Virtual Corrections to ZH Production via Gluon Fusion

Stephen Jones
IPPP, Durham / Royal Society URF

In collaboration with:
Chen, Heinrich, Kerner, Klappert, Schlenk
+ Jahn, Langer, Magerya, Põldaru, Villa

JHEP 03 (2021) 125 [2011.12325]
(pySecDec Release) [2107.XXXX]

Outline

Motivation & Background

Higgs precision, why corrections to $gg \to ZH$ are interesting

Setup of Calculation

Tensor decomposition/ handling γ_5 / integral reduction

Numerical calculation of multi-loop Feynman integrals

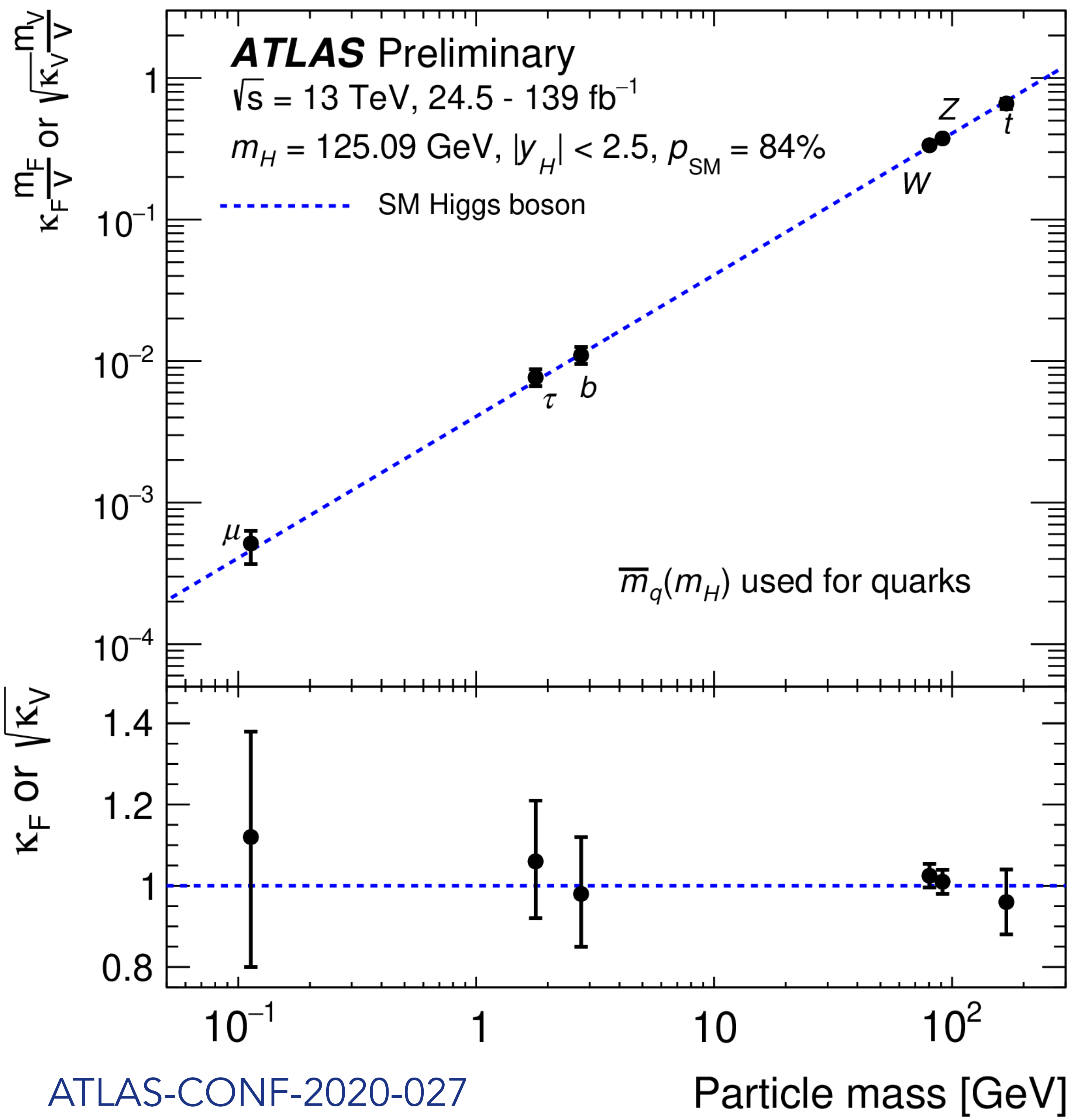
Sector decomposition

Evaluating amplitudes with pySecDec (new release coming soon)

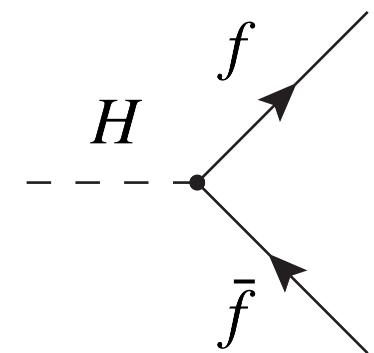
Results & Comparisons

Higgs Couplings

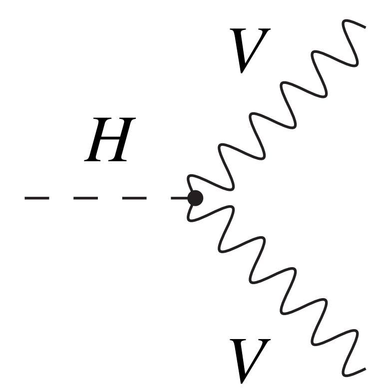
Incredible progress has been made in establishing properties of the Higgs Boson



ATLAS-CONF-2020-027



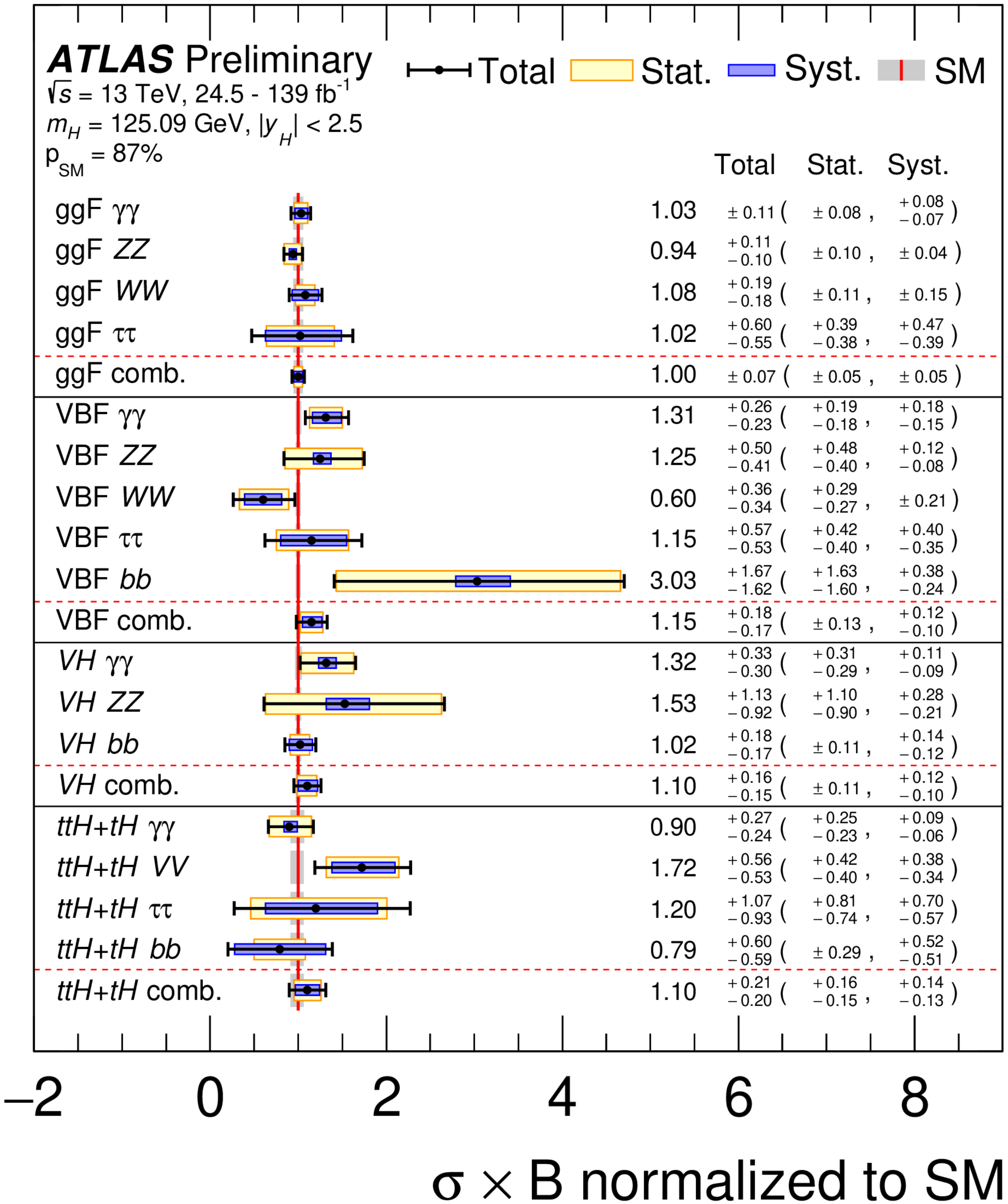
$$= -i\frac{m_f}{v}$$

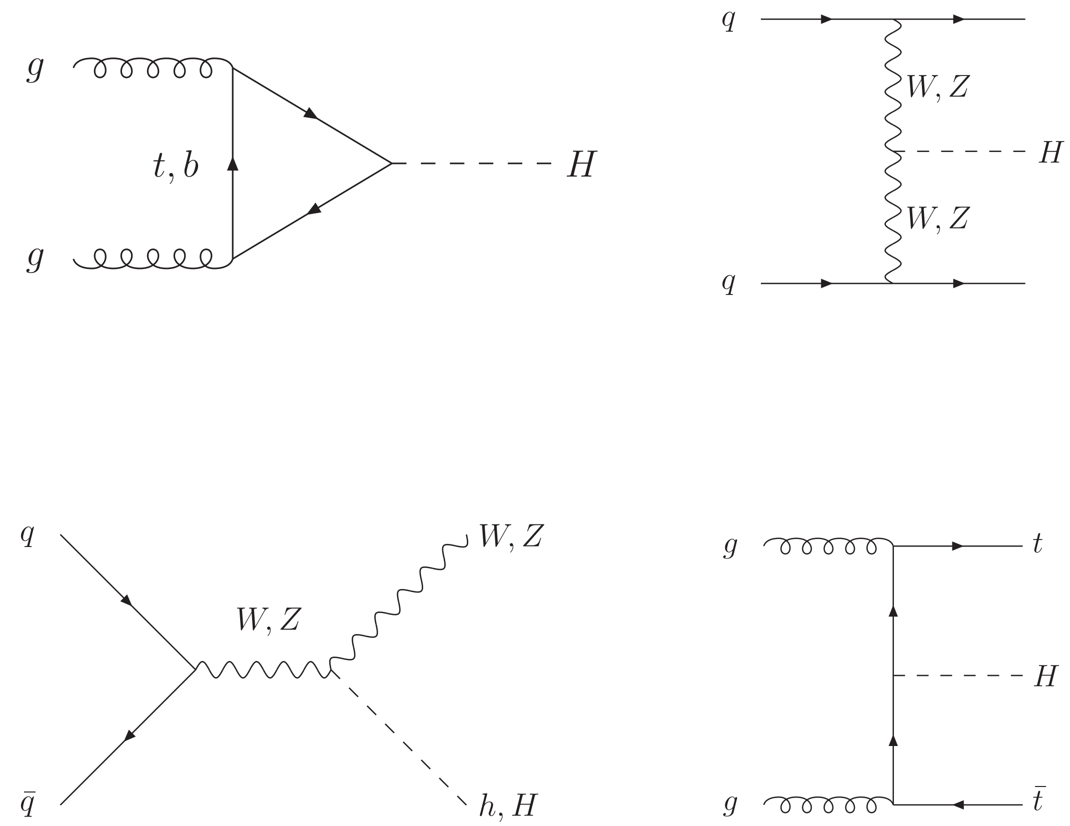


$$= 2i\frac{M_V^2}{v}g_{\mu\nu}$$

Higgs Production & Decay

ATLAS-CONF-2020-027





Crucial to have accurate theory predictions for comparison to data

Higher Order Computations

QCD Factorisation Formula

$$d\sigma = \int dx_a dx_b f(x_a) f(x_b) d\hat{\sigma}_{ab}(x_a, x_b) F_J + \mathcal{O}\left((\Lambda/Q)^m\right)$$

PDFs/ Input parameters

Hard Scattering Matrix Element

Non-perturbative effects ~ few %

With $\alpha_s \sim 0.1$

Typically: NLO ~ 10% correction, NNLO ~ 1% correction

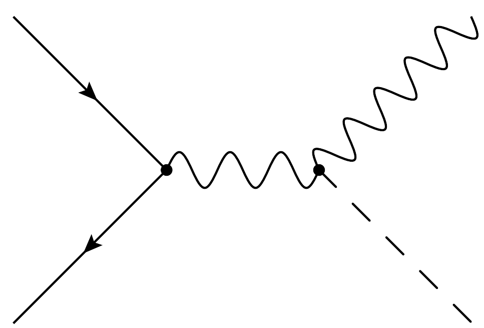
However, there are important exceptions:

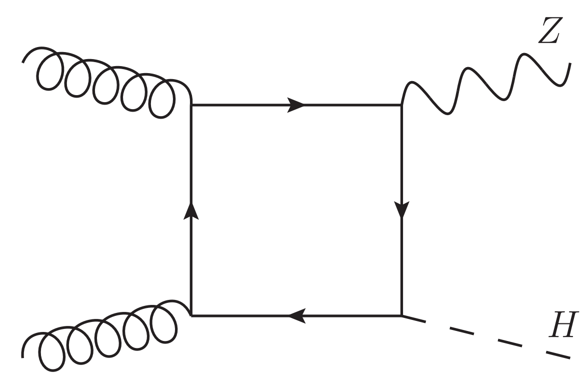
- Higgs Boson production (NLO ~100%, NNLO ~10%, N3LO ~ 2%)
- New partonic channels can open (e.g. $gg \to ZH$)

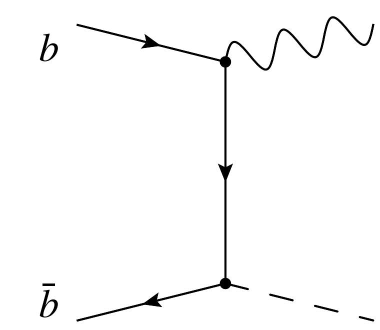
and

- Distributions can be modified substantially (even if σ_{tot} is stable)

Overview of $pp \to ZH$







Drell-Yan piece (NNLO)
Brein, Djouadi, Harlander 03;
Ferrera, Grazzini, Tramontano 14;
See also: Kumara, Mandal, Ravindran 14

+ $q\bar{q}$ piece with closed top loops (1-3%)

Gluon-fusion piece

$b\bar{b}$ piece (NNLO)
Ahmed, Ajjath, Chen, Dhani, Mukherjee, Ravindran 19

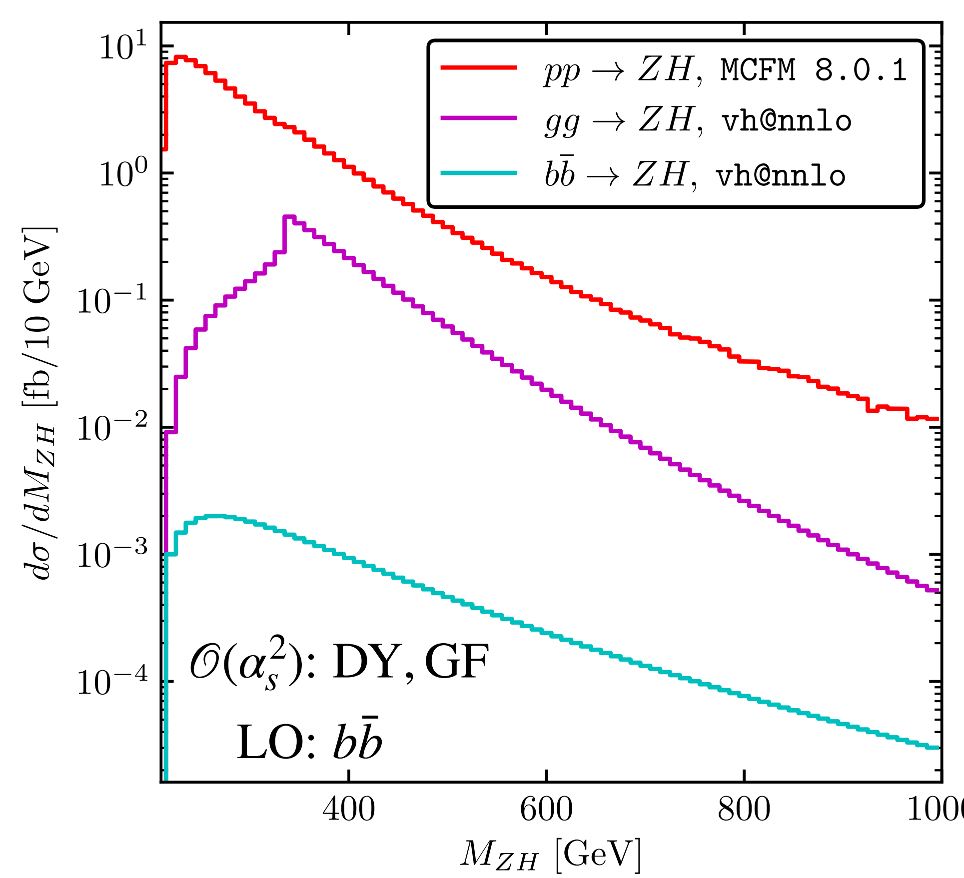
Available in various codes:

HAWK (NLO QCD + NLO EW)
Denner, Dittmaier, Kallweit, Mück 14

vh@nnlo (NNLO QCD + NLO EW)
Harlander, Klappert, Liebler, Simon 18;
Brein, Harlander, Zirke 12

MCFM (NNLO QCD)
Campbell, Ellis, Williams 16

GENEVA (NNLL′+NNLO with PS)
Alioli, Broggio, Kallweit, Lim, Rottoli 19



Harlander, Klappert, Liebler, Simon 18

Why Calculate $gg \to ZH$?

The practical reason …

~10% of total cross section

~100% scale uncertainty (and underestimated?)

