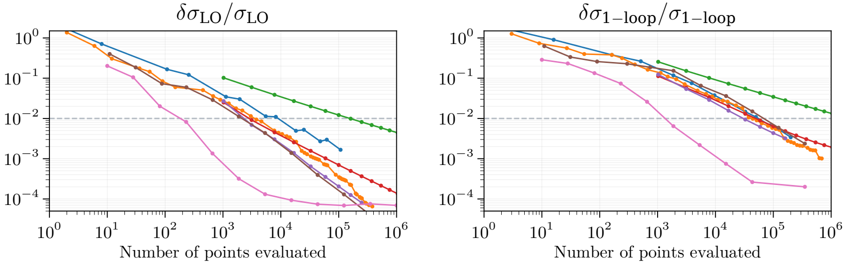


How to use the results?

Goal: precompute points on a 5-dimensional grid, *interpolate in between*.

- How few points do we need to evaluate for 1% approximation error?
  - How to define "approximation error" in the first place?
- Which interpolation method fits best?
  - Splines, polynomials, rationals, sparse grids, radial basis functions, low-rank decompositions, neural networks?
- At which points to sample?
  - Random unweighted samples, RAMBO samples, regular grids, sparse grids, lattices, Padua points, Fekete points, locally adaptive points?

Total cross section approximation error with various methods (PRELIMINARY!):



Summary & Outlook

Done:

- N_f-part of the two-loop virtual amplitude for $q\bar{q} \to t\bar{t}H$.
- IBP performance improvements with RATRACER.
- Peformace and precision improvements in pySECDEC.

In progress:

- The rest of the two-loop virtual amplitude for $q\bar{q} \to t\bar{t}H$.
- Interpolation for the results.

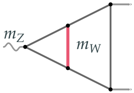
Future plans:

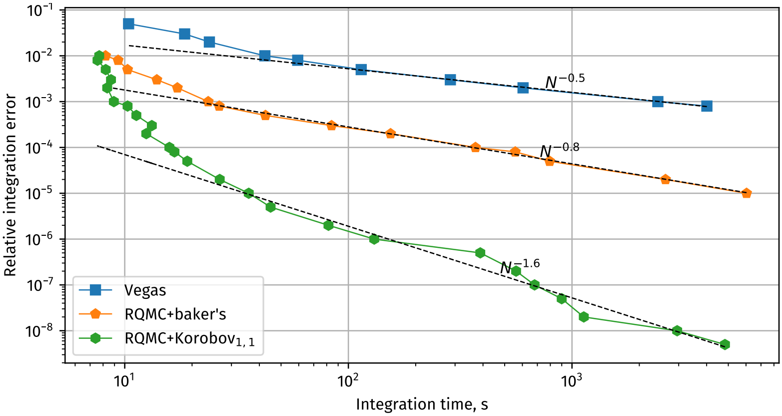
- Two-loop virtual amplitude for $gg \to t\bar{t}H$.
- Combination with real radiation.
- Phenomenological applications.

Backup slides

Monte Carlo vs RQMC

Integration time scaling for Monte Carlo (VEGAS) vs Randomized Quasi Monte Carlo (QMC).[2]





[2] pySECDEC v1.5.3 on NVidia A100 GPU.