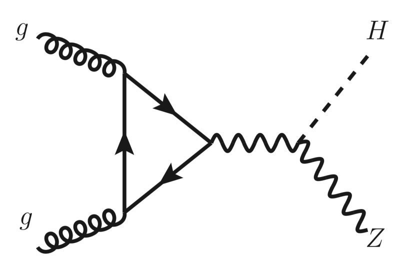
	Signal
Cross-section (scale)	0.7% (qq), $\boxed{25\% \ (gg)}$
$H \to b\bar{b}$ branching fraction	1.7%
Scale variations in STXS bins	3.0%–3.9% ($qq \to WH$), 6.7%–12% ($qq \to ZH$), $\boxed{37\%\text{–}100\% \ (gg \to ZH)}$
PS/UE variations in STXS bins	1%–5% for $qq \to VH$, 5%–20% for $gg \to ZH$
PDF+α_S variations in STXS bins	1.8%–2.2% ($qq \to WH$), 1.4%–1.7% ($qq \to ZH$), 2.9%–3.3% ($gg \to ZH$)
m_{bb} from scale variations	M+S ($qq \to VH$, $gg \to ZH$)
m_{bb} from PS/UE variations	M+S
m_{bb} from PDF+α_S variations	M+S
p_T^V from NLO EW correction	M+S

ATLAS 2007.02873

ATLAS and CMS use ggZH@LO(QCD) from POWHEG

[inclusive calculation: NLO + NLL (QCD)]

$\longrightarrow$ *Large scale uncertainties*

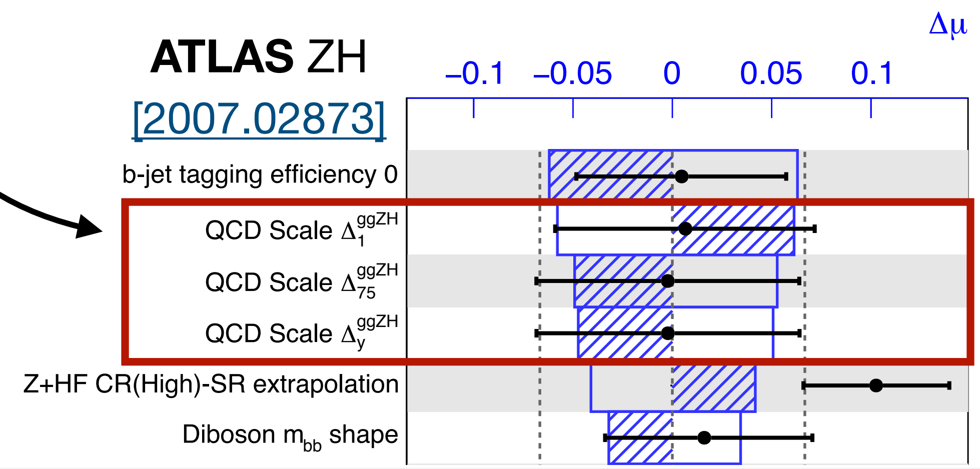


ggZH accounts only for ~10% of inclusive ZH cross section

Uncertainties still very important!

No full NLO calculation available for the foreseeable future (?)

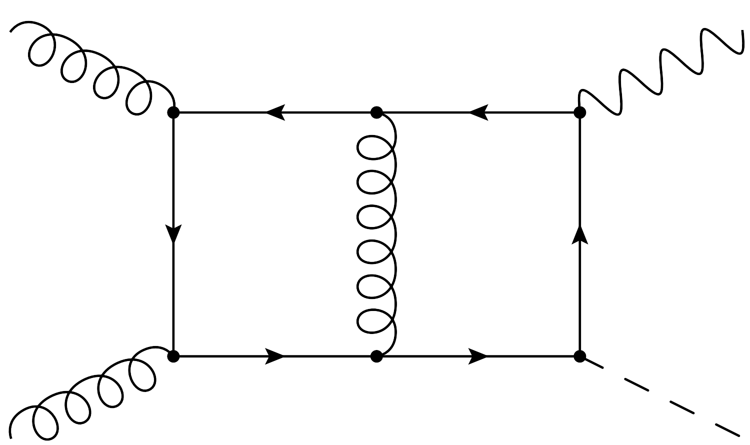
How to improve?

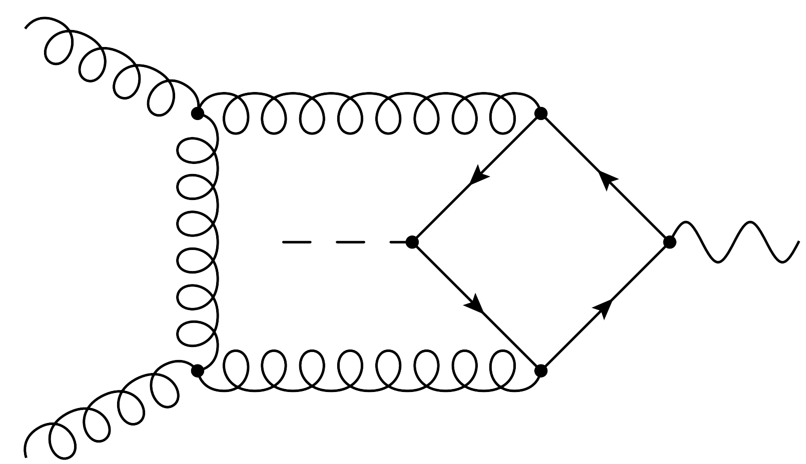


Philipp Windischhofer / ATLAS (LHCXSWG Meeting 9/11/2020)

Why Calculate $gg \to ZH$? (II)

And another reason…





Provides new and interesting challenges:

Amplitude depends on large number of scales $s, t, m_Z^2, m_H^2, m_T^2$

Feynman Integrals appearing are non-trivial (internal masses, elliptic…)

Can test our techniques to breaking point then develop new approaches!

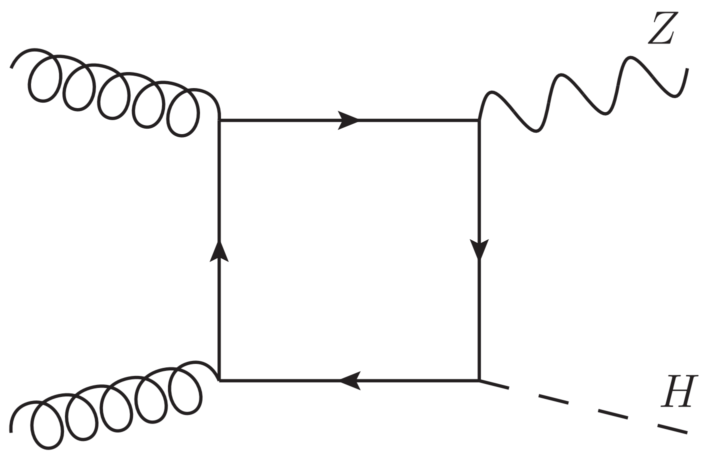
Can we find a basis of integrals with simple coefficients?

How can we obtain a reduction to a finite basis (many dots/numerators)?

Can we improve numerical performance near thresholds? …

ZH in Gluon Fusion

Full leading order (loop induced)
Dicus, Kao 88; Kniehl 90

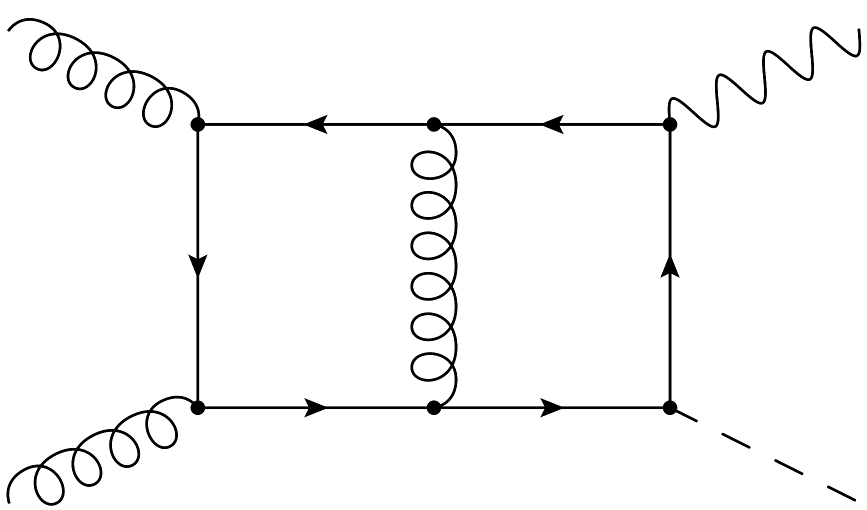


NLO in the limit of $m_t \to \infty$ $(K \approx 2)$
(asymptotic expansion)

Altenkamp, Dittmaier, Harlander, H. Rzehak, Zirke 12

Virtual Corrections:

Expansion around large top quark mass
($1/m_t^8$) + Padé approx
Hasselhuhn, Luthe, Steinhauser 17



Expansion around small top quark mass
($1/m_t^{10}$ & m_t^{32}) + Padé approx
Davies, Mishima, Steinhauser 20

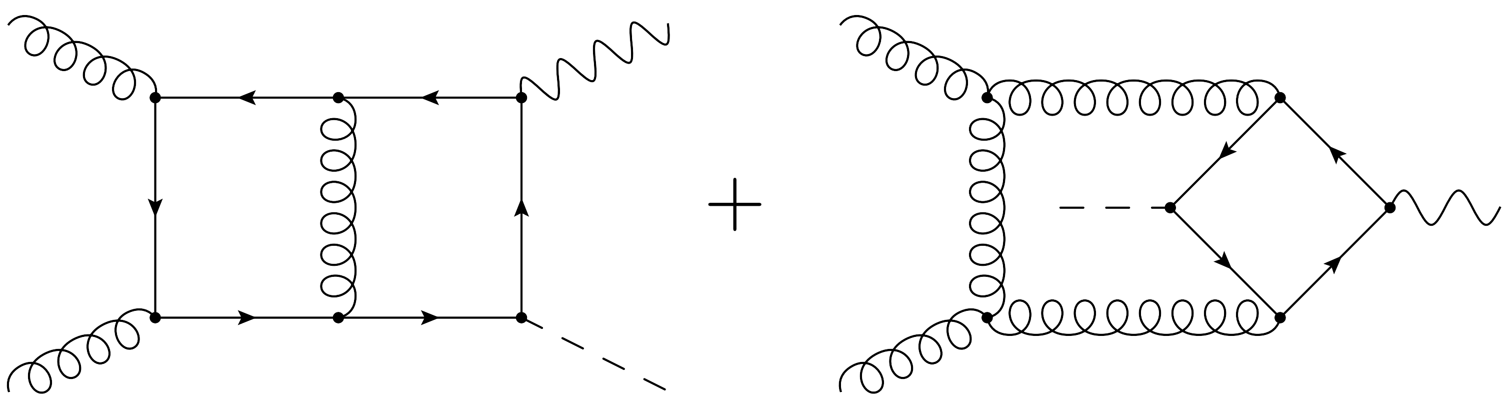
Expansion around small p_T up to p_T^4
Alasfar, Degrassi, Giardino, Gröber, Vitti 21

Full numerical result
Chen, Heinrich, SPJ, Kerner, Klappert, Schlenk 20

Setup & Amplitudes

Amplitudes

Schematically:

$$\mathscr{M}^{\mu\nu\rho} \sim$$  $$+ \dots$$

$$\mathscr{M}^{\mu\nu\rho} = \sum_i A_i T_i^{\mu\nu\rho}, \qquad A_i = \sum_k C_{i,k} \ I_k$$

Rational functions
Large num. terms/ high degree
Handled with specialist symbolic manipulation programs

Feynman integrals
Analytically: Involved special functions (Polylogs, Elliptic…)

In this work, we will compute them numerically

Decomposition: $gg \to ZH$

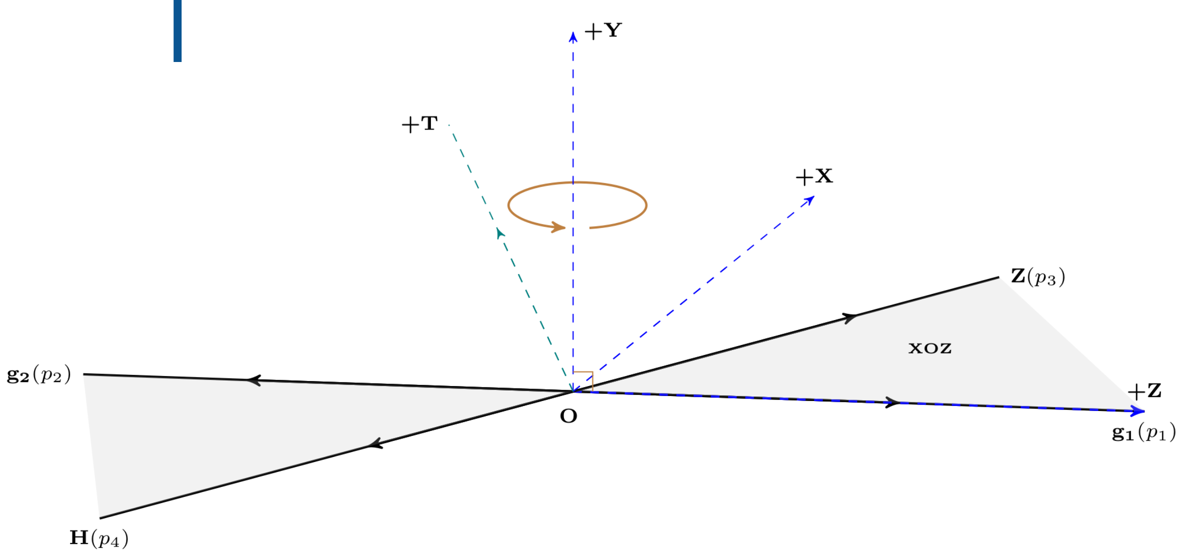
Idea: construct projectors for linearly polarised amplitudes in c.o.m frame, directly compute polarised amplitudes Chen 19

Polarisation vectors can be expressed (up to normalisation factors $\mathcal{N}_i$) in terms of external momenta:

$$
\begin{aligned}
\varepsilon_x^\mu &= \mathcal{N}_x \left(-s_{23} p_1^\mu - s_{13} p_2^\mu + s_{12} p_3^\mu\right) \\
\varepsilon_y^\mu &= \mathcal{N}_y \left(\epsilon^{\mu}_{\mu_1\,\mu_2\,\mu_3}\, p_1^{\mu_1}\, p_2^{\mu_2}\, p_3^{\mu_3}\right) \\
\varepsilon_T^\mu &= \mathcal{N}_T \Big(\left(-s_{23}(s_{13}+s_{23}) + 2m_z^2 s_{12}\right) p_1^\mu + \\
&\qquad \left(s_{13}(s_{13}+s_{23}) - 2m_z^2 s_{12}\right) p_2^\mu + s_{12}(-s_{13}+s_{23})\, p_3^\mu \Big) \\
\varepsilon_l^\mu &= \mathcal{N}_l \left(-2m_z^2 \left(p_1^\mu + p_2^\mu\right) + \left(s_{13}+s_{23}\right) p_3^\mu\right)
\end{aligned}
$$

$$
\begin{aligned}
\varepsilon_x \cdot \{p_1, p_2\} &= 0 \\
\varepsilon_y \cdot \{p_1, p_2\} &= 0 \\
\{\varepsilon_y, \varepsilon_T, \varepsilon_l\} \cdot p_3 &= 0 \\
\varepsilon_i^2 &= -1
\end{aligned}
$$

Projectors are just products of pol. vecs.

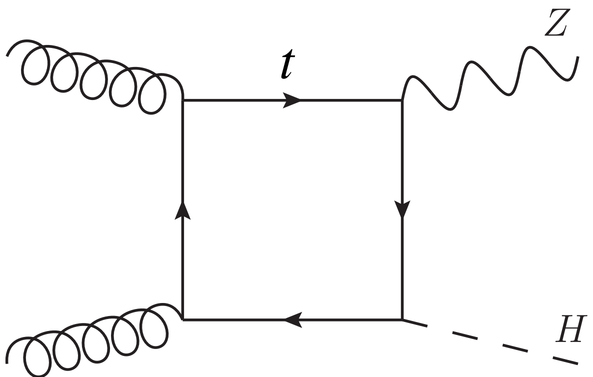


$$
\begin{aligned}
\mathcal{P}_1^{\mu_1\mu_2\mu_3} &= \varepsilon_x^{\mu_1}\,\varepsilon_x^{\mu_2}\,\varepsilon_y^{\mu_3} & \mathcal{P}_2^{\mu_1\mu_2\mu_3} &= \varepsilon_x^{\mu_1}\,\varepsilon_y^{\mu_2}\,\varepsilon_T^{\mu_3}, \\
\mathcal{P}_3^{\mu_1\mu_2\mu_3} &= \varepsilon_x^{\mu_1}\,\varepsilon_y^{\mu_2}\,\varepsilon_l^{\mu_3} & \mathcal{P}_4^{\mu_1\mu_2\mu_3} &= \varepsilon_y^{\mu_1}\,\varepsilon_x^{\mu_2}\,\varepsilon_T^{\mu_3}, \\
\mathcal{P}_5^{\mu_1\mu_2\mu_3} &= \varepsilon_y^{\mu_1}\,\varepsilon_x^{\mu_2}\,\varepsilon_l^{\mu_3} & \mathcal{P}_6^{\mu_1\mu_2\mu_3} &= \varepsilon_y^{\mu_1}\,\varepsilon_y^{\mu_2}\,\varepsilon_y^{\mu_3}.
\end{aligned}
$$

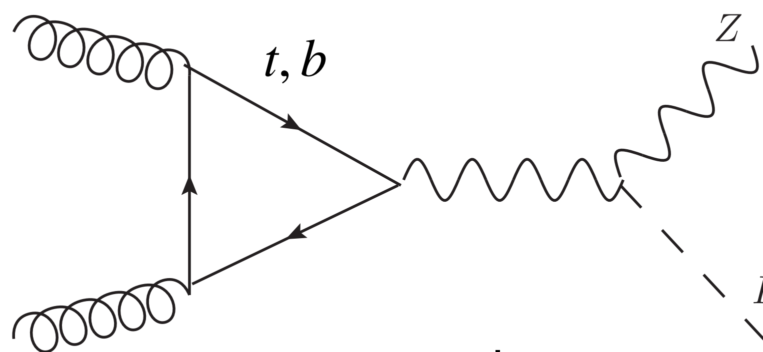
Cross checked with conventional form factor decomposition at LO and at NLO with expansions Davies, Mishima, Steinhauser 20

Diagrams: $gg \to ZH$

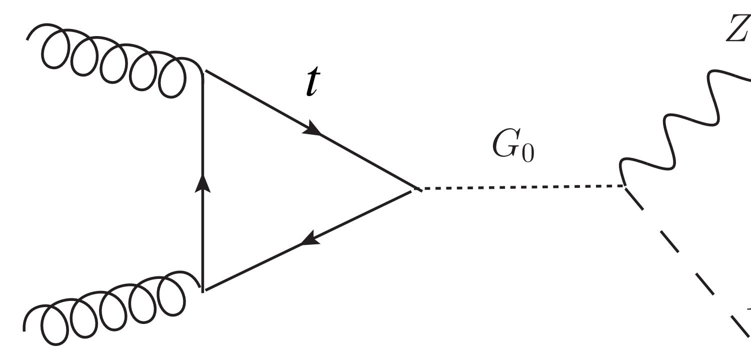
Leading Order (1-loop) Diagrams



Box

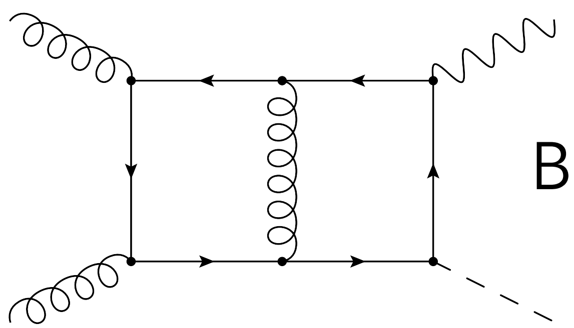


Triangle

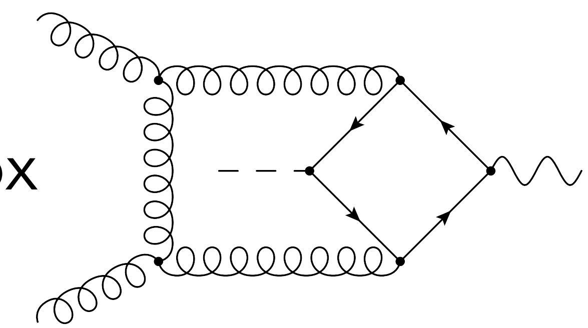


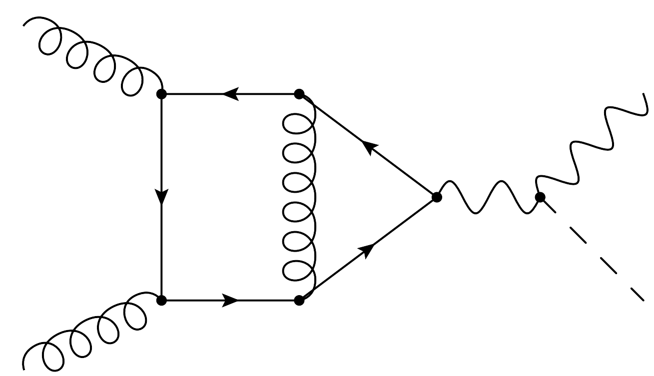
Goldstone

NLO (2-loop) Virtual Diagrams

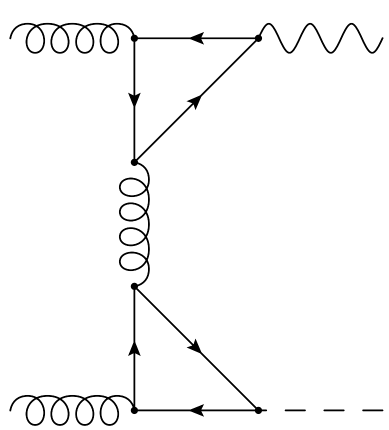


Box





Triangle &
Goldstone



1PR
(start at 2-loop)

We compute in Feynman Gauge

Keep the dependence on all EW couplings symbolic (can be varied)

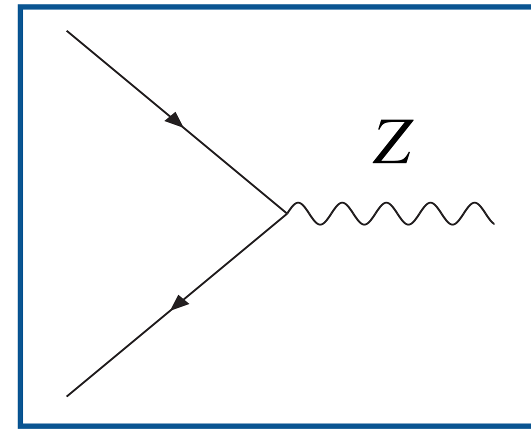
Set $m_b = 0$

Dimensional Regularisation & γ_5

Z-Fermion Vertex

Contains vector $\sim v_t \gamma_\mu$ and axial-vector $\sim a_t \gamma_\mu \gamma_5$:

$$\mathscr{V}_\mu^{Vf\bar{f}} = i \frac{e}{2 \sin \theta_W \, \cos \theta_W} \gamma_\mu \left(v_t + a_t \gamma_5 \right)$$



We use dimensional regularisation ($d = 4 - 2\epsilon$) to regulate divergences appearing in loop integrals, however, one can't retain all properties of γ_5 in $d \neq 4$ dimensions

Larin Scheme (ZH, ZZ)

Sacrifice anti-commuting property of γ_5

$$J_\mu^5 = Z_{5,ns} \, J_{\mu,B}^5 = Z_{5,ns} \left[\frac{i}{3!} \epsilon_{\mu\nu\rho\sigma} \bar{\psi} \gamma^\nu \gamma^\rho \gamma^\sigma \bar{\psi} \right]$$

$$P^5 = Z_{5,p} \, P_B^5 = Z_{5,p} \left[\frac{i}{4!} \epsilon_{\mu\nu\rho\sigma} \bar{\psi} \gamma^\mu \gamma^\nu \gamma^\rho \gamma^\sigma \bar{\psi} \right]$$

Fix Ward identities/ABJ anomaly:

$$Z_{5,ns} = 1 + \alpha_s(-4C_F) + \ldots$$
$$Z_{5,p} = 1 + \alpha_s(-8C_F) + \ldots$$

Larin, Vermaseren 91; Larin 93

Alternative schemes exist e.g:

Kreimer Scheme (ZZ)

Retain $\{\gamma_5, \gamma^\mu\} = 0$, but, sacrifice cyclicity of traces involving γ_5

Define `reading point' and carefully manipulate all traces

Kreimer 90; Korner, Kreimer, Schilcher 92

Dealing with the Integrals

Feynman Integrals

Feynman integrals have many faces, each make different properties manifest…
Switching the representation of our integrals allows us to understand/simplify/complete the calculation

Feynman Parametrisation

$$I \sim \int_0^\infty \prod_{j=1}^{N} \mathrm{d}x_j \;\; x_j^{\nu_1 -1} \dots x_j^{\nu_N -1} \delta(1 - \sum_{i=1}^{N} x_i) \frac{\mathcal{U}^{N_\nu - (L+1)d/2}}{\mathcal{F}^{N_\nu - Ld/2}}$$

$$I = \int \mathrm{d}^d k_1 \dots \mathrm{d}^d k_l \; \frac{1}{\prod_{i=1}^{N} (q_i^2 - m_i^2)_i^{\nu}}$$

…

⋮

…

Lee-Pomeransky Representation

$$I \sim \int_0^\infty \prod_{j=1}^{N} \mathrm{d}x_j \; x_j^{\nu_1 -1} \dots x_j^{\nu_N -1} G^{-D/2}$$

$$G = \mathcal{U} + \mathcal{F}$$

Lee, Pomeransky 13

Baikov Representation

$$I \sim \int dz_1 ... dz_N \; \frac{1}{\prod_{i=1}^{N} \; z_i^{\nu_i}} \; P^{\frac{d-L-E-1}{2}}$$

Baikov 96

Reduction (IBPs)

Integration by parts Identities:

loop/external momentum

$$\int \mathrm{d}^d k_1 \cdots \mathrm{d}^d k_l \, \frac{\partial}{\partial k_i^\mu} \left[v^\mu \mathrm{I}(k_1, \ldots, k_l; p_1, \ldots, p_m) \right] = 0$$

Produce linear relations between integrals Tkachov 81; Chetyrkin 81

Can perform e.g. Gaussian elimination on system of equations
Relate integrals to a smaller set of **Master integrals**

Example:

$$I_{\alpha_1 \alpha_2 \alpha_3} = \int \frac{\mathrm{d}^d p}{i\pi^{d/2}} \frac{1}{D_1^{\alpha_1} D_2^{\alpha_2} D_3^{\alpha_3}}$$

$$k_1^2 = k_2^2 = 0, k_3^2 = s$$

$$D_1 = p^2$$

$$D_2 = (p + k_1)^2$$

$$D_3 = (p + k_1 + k_2)^2$$

IBP Identities:

$$(d-4)I_{111} - I_{102} - I_{201} = 0$$

$$sI_{102} + (d-3)I_{101} = 0$$

$$sI_{201} + (d-3)I_{101} = 0$$

$$\therefore I_{111} = \frac{-2(d-3)}{s(d-4)} I_{101}$$

Finite Integrals

We have freedom regarding which integrals we choose as a basis (**Master Integrals**)

The choice impacts:

1) **Complexity of the coefficients in the amplitude**
2) **Difficulty of computing the integrals**

Always possible to pick a basis of finite integrals using:
Panzer 14; von Manteuffel, Panzer, Schabinger 15

- Dimension Shifts Tarasov 96; Lee 10
- Dots
- Numerator Insertions *(optional, used for $gg \to ZZ$)*

The finite basis greatly improves numerical performance

In fact, the integral basis choice can make the difference between the problem being possible/impossible (see reason 1 or reason 2 above)

Master Integral Basis ($gg \to ZH$)

To select our master integrals, we took the following pragmatic approach:

1) Consider quasi-finite integrals (prefer finite integrals)

$$I = \frac{I_{-2}}{\epsilon^2} + \frac{I_{-1}}{\epsilon} + I_0 + \ldots \quad \to \quad I' = I'_0 + \ldots$$

2) Choose a basis in which the d-dependence of denominators factorises from the kinematic dependence (in practice we achieve this by brute force neglecting subsectors, public tools are available Smirnov, Smirnov 20; Usovitsch 20)

$$\frac{N(s,t,d)}{D(s,t,d)} I + \ldots \to \frac{N'(s,t,d)}{D'_1(d) D'_2(s,t)} I' + \ldots$$

3) Prefer simple denominator factors
4) Prefer computing fewer orders in epsilon for each master (found a basis in which all 7-propagator integrals start contributing only at ϵ^{-1})
5) Prefer simpler numerators (check number of terms/file size)

See also: Matthias Kerner, Loops and Legs Proceedings 2018

Steps 2-5 reduced the size of amplitude by factor of 5

Largest coefficient (double-tadpole) 150 MB $\to$ 5 MB

Baikov Representation (used only for $gg \to ZZ$)

General scalar integral can also be written as Baikov 96

$$I = \mathcal{N} \int dz_1 ... dz_N \, \frac{1}{\prod_{i=1}^N z_i^{\nu_i}} \, P^{\frac{d-L-E-1}{2}}$$

IBPs become

$$0 = \int dz_1 ... dz_N \sum_{i=1}^N \frac{\partial}{\partial z_i} \left(f_i \, \frac{1}{\prod_{i=1}^N z_i^{\nu_i}} \, P^{\frac{d-L-E-1}{2}} \right)$$

$$0 = \int dz_1 ... dz_N \sum_{i=1}^N \left(\frac{\partial f_i}{\partial z_i} + \boxed{\frac{d-L-E-1}{2P} f_i \frac{\partial P}{\partial z_i}} - \boxed{\nu_i \frac{f_i}{z_i}} \right) \frac{1}{\prod_{i=1}^N z_i^{\nu_i}} \, P^{\frac{d-L-E-1}{2}}$$

Dimension shift **Dots (doubled propagators)**

No dimension shifted integrals, few dots in amplitudes

IBPs generate such integrals → (unnecessarily) huge linear system to solve

Impose \`Syzygy' Constraints

$$\boxed{\left(\sum_{i=1}^N f_i \frac{\partial P}{\partial z_i} \right) + f_{N+1} P = 0} \quad \to \text{No dimension shift}$$

$$\boxed{f_i \sim z_i} \quad \to \text{No doubled props.}$$

Larsen, Zhang 15;
Abreu, Febres Cordero, Ita, Page, Zeng 17;
Boehm, Georgoudis, Larsen, Schoenemann, Zhang 17, 18;

Reduction

Need tools to actually solve these systems of equations…

ZZ Computation	**ZH Computation**
FinRed — von Mantueffel (Private) + Syzygy Solver — Agarwal, von Mantueffel	Kira 2 + FireFly — Maierhöfer, Usovitsch, Uwer 18; Klappert, Lange, P. Maierhöfer, Usovitsch 20; Klappert, Lange 20; Klappert, Klein, Lange 20
Master Integrals: 264	**Master Integrals:** 452

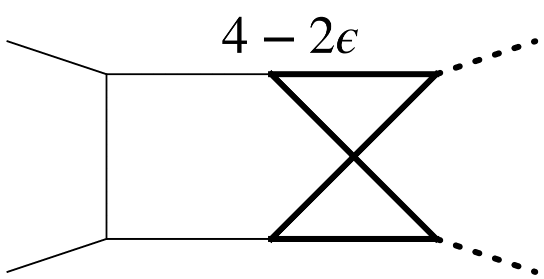
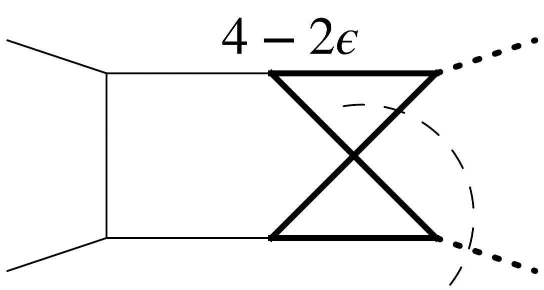
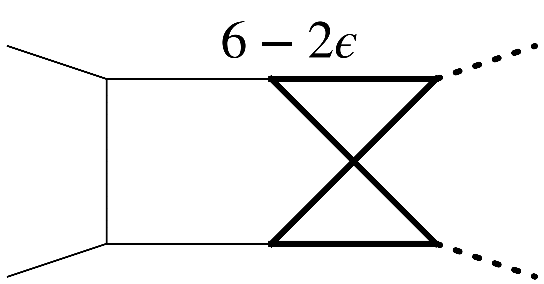
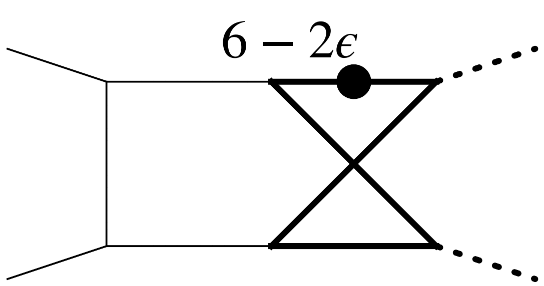
Both toolchains extensively rely on the use of finite fields

See e.g: von Mantueffel, Schabinger 14; Peraro 16

Even with these tools, still too difficult to obtain fully symbolic amplitudes

Fix mass ratios ZZ: $\frac{m_z^2}{m_t^2} = \frac{5}{18}$ and ZH: $\frac{m_z^2}{m_t^2} = \frac{23}{83}, \frac{m_H^2}{m_t^2} = \frac{12}{23}$

Finite Integrals (example from $gg \to ZZ$)

	Finite	ϵ Order	Rel Err	Timing (s)
	No	0	$2 \cdot 10^{-3}$	45
	No	0	$4 \cdot 10^{-2}$	63
	Yes	1	$8 \cdot 10^{-6}$	60
	Yes	1	$8 \cdot 10^{-4}$	55
Linear combination (numerator insertion)	Yes	1	$1 \cdot 10^{-4}$	18

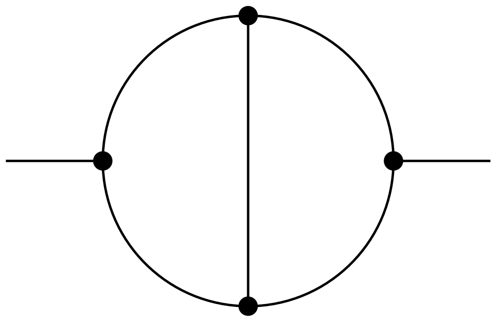
Figures & Timings: Bakul Agarwal

pySecDec + QMC
Nvidia Tesla V100

Computing Integrals Numerically

pySecDec - automated tool for numerically computing Feynman Integrals

Borowka, Heinrich, Jahn, SPJ, Kerner, Langer, Magerya, Põldaru, Schlenk, Villa, Zirke

Input: $I = \int \mathrm{d}^d k_1 \cdots \mathrm{d}^d k_l \frac{1}{(k_1^2 - m^2)((k_1-k_2)^2 - m^2)\cdots}$ **or** $G =$ 

Feynman Parametrise

$$\mathcal{U} = x_1x_3 + x_1x_4 + x_1x_5 + \ldots$$
$$\mathcal{F} = -p^2x_1x_2x_3 - p^2x_1x_2x_4 - \ldots$$

Resolve Singularities (Sector Decomposition)

C++ library ready for numerical integration

Output:

```
$ time python integrate_elliptic2L_euclidean.py
 + (2.470741991407315 60e-01 +/- 4.360755998051713 55e-16) + O(eps)
real        0m2.576s
```

Code publicly available (**GitHub**), widely used and well tested

Amplitude Evaluation

Use Quasi-Monte Carlo (QMC) integration $\mathcal{O}(n^{-1})$ error scaling

Li, Wang, Yan, Zhao 15; Review: Dick, Kuo, Sloan 13;

Implemented in CUDA, evaluated on GPUs

Entire 2-loop amplitude evaluated with a single code

$$A = \sum_i \underbrace{\left(\sum_j C_{i,j} \epsilon^j \right)}_{\text{coeff.}} \underbrace{\left(\sum_k I_{i,k} \epsilon^k \right)}_{\text{integral}} = \epsilon^{-2} \left[C_{1,-2} \cdot I_{1,0} + \ldots \right] + \epsilon^{-1} \left[C_{1,-1} \cdot I_{1,0} + \ldots \right] + \ldots$$

compute once

Dynamically set target precision for each sector, minimising time to obtain amplitude to requested precision:

$$T = \sum_i t_i + \bar{\lambda} \left(\sigma^2 - \sum_i \sigma_i^2 \right), \quad \sigma_i \sim t_i^{-e}$$

$\bar{\lambda}$ – Lagrange multiplier

σ – precision goal

σ_i – error estimate

Amplitude Evaluation (II)

New release (soon): can evaluate entire amplitudes

Let's see how this works in a simple example 1-loop 4-photon amp.:

Step 1: Define Integrals

```
import pySecDec as psd

### Integral definitions ###

I = [

    # one loop bubble (u)
    psd.loop_integral.LoopIntegralFromGraph(
    internal_lines = [[0,[1,2]],[0,[2,1]]],
    external_lines = [['p1',1],['p2',2]],
    replacement_rules = [('p1*p1', 'u'),('p2*p2', 'u'),('p1*p2', 'u')]),

    # one loop bubble (t)
    psd.loop_integral.LoopIntegralFromGraph(
    internal_lines = [[0,[1,2]],[0,[2,1]]],
    external_lines = [['p1',1],['p2',2]],
    replacement_rules = [('p1*p1', 't'),('p2*p2', 't'),('p1*p2', 't')]),

    # one loop box (in 6 dimensions)
    psd.loop_integral.LoopIntegralFromGraph(
    internal_lines = [['0',[1,2]],[0,[2,3]],[0,[3,4]],[0,[4,1]]],
    external_lines = [['p1',1],['p2',2],['p3',3],['p4',4]],
    replacement_rules = [
                        ('p1*p1', 0), ('p2*p2', 0),
                        ('p3*p3', 0), ('p4*p4', 0),
                        ('p3*p2', 'u/2'), ('p1*p2', 't/2'),
                        ('p1*p4', 'u/2'), ('p1*p3', '-u/2-t/2'),
                        ('p2*p4', '-u/2-t/2'), ('p3*p4', 't/2')
                    ],
    dimensionality= '6-2*eps'
    ),

    # one loop box (in 8 dimensions)
    # ...

    )
]
```

Usual pySecDec Syntax

Step 2: Define Coefficients

```
from pySecDec.code_writer.sum_package import Coefficient

### Coefficients definitions ###

coeff = [
    # M++--
    [
        # bubble (u) coefficient
        Coefficient(['-8*(t-u)'],['-u-t'],['eps'],['t','u']),

        # bubble (t) coefficient
        Coefficient(['-8*(u-t)'],['-u-t'],['eps'],['t','u']),

        # box6 coefficient
        Coefficient(['-8*(t**2+u**2)'],['-u-t'],['eps'],['t','u']),

        # box8 coefficient
        Coefficient(['-8*3*(2*eps)'],['1'],['eps'],['t','u'])
    ]
]
```

Numerator **Denominator**

Regulators **Variables**

Supports:
Multiple regulators & variables
Coeffs with poles in regulators
Multiple amplitudes at once

Amplitude Evaluation (III)

Step 3: Generate

```
from pySecDec.loop_integral import LoopPackage
from pySecDec.code_writer import sum_package

# import integrals and coefficients
import integrals
import coefficients

# 4-photon amplitude M++--

if __name__ == '__main__':

    # define input to loop_package
    package_generators = []
    names = ['bubble_u', 'bubble_t', 'box_6', 'box_8']
    for name, integral in zip(names, integrals.I):
        package_generators.append(
            LoopPackage(
                name = name,
                loop_integral = integral,
                requested_orders = [0]
                )
        )

    # generate code sum of (int * coeff)
    sum_package(
        'yyyy1L',
        package_generators,
        regulators = ['eps'],
        requested_orders = [0],
        real_parameters = ['t','u'],
        coefficients = coefficients.coeff,
        complex_parameters = []
    )
```

List of requested integrals

Pass integrals & coefficients

Step 4: Integrate

```
import math
from pySecDec.integral_interface import IntegralLibrary

if __name__ == "__main__":

    # load c++ library
    amp = IntegralLibrary('yyyy1L/yyyy1L_pylink.so')

    # choose Qmc integrator
    amp.use_Qmc()

    # integrate
    _, _, result = amp([-1.3,-0.8]) # t, u

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

Uses all GPUs/CPUs on machine

Usual pySecDec Syntax

Step 5: Generate, Compile, Run, …, Profit!

```
$ python3 generate_yyyy1L.py
$ export CXX=nvcc
$ export SECDEC_WITH_CUDA_FLAGS="-gencode arch=compute_70,code=sm_70"
$ make -C yyyy1L -j 8
$ time python3 integrate_yyyy1L.py
> Numerical Result:
+ ((4.44089209850062616e-16,0.00000000000000000e+00)
+/- (2.30780196667706177e-16,0.00000000000000000e+00))*eps^-1

+ ((-2.84315958344628044e+01,-1.77910044838014154e-10)
+/- (4.25673251664533604e-09,2.41058630701162800e-09))

+ O(eps)

real    0m0.764s
user    0m2.884s
sys     0m0.188s
```

Timings on laptop CPU

Amplitude Evaluation (IV)

A peak behind the curtain (enabled by setting verbose=True)

```
#
# Compute a first estimate of each integral (n: 50789 x 32 samples of integral)
# Note: each order of each sector is computed separately (and can be known to a different precision)
#
computing integrals to satisfy mineval 50000
integral 1/16: bubble_u_sector_1_order_0, time: 0.0 s  res: (0,0) +/- (0,0) -> (1,0) +/- (8.89904e-17,0), n: 0 -> 50789
integral 2/16: bubble_u_sector_1_order_1, time: 0.0 s  res: (0,0) +/- (0,0) -> (2.22314,0) +/- (1.9927e-15,0), n: 0 -> 50789
integral 3/16: bubble_t_sector_1_order_0, time: 0.0 s  res: (0,0) +/- (0,0) -> (1,0) +/- (9.00691e-17,0), n: 0 -> 50789
integral 4/16: bubble_t_sector_1_order_1, time: 0.0 s  res: (0,0) +/- (0,0) -> (1.73764,0) +/- (2.19938e-15,0), n: 0 -> 50789
...

#
# Estimate amplitude(s) using integral results
#
amplitude0 =  + ((0,0) +/- (2.41174e-16,0))*eps^-1 + ((-28.4316,-1.6741e-09) +/- (4.06609e-09,2.39249e-09)) + O(eps)

#
# Examine contribution of each integral to err. of amp. and estimate how many samples we need (n: known -> required)
# Note: not all sectors will be recomputed, different sectors will be computed to different precisions
#
sum: sum_eps^0, term: WINTEGRAL, integral 5: box_6_sector_1_order_0, current integral result: (0.478203,4.5158e-11) +/-
(2.31205e-10,4.72335e-11), contribution to sum error: 2.0946e-09, increase n:  50789 -> 968960972
sum: sum_eps^0, term: WINTEGRAL, integral 15: box_8_sector_5_order_0, current integral result: (0.0173611,-5.39441e-13) +/-
(3.16251e-12,1.29608e-12), contribution to sum error: 1.64054e-10, increase n: 50789 -> 193374172
…

#
# Iterate until we obtain the desired precision
# Note: reason for iteration is printed (long running/run away jobs can be debugged straightforwardly)
#
run further refinements: true
computing integrals to satisfy error goals on sums: epsrel 1e-14, epsabs 1e-14
...
```

Results $gg \to ZH$

Finite Virtual Correction

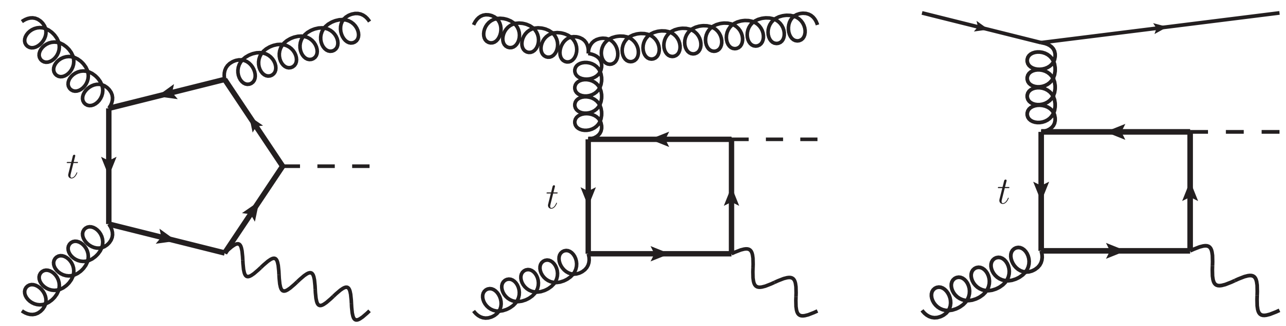
Schematically,

$$\hat{\sigma} = \hat{\sigma}^{\mathrm{LO}} + \hat{\sigma}^{\mathrm{NLO}}$$

$$\hat{\sigma}^{\mathrm{LO}} = \int_n \mathrm{d}\sigma^{\mathrm{B}}$$

$$\hat{\sigma}^{\mathrm{NLO}} = \int_n \mathrm{d}\sigma^{\mathrm{V}} + \int_{n+1} \mathrm{d}\sigma^{\mathrm{R}} + \int_n \mathrm{d}\sigma^{\mathrm{C}}$$

Reals (not included in our work yet)



Virtual part ($\mathrm{d}\sigma^V$) and real part ($\mathrm{d}\sigma^R$) not separately finite for $\epsilon \to 0$

However, we can define a finite virtual contribution as follows:

1) UV renormalize: α_s in $\overline{\mathrm{MS}}$ & top quark mass in OS scheme

2) IR structure well known at NLO, subtract divergences

$$\mathcal{A}_i^{(0),\mathrm{fin}} = \mathcal{A}_i^{(0),\mathrm{UV}},$$

$$\mathcal{A}_i^{(1),\mathrm{fin}} = \mathcal{A}_i^{(1),\mathrm{UV}} - I_1 \mathcal{A}_i^{(0),\mathrm{UV}},$$

$$I_1 = I_1^{\mathrm{soft}} + I_1^{\mathrm{coll}},$$

$$I_1^{\mathrm{soft}} = -\frac{e^{\epsilon\gamma_E}}{\Gamma(1-\epsilon)} \left(\frac{\mu_R^2}{s}\right)^{\epsilon} \left(\frac{1}{\epsilon^2} + \frac{i\pi}{\epsilon}\right) 2C_A,$$

$$I_1^{\mathrm{coll}} = -\frac{\beta_0}{\epsilon} \left(\frac{\mu_R^2}{s}\right)^{\epsilon}.$$

A Few Conventions

We present results for the Born and Born-Virtual interference helicity amplitudes

Expand the helicity amplitudes in α_S

$$\mathscr{A}^{\text{fin}}_{\lambda_1\lambda_2\lambda_3} = \left(\frac{\alpha_s}{4\pi}\right) \mathscr{A}^{(0),\text{fin}}_{\lambda_1\lambda_2\lambda_3} + \left(\frac{\alpha_s}{4\pi}\right)^2 \mathscr{A}^{(1),\text{fin}}_{\lambda_1\lambda_2\lambda_3} + \dots$$

Compute the square/interference

$$\mathscr{B} = \frac{1}{4} \sum_{\lambda_1\lambda_2\lambda_3} |\mathscr{A}^{(0),\text{fin}}_{\lambda_1\lambda_2\lambda_3}|^2,$$

$$\mathscr{V} = \frac{1}{4} \sum_{\lambda_1\lambda_2\lambda_3} 2\,\text{Re}\left(\mathscr{A}^{*(0),\text{fin}}_{\lambda_1\lambda_2\lambda_3} \mathscr{A}^{(1),\text{fin}}_{\lambda_1\lambda_2\lambda_3}\right)$$

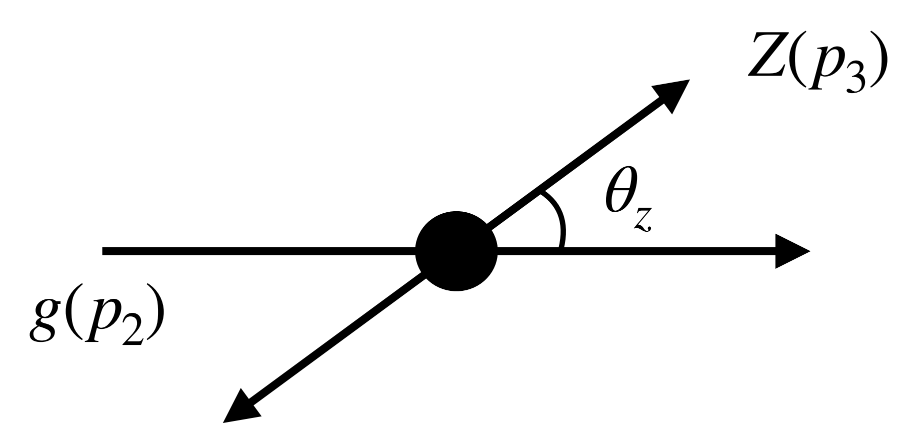
Renormalization scale set to $\mu_R^2 = s$
Electroweak coupling $e^2 = 4\pi\alpha = 1$
(can easily vary couplings/scales)

$2 \to 2$ amplitude depends on two kinematic variables (after fixing masses)

Choose:

$s = (p_1 + p_2)^2$

θ_z - angle in c.o.m frame between p_2-axis and p_3



Evaluation of the Amplitude

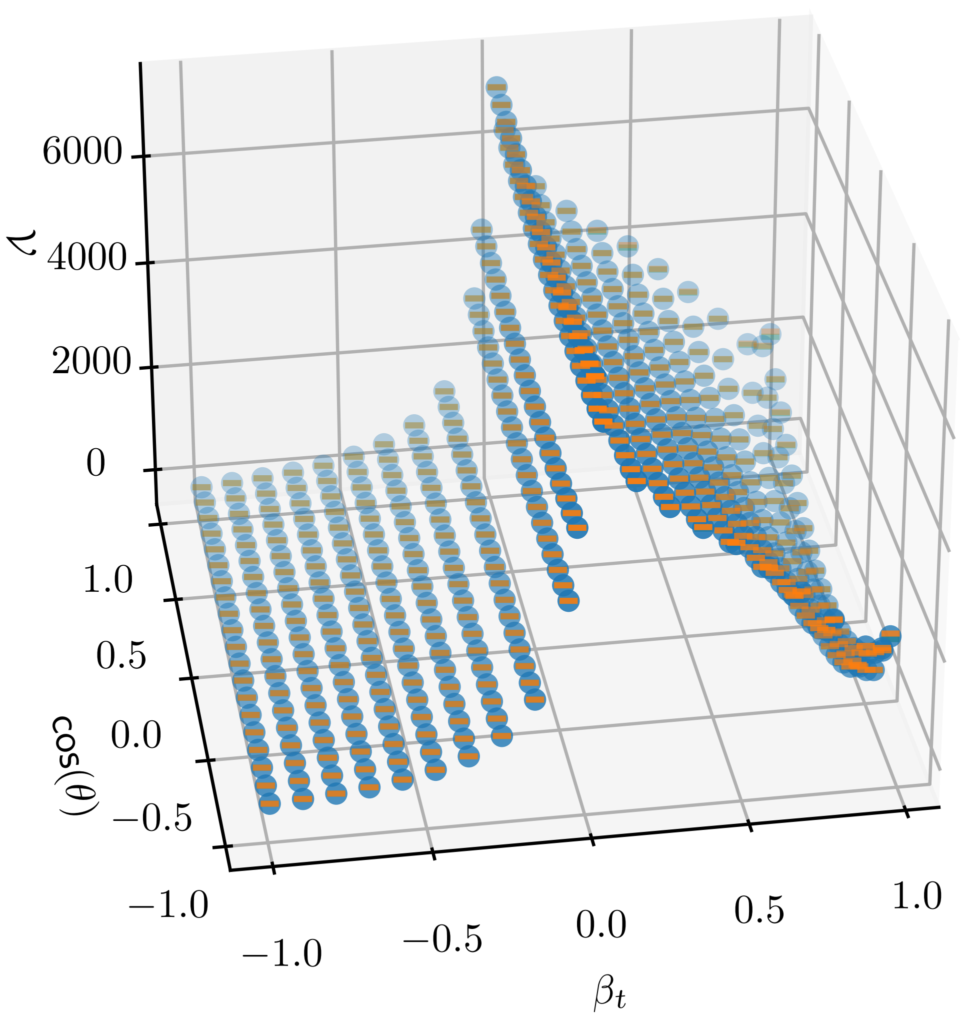
Again, $2 \rightarrow 2$ amplitude depends on two kinematic variables (after fixing masses)

$$\beta_t = \frac{s - 4m_t^2}{s + 4m_t^2 - (2m_z + m_h)^2},$$

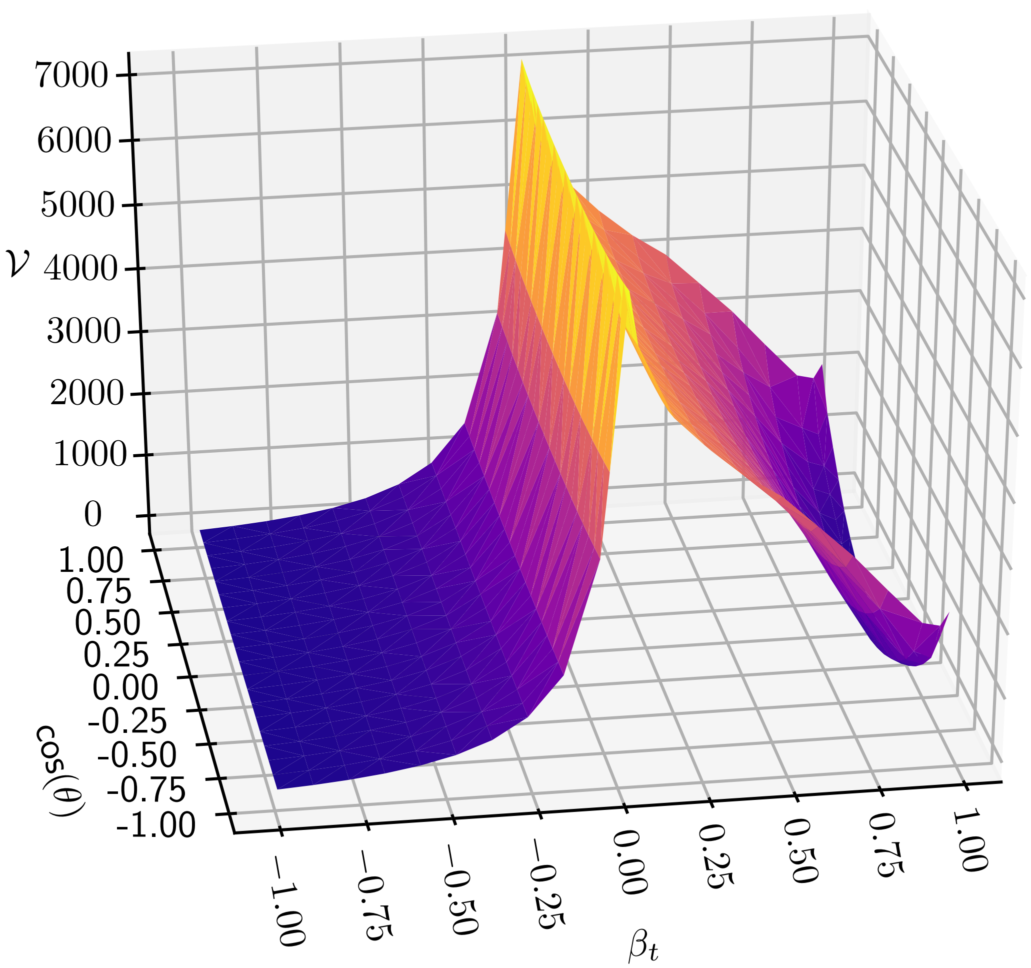
θ_z - angle in c.o.m frame between p_2-axis and Z-boson (p_3)

Sample grid of 20×20 points $+80$ extra top threshold/high-energy points in range:

$$-0.99 < \beta_t < 0.99 \text{ and } -0.99 < \cos(\theta_z) < 0.99$$



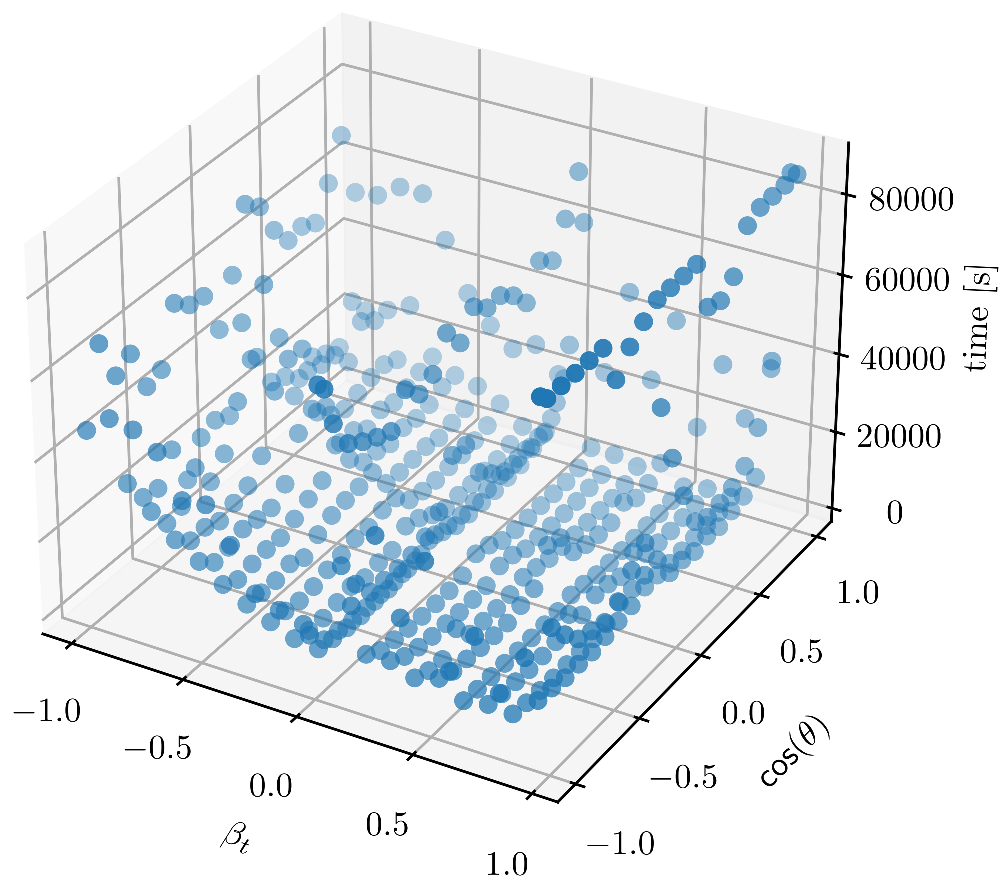
Interpolated surface (visualisation)

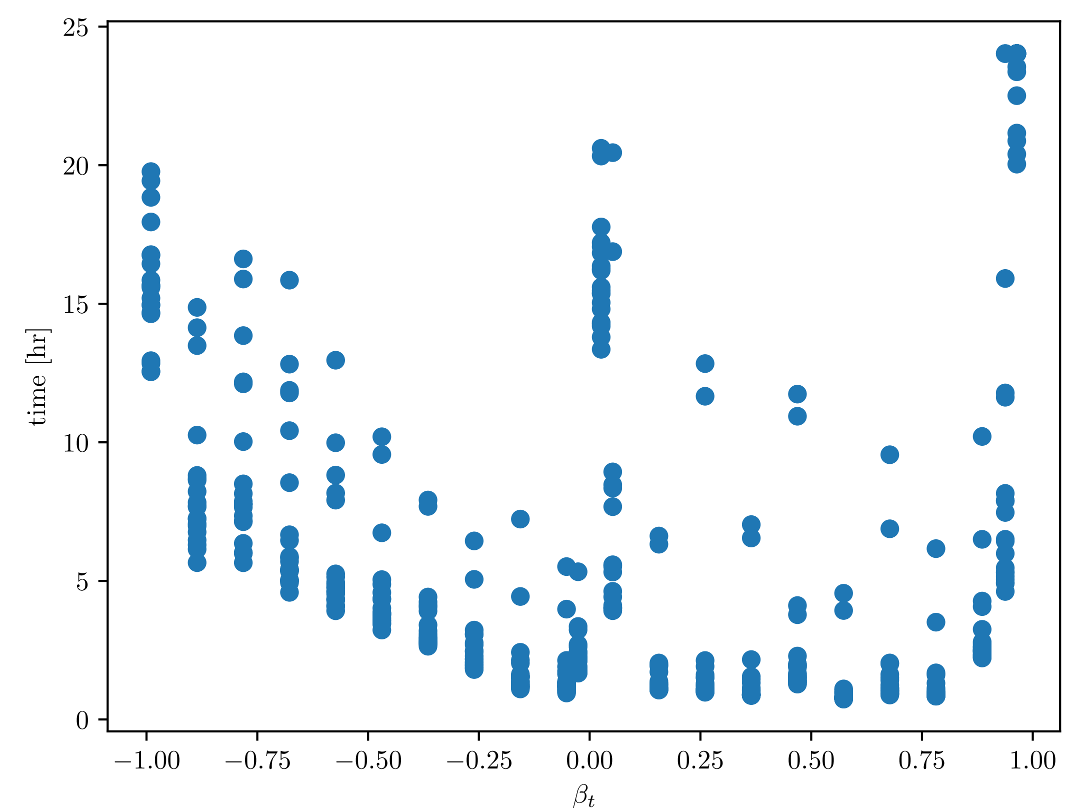


Evaluation of the Amplitude (Timing)

Each phase-space point evaluated with 2 x Nvidia Tesla V100 GPUs
Precision goal set to 0.3% for each (linearly polarised) amplitude

Timing/ point:
Min: 45 mins, **Max:** 24 hr (wall-clock), ~65 hr (high-energy), **Median:** 3.5 hr





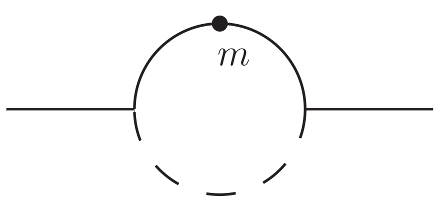
Worst performance near to ZH, $t\bar{t}$ thresholds, high-energy and forward scattering

Aside: Expansion by Regions

One option for dealing with these numerically unstable high-energy/threshold regions is expansion by regions **(not used in our ZH calculation)**

Beneke, Smirnov 98; Rakhmetov, Pak, Jantzen, Semenova, Becher, Neubert, Broggio, Ferroglia,…
(See e.g. Jantzen 11 for an introduction)

Idea: expand integrals around some small parameter, e.g. m^2/p^2



$$= \mu^{2\epsilon} \int dk \, \frac{1}{(k+p)^2(k^2-m^2)^2}$$

$$(\text{hard}): \qquad |k^2| \gg m^2, \qquad \frac{1}{(k+p)^2(k^2-m^2)^2} \rightarrow \frac{1}{(k+p)^2\,(k^2)^2}\left(1 + 2\frac{m^2}{k^2} + \ldots\right)$$

$$(\text{soft}): \qquad |k^2|, |k\cdot p| \ll p^2, \qquad \frac{1}{(k+p)^2(k^2-m^2)^2} \rightarrow \frac{1}{p^2\,(k^2-m^2)^2}\left(1 - \frac{k^2+2p\cdot k}{p^2} + \ldots\right)$$

Integrate expanded integrals over full integration range, sum over all regions

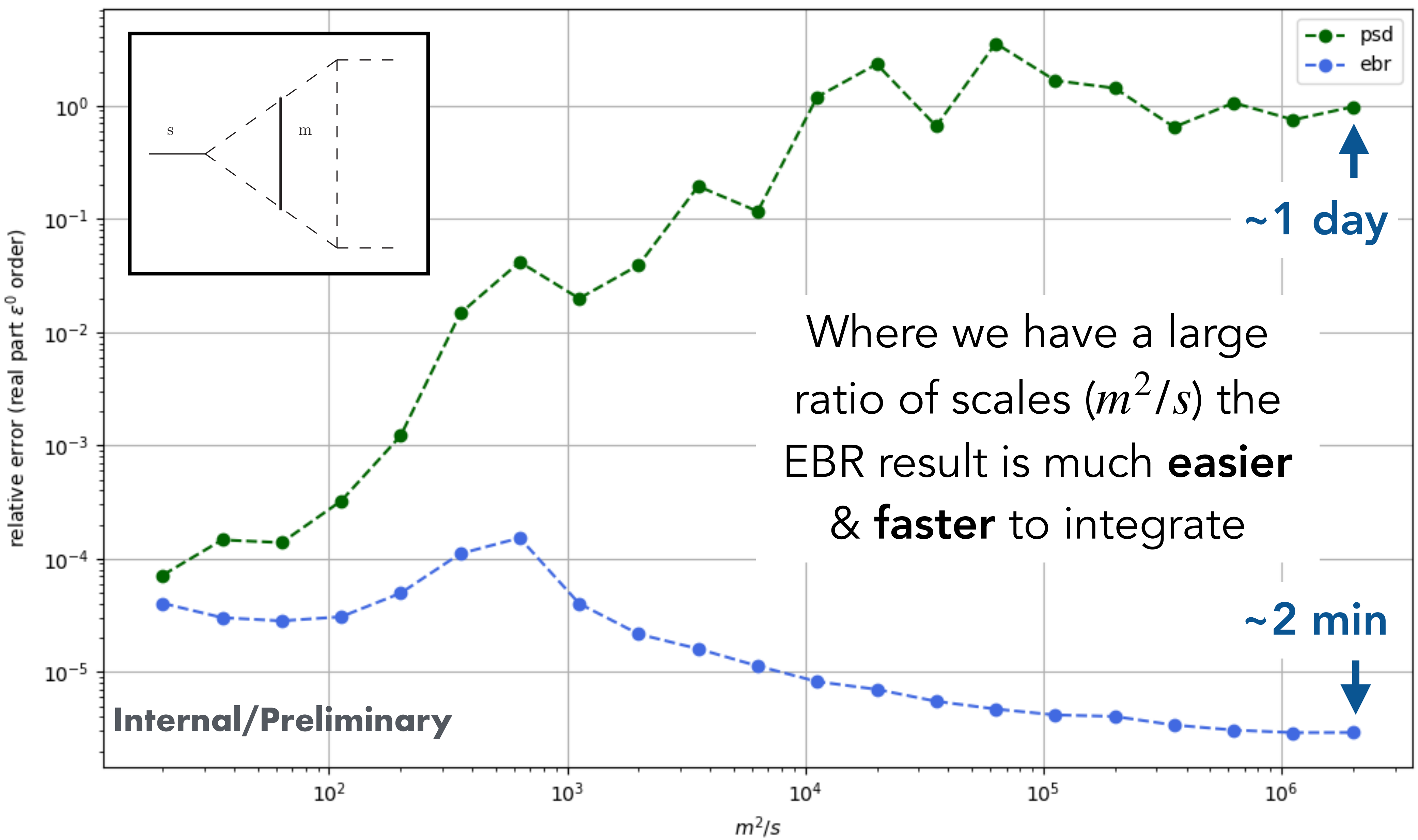
Concept can be systematically applied also in Feynman parameter space

Aside: Expansion by Regions (II)

Implemented in tools such as FIESTA/ ASY/ ASY2

Smirnov 15;
Smirnov, Smirnov, Tentyukov 09;

Available in next release of pySecDec



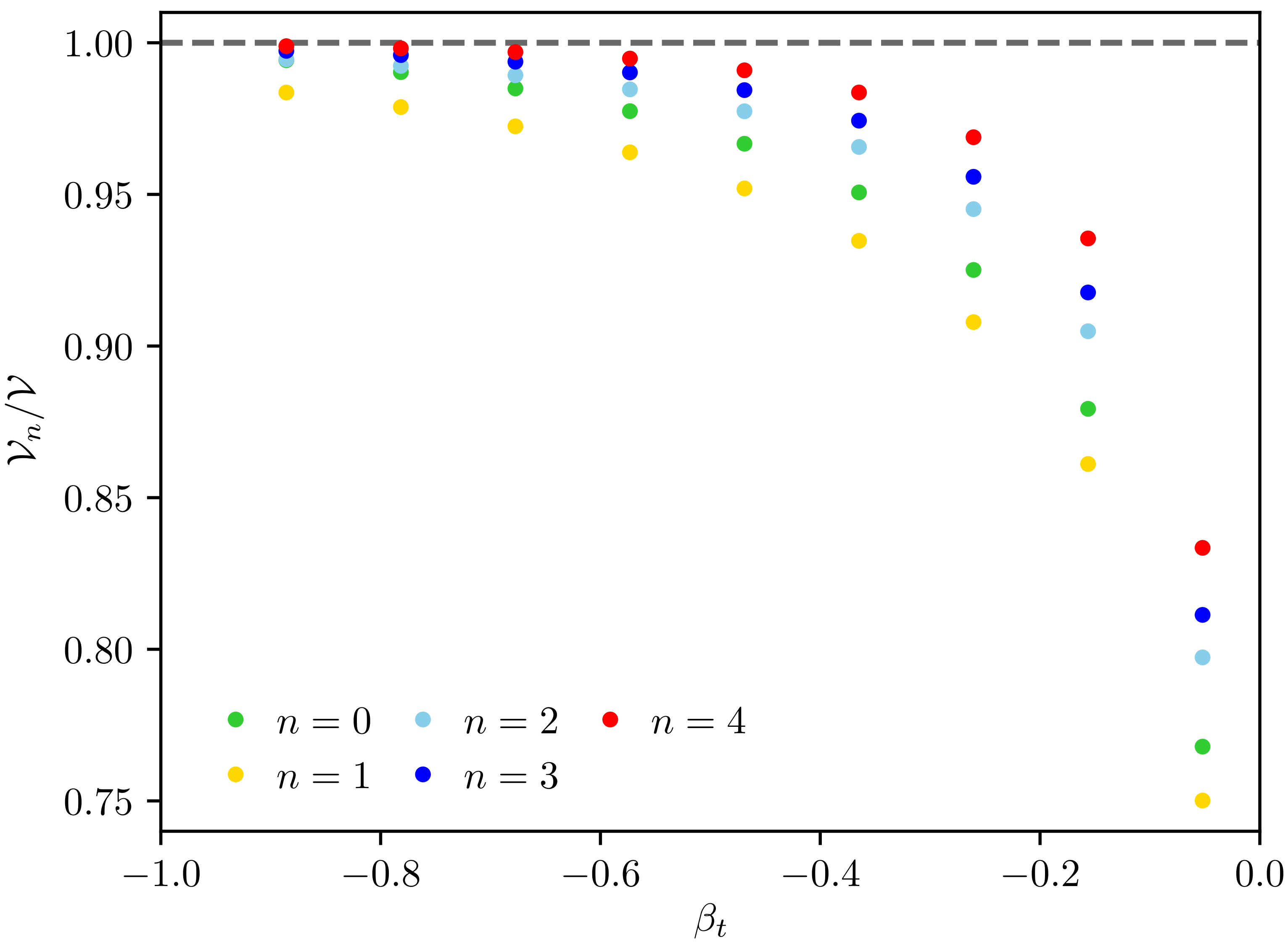
Details in talk of: Gudrun Heinrich @ HU Berlin 28/06/21

Comparison to Expansion (Large m_t)

The amplitude has been expanded around large-m_t and computed analytically
Hasselhuhn, Luthe, Steinhauser 17; Davies, Mishima, Steinhauser 20

Let us compare our result to the Born reweighted $1/m_t^{2n}$ expansion:

$$\mathcal{V}_n = \frac{\mathcal{B}}{\mathcal{B}_n}\widetilde{\mathcal{V}}_n + \mathcal{V}^{\mathrm{1PR}}$$



Per mille level agreement far below top quark threshold:

$$\mathcal{V}_4/\mathcal{V} = 0.9989$$

Expansion breaks down at threshold, observe that it differs from our result

Observation: $n = 1$ apparently worse than $n = 0$

A Note on ~~High-Energy~~ Small Mass Expansions

Each form factor can be expanded in m_z^2, m_h^2, m_t^2:

Davies, Mishima, Steinhauser 20; Mishima 18

1) Express amplitude in terms of Feynman integrals:
   $I(m_z^2, m_h^2, m_t^2)$

2) Taylor expand around $m_z^2 = 0$ and $m_h^2 = 0$:
   $I(0{,}0{,}m_t^2) + m_z^2 I'(0{,}0{,}m_t^2) + \ldots$

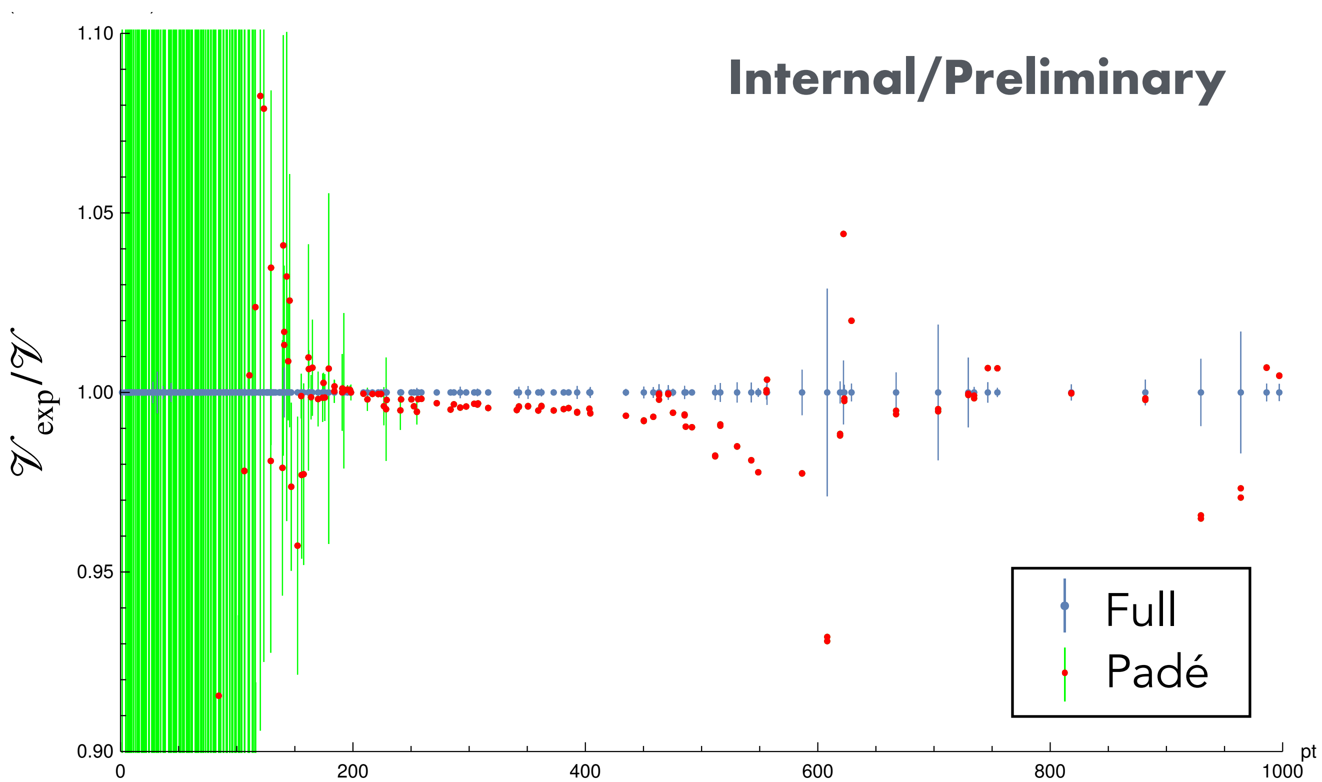
3) IBP reduce Feynman integrals to master integrals:
   $J(0{,}0{,}m_t^2)$

4) Compute master integrals in expansion around $m_t^2 = 0$:
   $$J(0{,}0{,}m_t^2) = \sum_{m,n} C_{m,n}\,(m_t^2)^m \,\log(m_t^2)^n$$

Results expanded up to m_t^{32}

Expansion expected to be valid for: $m_z^2, m_h^2 \ll m_t^2 \ll s, |t|, |u|$

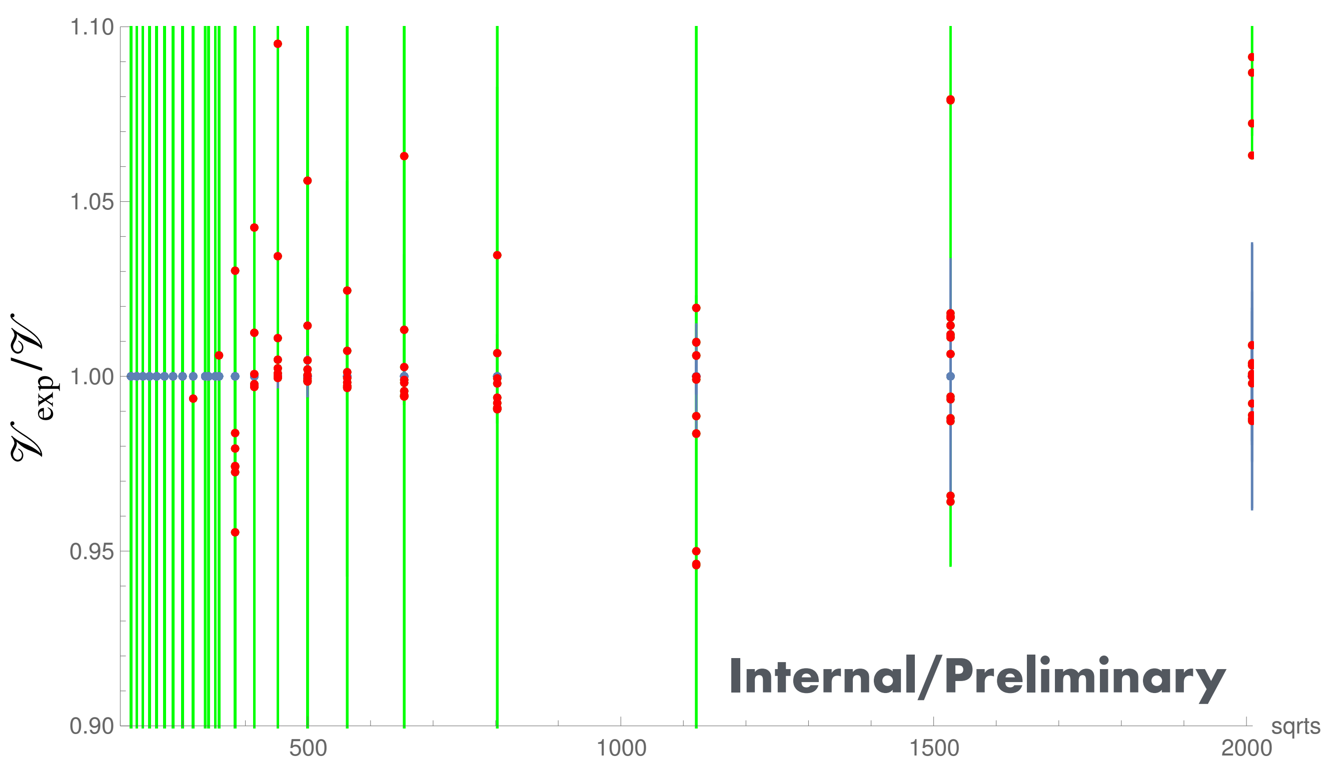
Comparison to Expansion (Small m_t)



Agreement:
~2% level for $p_T \gtrsim 225$
~10% level for $150 < p_T < 225$

Consistent with Padé/ full at LO level

Currently investigating behaviour at moderate/ larger p_T (m_z^4, m_h^4 effects? error propagation?)



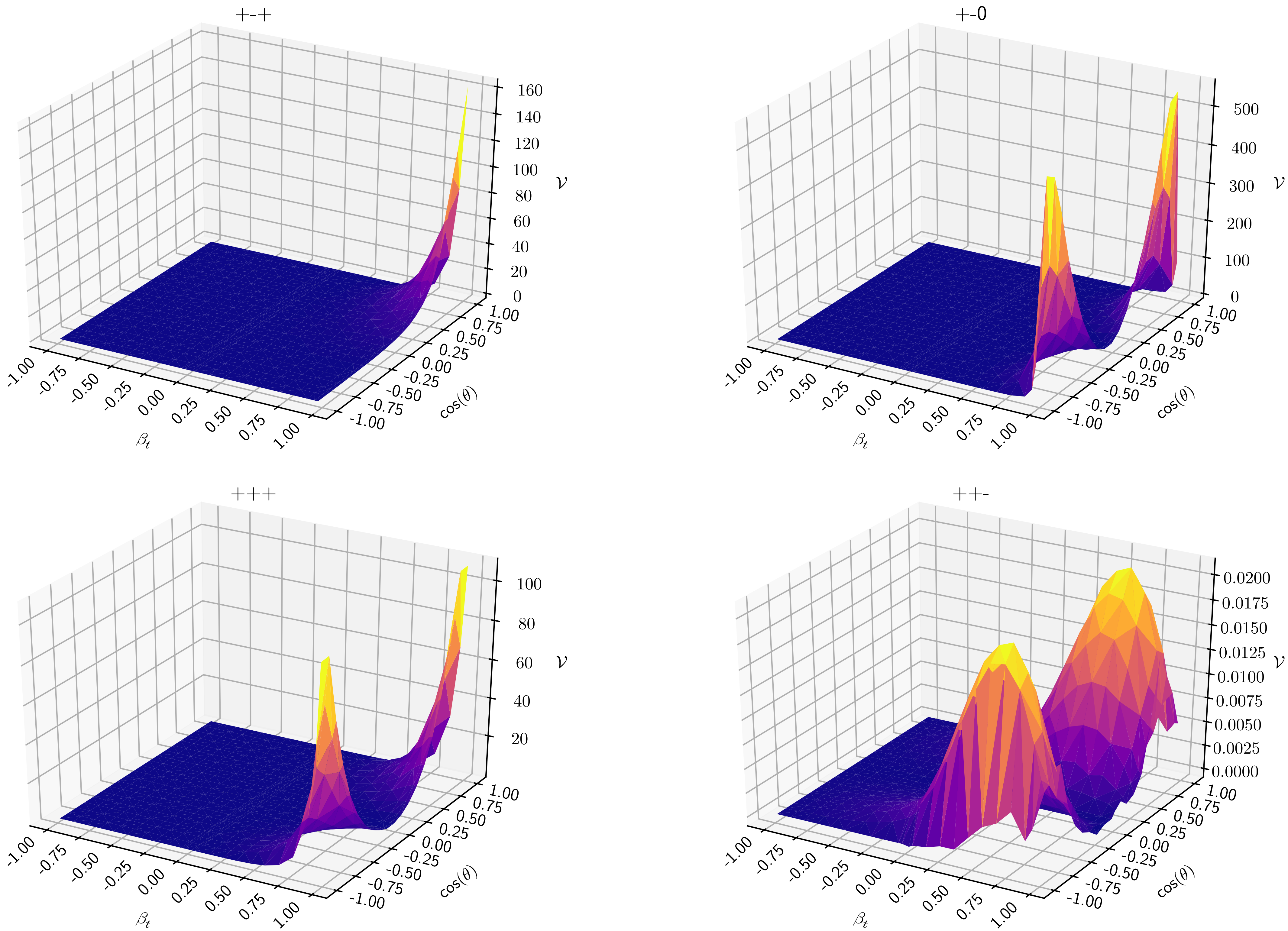
Plots courtesy of: Davies, Mishima, Steinhauser

Not all points agreeing well even for large $\sqrt{s}$

Convergence of the Padé result depends on $m_T \ll s, |t|, |u|$ can have small $|t|$ even for large s (if p_T is small)

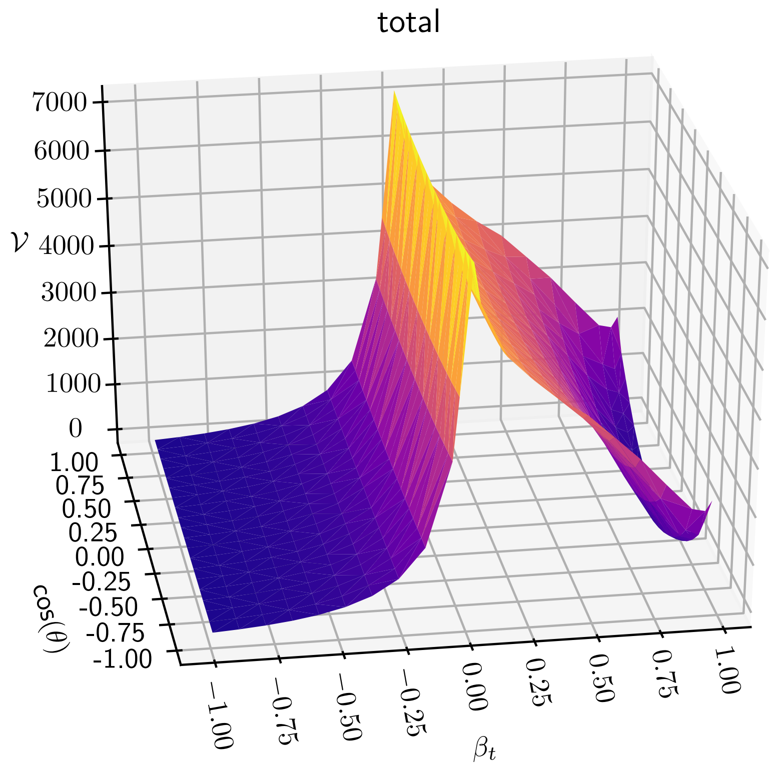
Helicity Amplitudes

Possible to produce precise results for all helicity amplitudes also in kinematic limits!

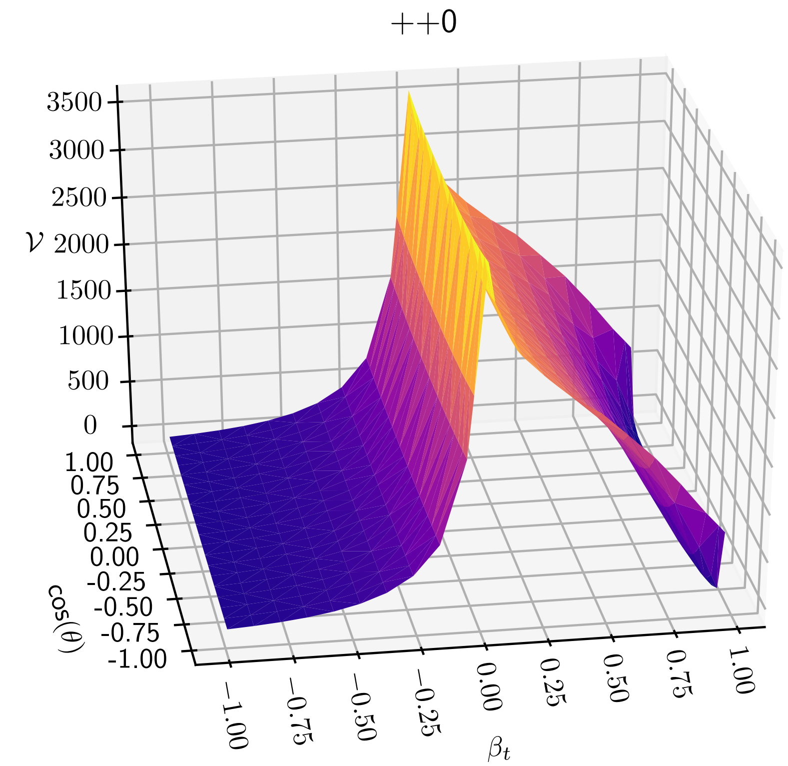


Comparison to Expansion

Observe that modes with longitudinally polarised Z boson dominate total

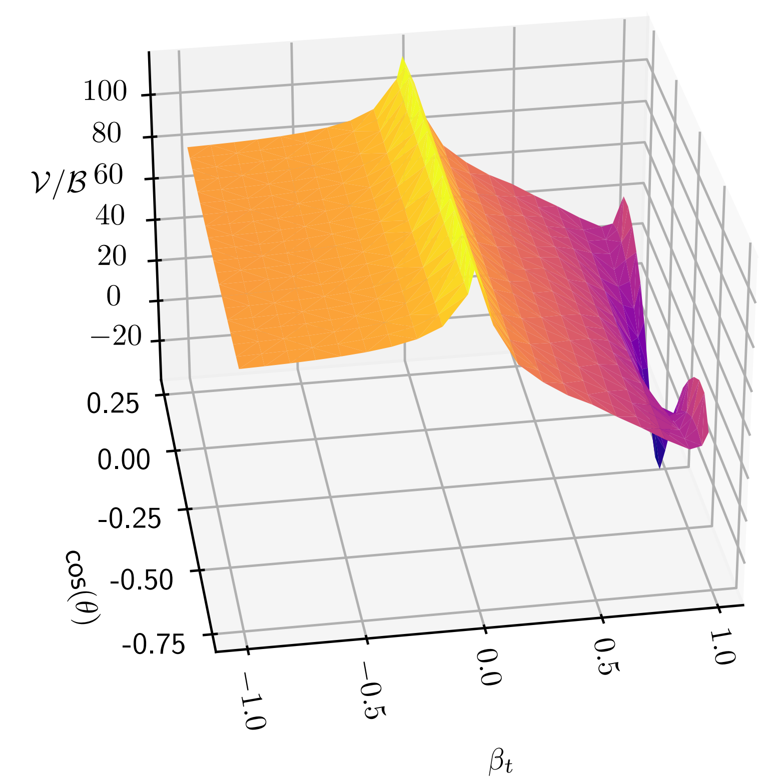


$\sim 2\times$



Ratio between Born squared amplitude and Born-Virtual interference not flat

Reminder: missing real contribution (so plot is not NLO/LO `K-factor')



Conclusion

We have entered the precision Higgs era

- Over coming years, expect greater demand for precise theory predictions (required to exploit experimental measurements)
- Detailed searches for deviations, e.g: differential measurements, constraining EFT couplings, off-shell measurements, …

I have presented a calculation which underscores the usefulness of

- Modern approaches to solving IBPs and integral basis choices
- Numerical methods for solving Feynman integrals **(new pySecDec release soon!)**

Next steps…

- Put the pieces together in order to obtain complete NLO results
- Incorporate into public tools for $pp \to ZH$ (?)

Thank you for listening!

Backup

Results $gg \to ZZ$

Firstly: A Few Conventions

We present results for the Born and Born-Virtual interference helicity amplitudes

Expand the helicity amplitudes in α_S

$$\mathscr{M}^{fin}_{\lambda_1\lambda_2\lambda_3\lambda_4} = \left(\frac{\alpha_s}{2\pi}\right)\mathscr{M}^{(1)}_{\lambda_1\lambda_2\lambda_3\lambda_4} + \left(\frac{\alpha_s}{2\pi}\right)^2\mathscr{M}^{(2)}_{\lambda_1\lambda_2\lambda_3\lambda_4} + \ldots$$

Compute the square/interference

$$\mathscr{V}^{(1)}_{\lambda_1\lambda_2\lambda_3\lambda_4} = |\mathscr{M}^{(1)}_{\lambda_1\lambda_2\lambda_3\lambda_4}|^2,$$

$$\mathscr{V}^{(2)}_{\lambda_1\lambda_2\lambda_3\lambda_4} = 2\,\mathrm{Re}\left(\mathscr{M}^{*(1)}_{\lambda_1\lambda_2\lambda_3\lambda_4}\mathscr{M}^{(2)}_{\lambda_1\lambda_2\lambda_3\lambda_4}\right)$$

We also define

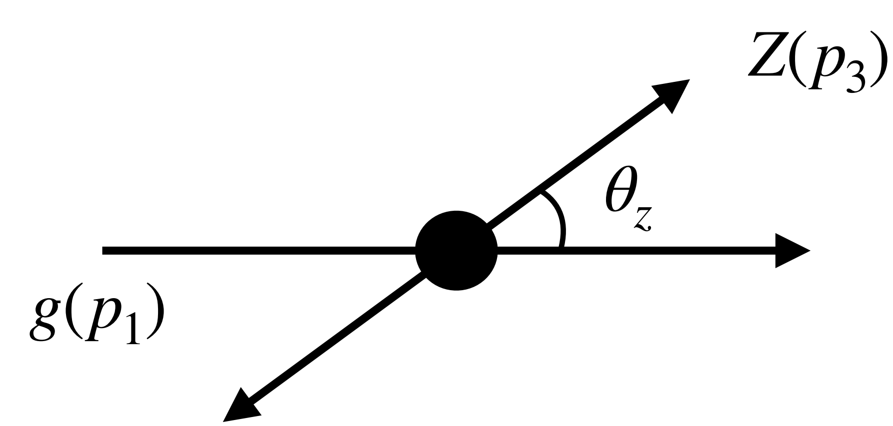
$$\mathscr{V}^{(i)}_{\lambda_3\lambda_4} = \frac{1}{4}\sum_{\lambda_1,\lambda_2}\mathscr{V}^{(i)}_{\lambda_1\lambda_2\lambda_3\lambda_4} \quad \text{and} \quad \mathscr{V}^{(i)} = \sum_{\lambda_3,\lambda_4}\mathscr{V}^{(i)}_{\lambda_3\lambda_4}$$

$2 \to 2$ amplitude depends on two kinematic variables (after fixing masses)

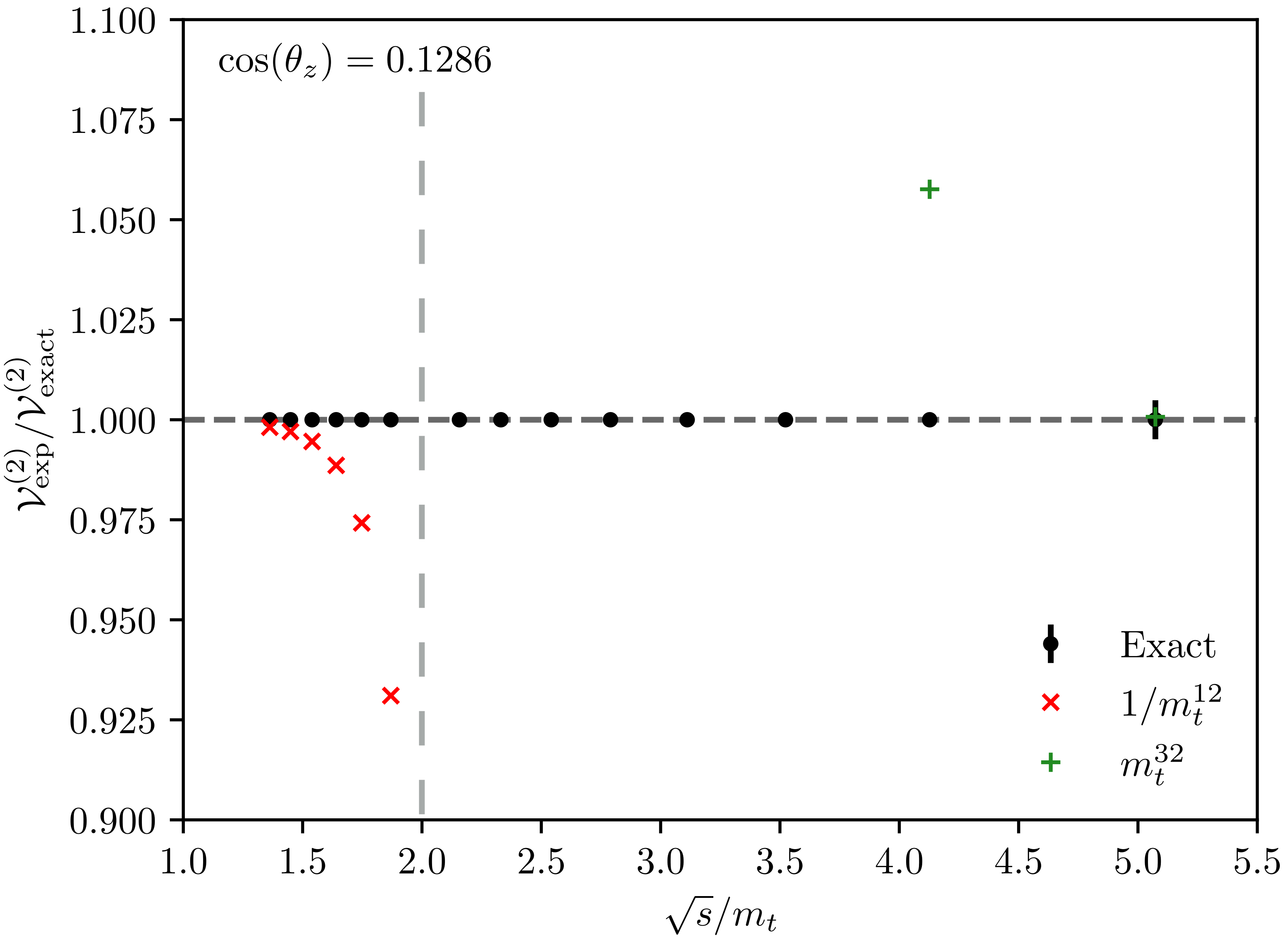
Choose:

$s = (p_1 + p_2)^2$

θ_z - angle in c.o.m frame between p_1-axis and p_3



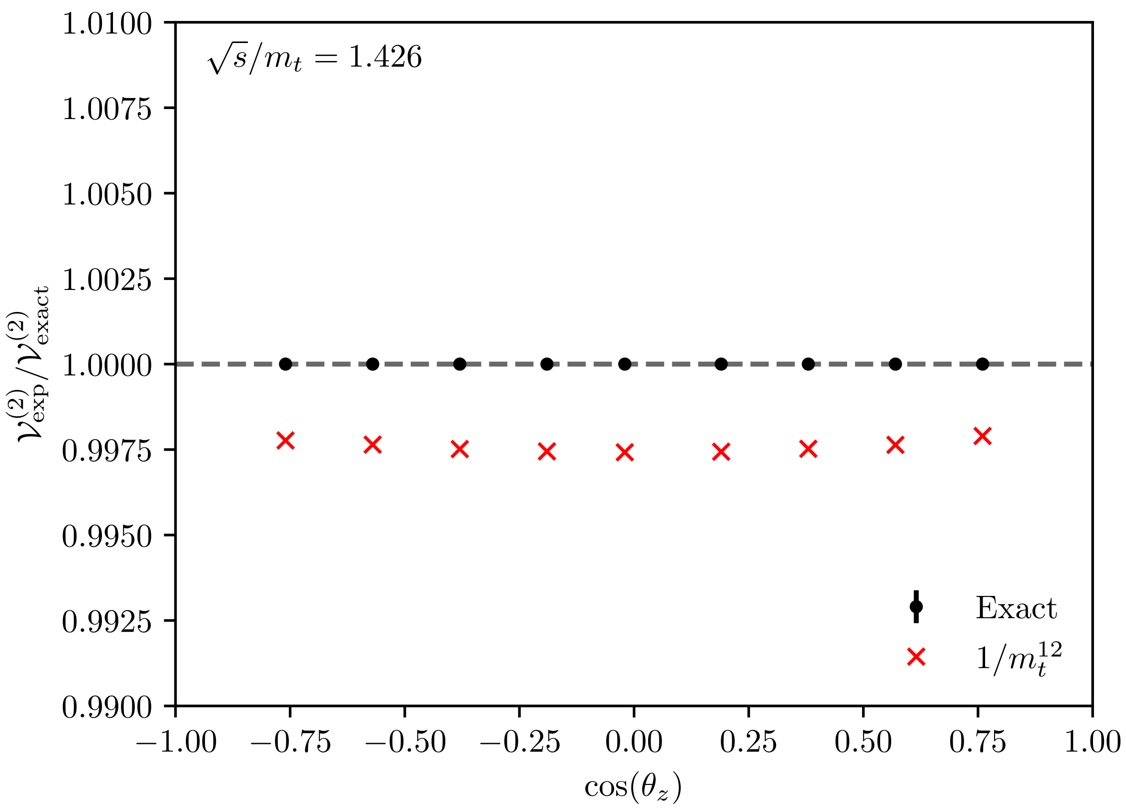
Comparison to Expansion



Expanded results from: Davies, Mishima, Steinhauser, Wellmann 20

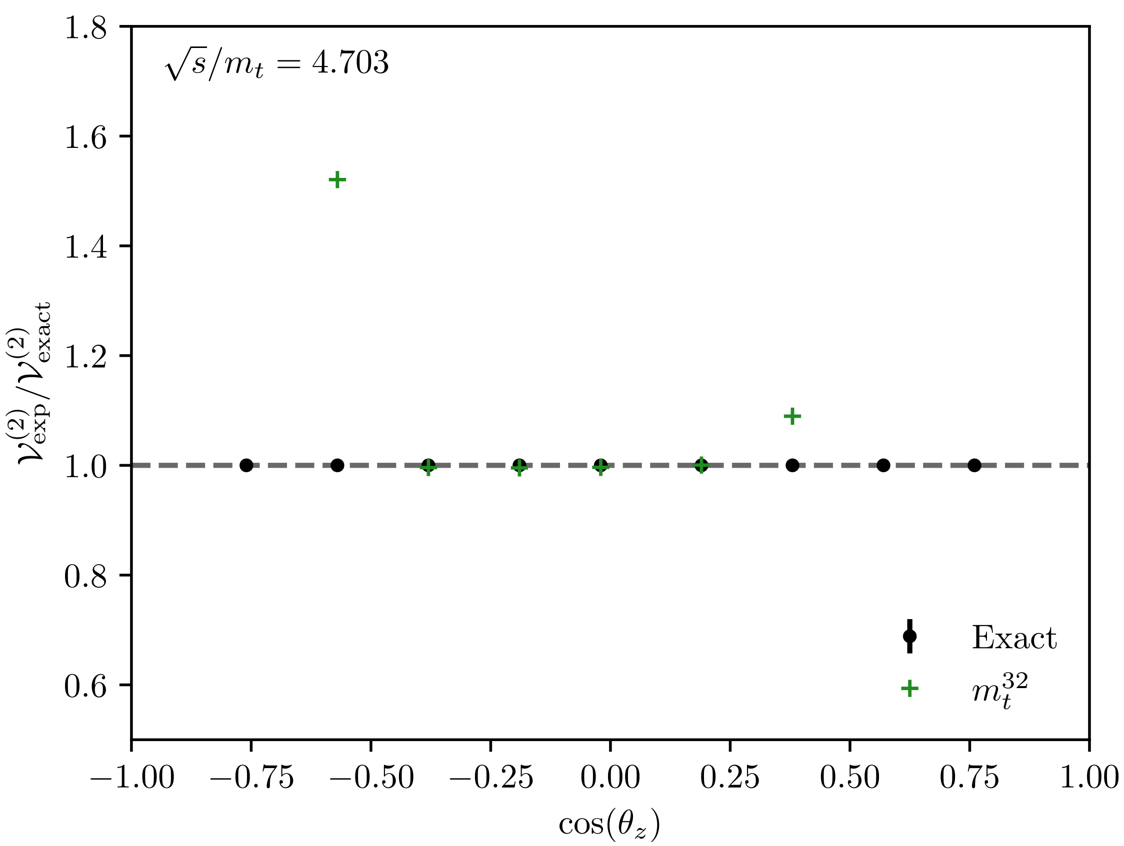
Expansion can be likely be improved by fitting Padé approximants & conformal mapping, have not compared to these approaches (yet) Campbell, Ellis, Czakon, Kirchner 16

Comparison to Expansion II



Large m_t expansion

Excellent agreement with expansion sufficiently below top quark threshold (also as a function of θ_z)



Small m_t, m_z expansion

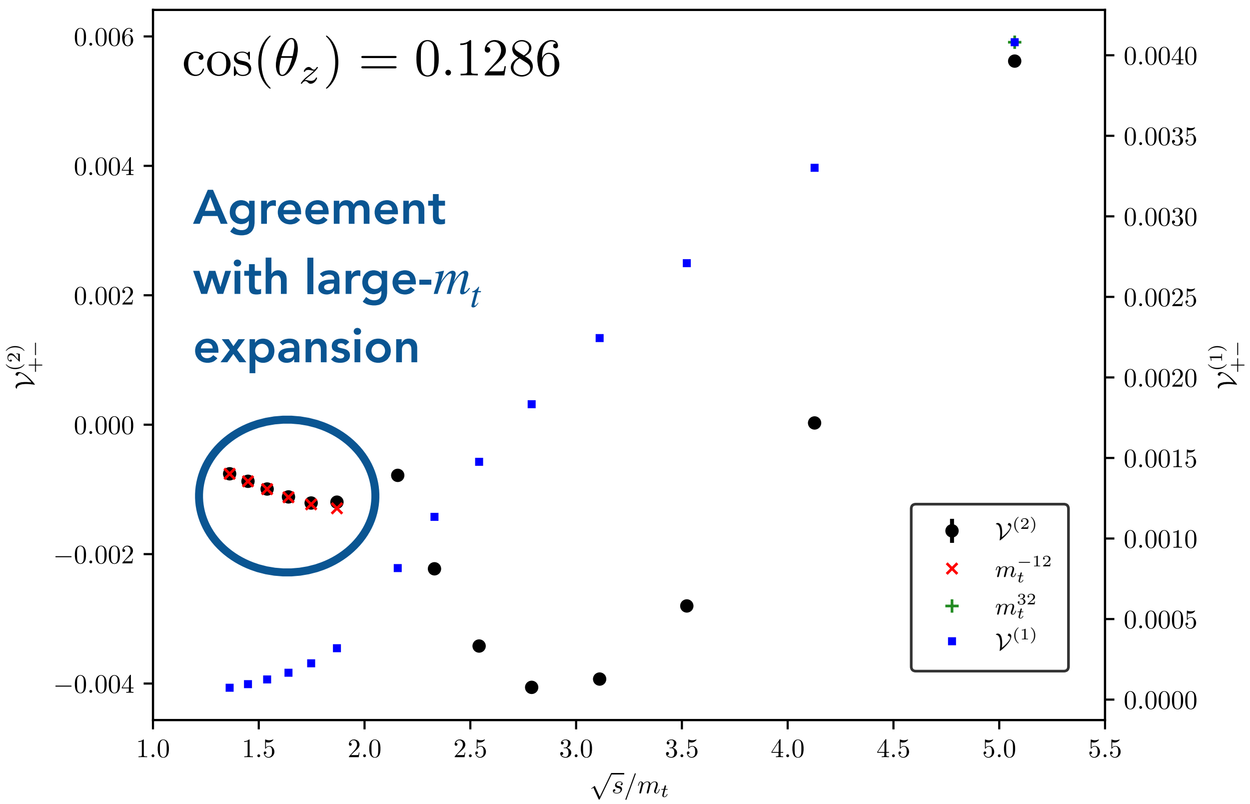
Excellent agreement for small $\cos(\theta_z)$

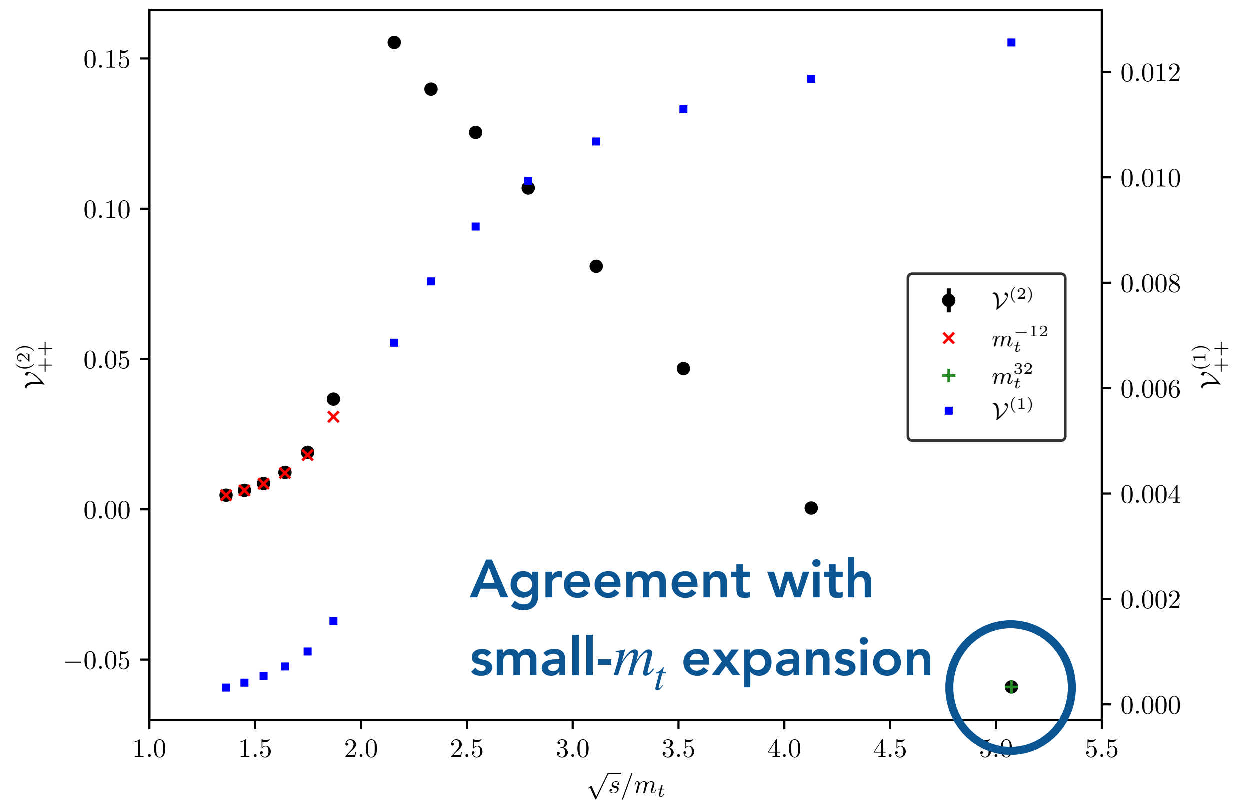
The small m_t expansion assumes:

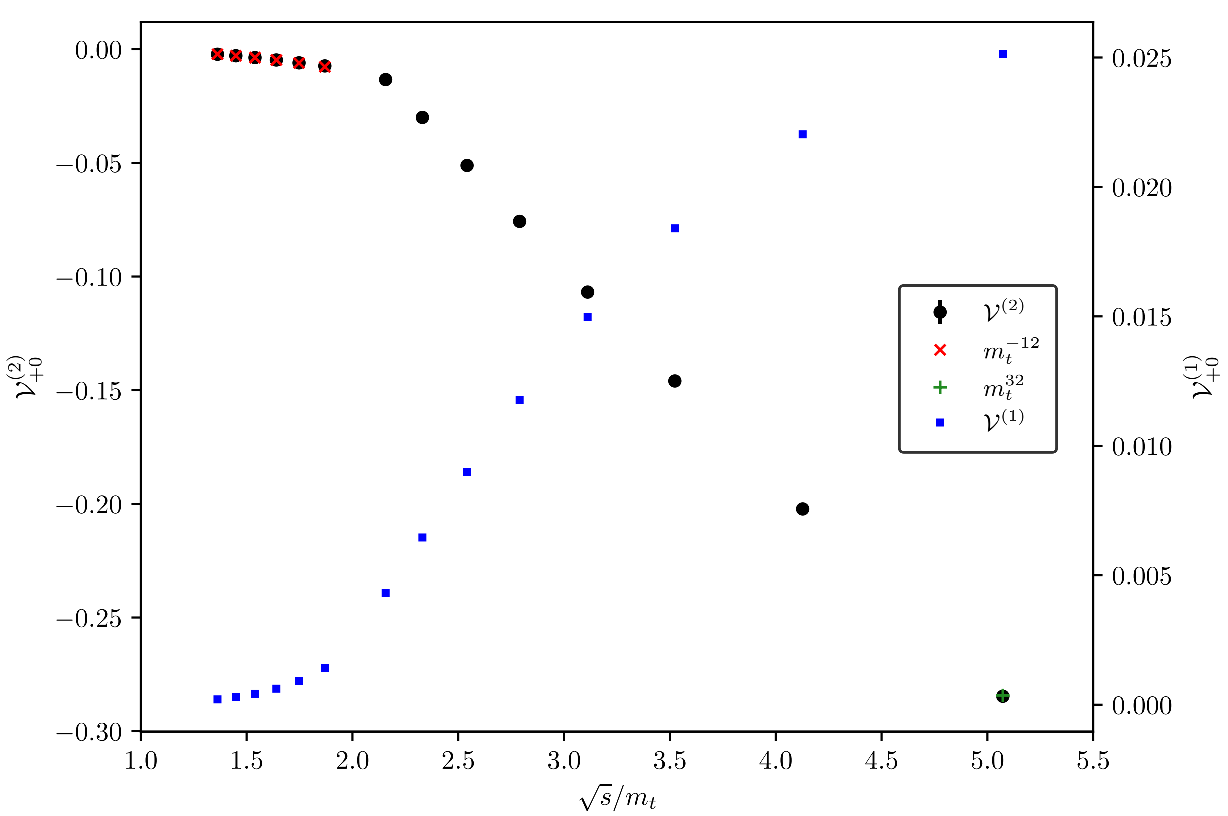
$m_t^2 \ll s, |t|, |u|$ (all invariants in problem)

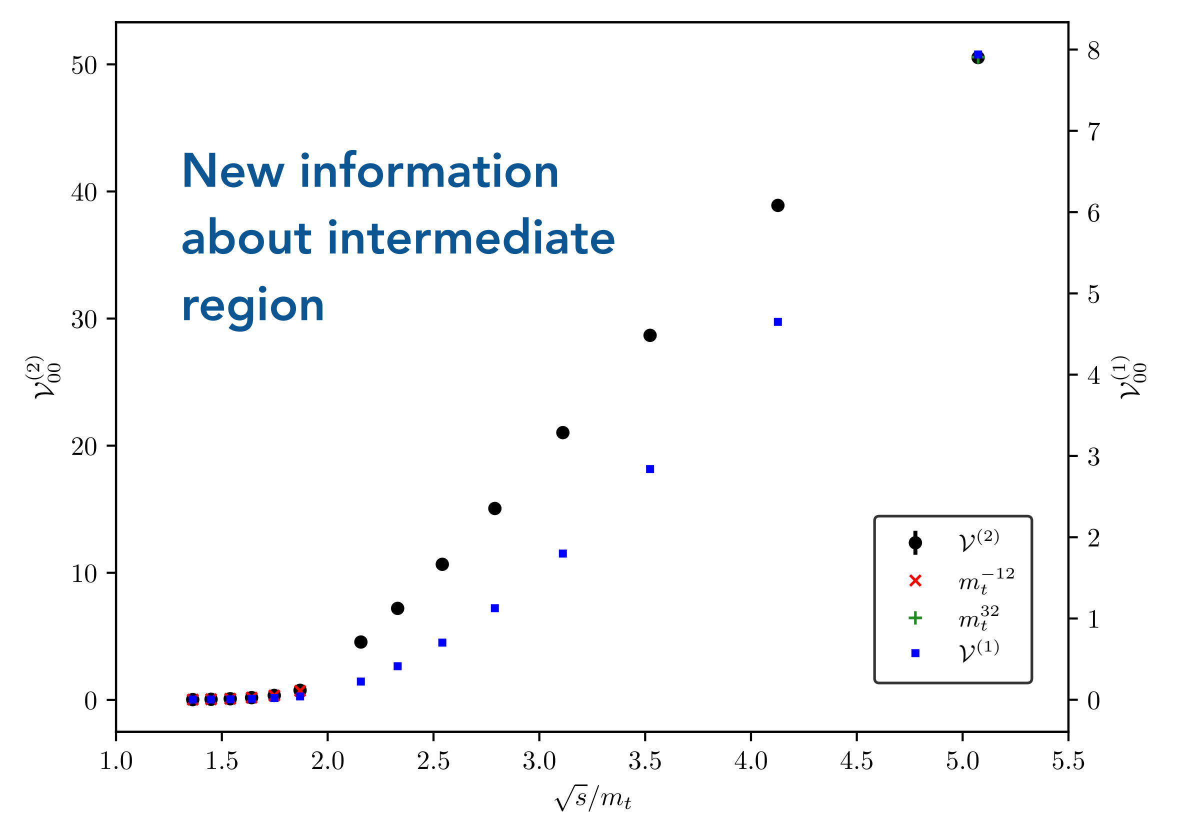
For small scattering angle we do not necessarily have $m_t^2 \ll |t|, |u|$ and we can expect the expansion to break down

Helicity Amplitudes

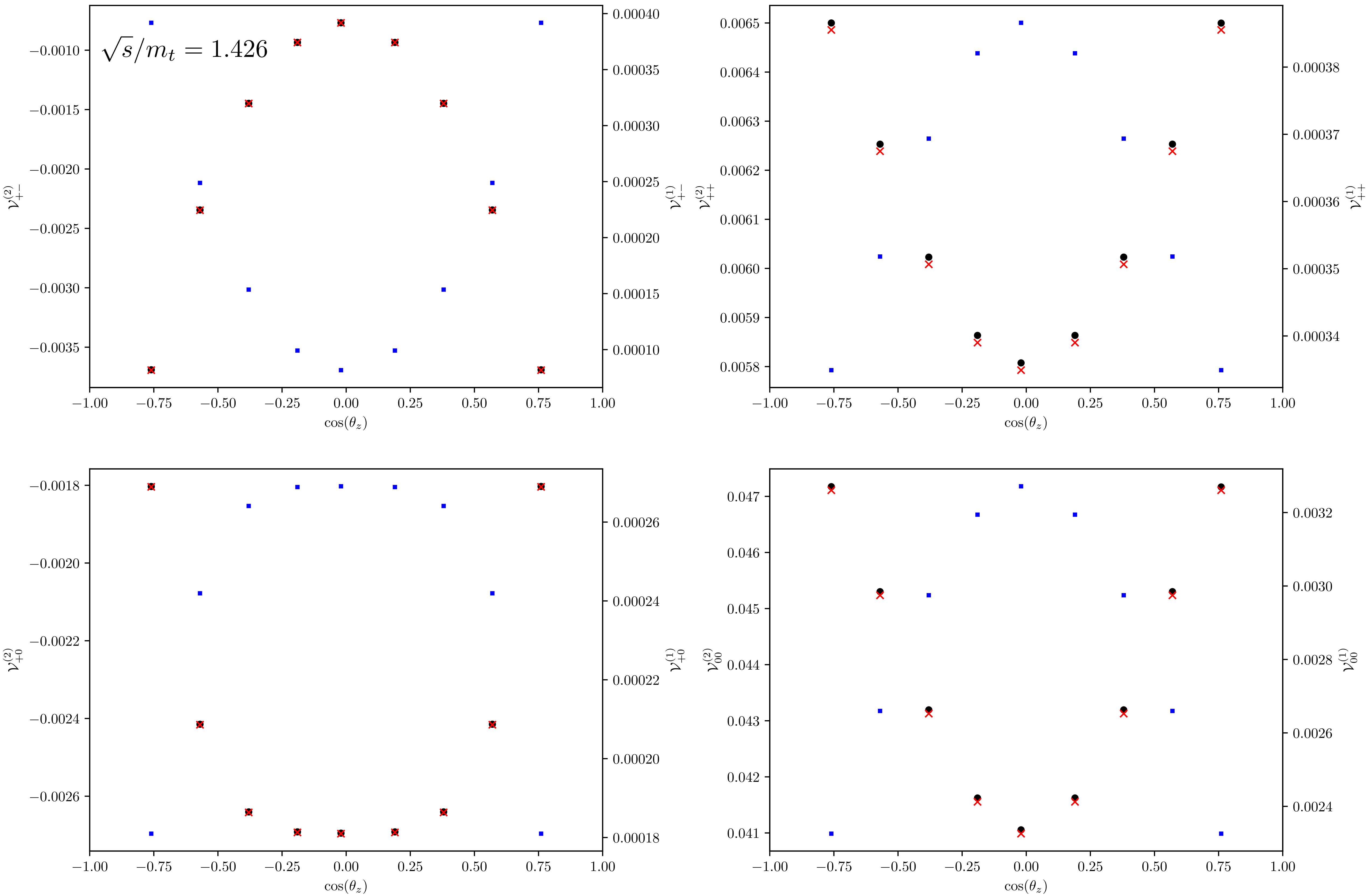




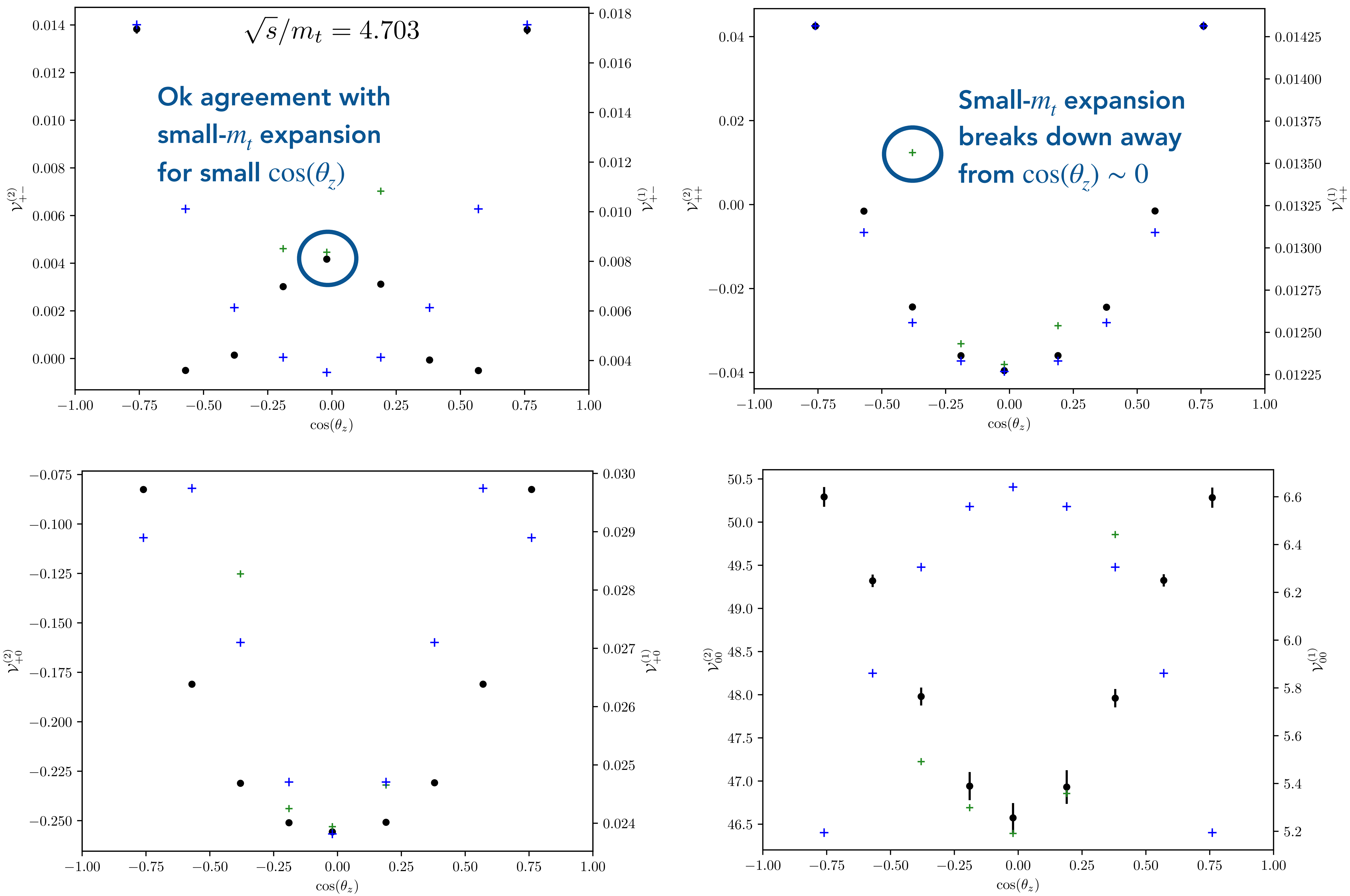




Helicity Amplitudes



Helicity Amplitudes



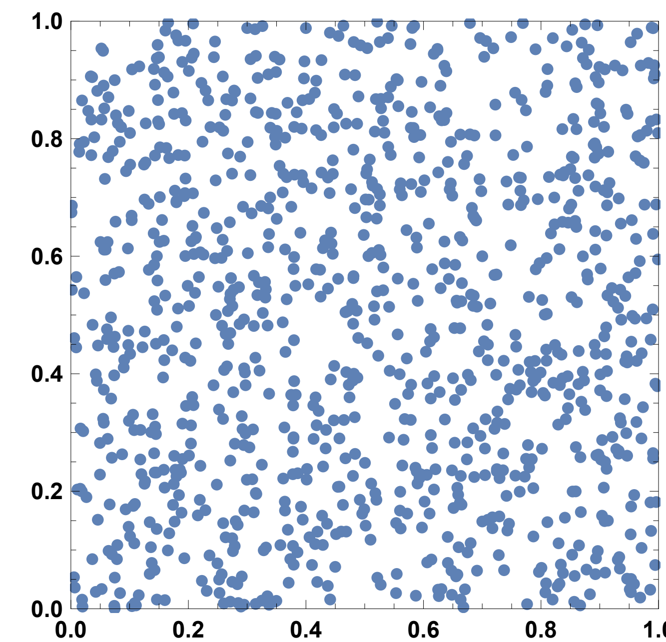
Numerical Integration

Numerical Integration

$$I[f] \equiv \int_{[0,1]^d} \mathrm{d}\mathbf{x}\ f(\mathbf{x}) \quad \approx \quad Q[f] = \frac{1}{N}\sum_{i=1}^{N} w_i\ f(\mathbf{x}_i)$$

Goal: select points to minimise integration error

$$\varepsilon \equiv |I[f] - Q[f]|$$

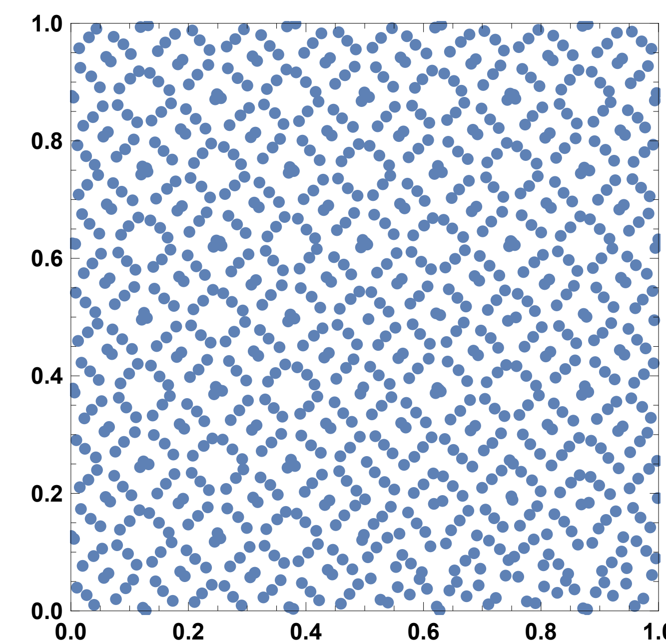


Monte Carlo:

Randomly select sampling points

$$\varepsilon \approx \mathrm{Var}[f]/\sqrt{N}, \quad \varepsilon \sim \mathcal{O}(N^{-1/2})$$

Improves slowly with N



Quasi-Monte Carlo

Select points with low discrepancy D_N

$$\varepsilon \le D_N \cdot \mathrm{Var}[f], \quad \varepsilon \sim \mathcal{O}(\log^d(N)/N)$$

Poor performance for large d

Both methods implemented in Cuba Hahn 04; Hahn14

Quasi-Monte Carlo (Rank 1 Lattices)

Quasi-Monte Carlo (QMC) in a Weighted Function Space

First applications to loop integrals, see: Li, Wang, Yan, Zhao 15; de Doncker, Almulihi, Yuasa 17, 18; de Doncker, Almulihi 17; Kato, de Doncker, Ishikawa, Yuasa 18

$\varepsilon \leq e_{\boldsymbol{\gamma}} \cdot ||f||_{\boldsymbol{\gamma}}, \quad \varepsilon \sim \mathcal{O}(N^{-1})$ or better

$$I[f] \approx \bar{Q}_{n,m}[f] \equiv \frac{1}{m} \sum_{k=0}^{m-1} Q_n^{(k)}[f], \quad Q_n^{(k)}[f] \equiv \frac{1}{n} \sum_{i=0}^{n-1} f\left(\left\{ \frac{i\mathbf{z}}{n} + \boldsymbol{\Delta}_k \right\}\right)$$

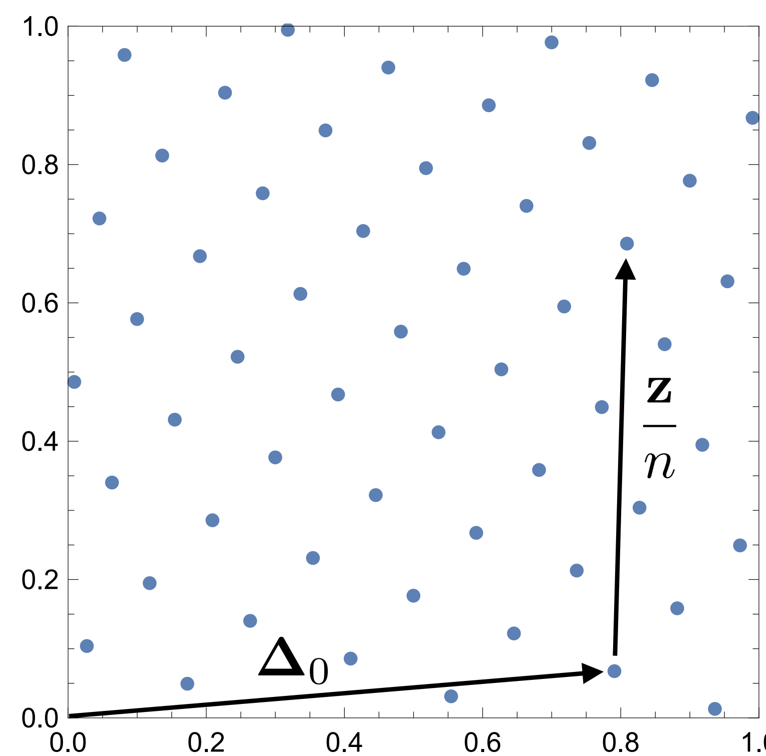
$\mathbf{z}$ - Generating vec.

$\boldsymbol{\Delta}_k$ - Random shift vec.

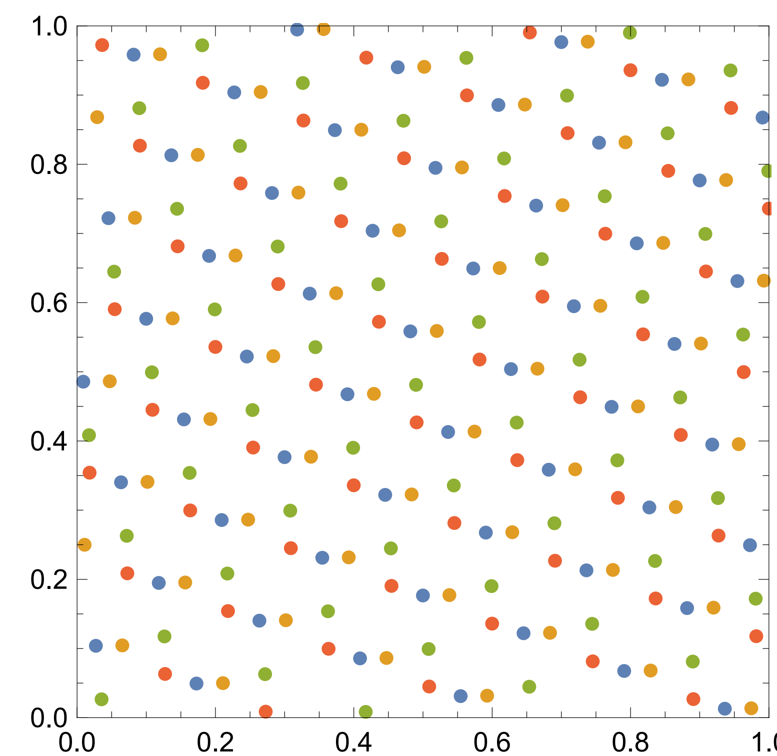
$\{\}$ - Fractional part

n - # Lattice points

m - # Random shifts



1 Shift



4 Shifts

Unbiased error estimate computed using (10-50) random shifts

Weighted Function Spaces

Assign weights $\gamma_{\mathfrak{u}}$ to each subset of dimensions $\mathfrak{u} \subseteq \{1, \dots, d\}$ Review: Dick, Kuo, Sloan 13

Sobolev Space

Functions with square integrable first derivatives

Norm
$$||f||^2_{\boldsymbol{\gamma}} = \sum_{\mathfrak{u} \subseteq \{1,\dots,d\}} \frac{1}{\gamma_{\mathfrak{u}}} \int_{[0,1]^{|\mathfrak{u}|}} \left(\int_{[0,1]^{d-|\mathfrak{u}|}} \frac{\partial^{|\mathfrak{u}|} f(\mathbf{x})}{\partial \mathbf{x}_{\mathfrak{u}}} \mathrm{d}\mathbf{x}_{-\mathfrak{u}} \right)^2 \mathrm{d}\mathbf{x}_{\mathfrak{u}}$$

Worst-case error
$$e^2_{\boldsymbol{\gamma}} \leq \left(\frac{1}{\psi(n)} \sum_{\emptyset \neq \mathfrak{u} \subseteq \{1,\dots,d\}} \gamma_{\mathfrak{u}}^{\lambda} \left(\frac{2\zeta(2\lambda)}{(2\pi^2)^{\lambda}} \right)^{|\mathfrak{u}|} \right)^{\frac{1}{\lambda}}$$

$\forall \lambda \in (1/2, 1]$

$$\boxed{\varepsilon \sim \mathcal{O}(n^{-1})}$$

Korobov Space

Periodic functions which are α times differentiable in each variable

$$||f||^2_{\boldsymbol{\gamma}} = \sum_{\mathbf{h} \in \mathbb{Z}^d} \frac{\prod_{j \in \mathfrak{u}(\mathbf{h})} |h_j|^{2\alpha}}{\gamma_{\mathfrak{u}(\mathbf{h})}} |\hat{f}(\mathbf{h})|^2$$

Fourier Coefficient

$$e^2_{\boldsymbol{\gamma}} \leq \left(\frac{1}{\psi(n)} \sum_{\emptyset \neq \mathfrak{u} \subseteq \{1,\dots,d\}} \gamma_{\mathfrak{u}}^{\lambda} \left(2\zeta(2\alpha\lambda)\right)^{|\mathfrak{u}|} \right)^{\frac{1}{\lambda}}$$

$\forall \lambda \in (1/(2\alpha), 1]$, smoothness α

$$\boxed{\varepsilon \sim \mathcal{O}(n^{-\alpha})}$$

Generating vector $\mathbf{z}$ precomputed for a **fixed** number of lattice points, chosen to minimise the worst-case error, we use component-by-component (CBC) construction

Nuyens 07

In our public code, we distribute lattice rules generated using product weights:
$\gamma_{\mathfrak{u}} = \prod_{i \in \mathfrak{u}} \gamma_i, \;\; \gamma_i = 1/d$ produced for a Korobov space with $\alpha = 2$

Periodising Transforms

Lattice rules work especially well for continuous, smooth and periodic functions

Functions can be periodized by a suitable change of variables: $\mathbf{x} = \phi(\mathbf{u})$

$$I[f] \equiv \int_{[0,1]^d} \mathrm{d}\mathbf{x}\; f(\mathbf{x}) = \int_{[0,1]^d} \mathrm{d}\mathbf{u}\; \omega_d(\mathbf{u}) f(\phi(\mathbf{u}))$$

$$\phi(\mathbf{u}) = (\phi(u_1), \ldots, \phi(u_d)), \qquad \omega_d(\mathbf{u}) = \prod_{j=1}^{d} \omega(u_j) \qquad \text{and} \qquad \omega(u) = \phi'(u)$$

Korobov transform: $\omega(u) = 6u(1-u), \quad \phi(u) = 3u^2 - 2u^3$

Sidi transform: $\omega(u) = \pi/2 \sin(\pi u), \quad \phi(u) = 1/2(1 - \cos \pi t)$

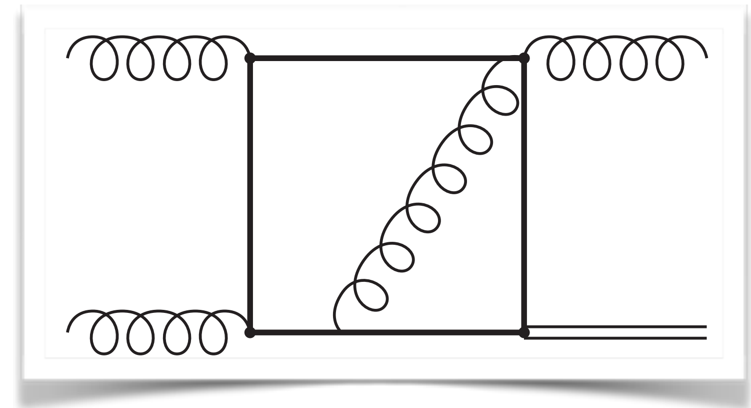
Baker transform: $\phi(u) = 1 - |2u - 1|$

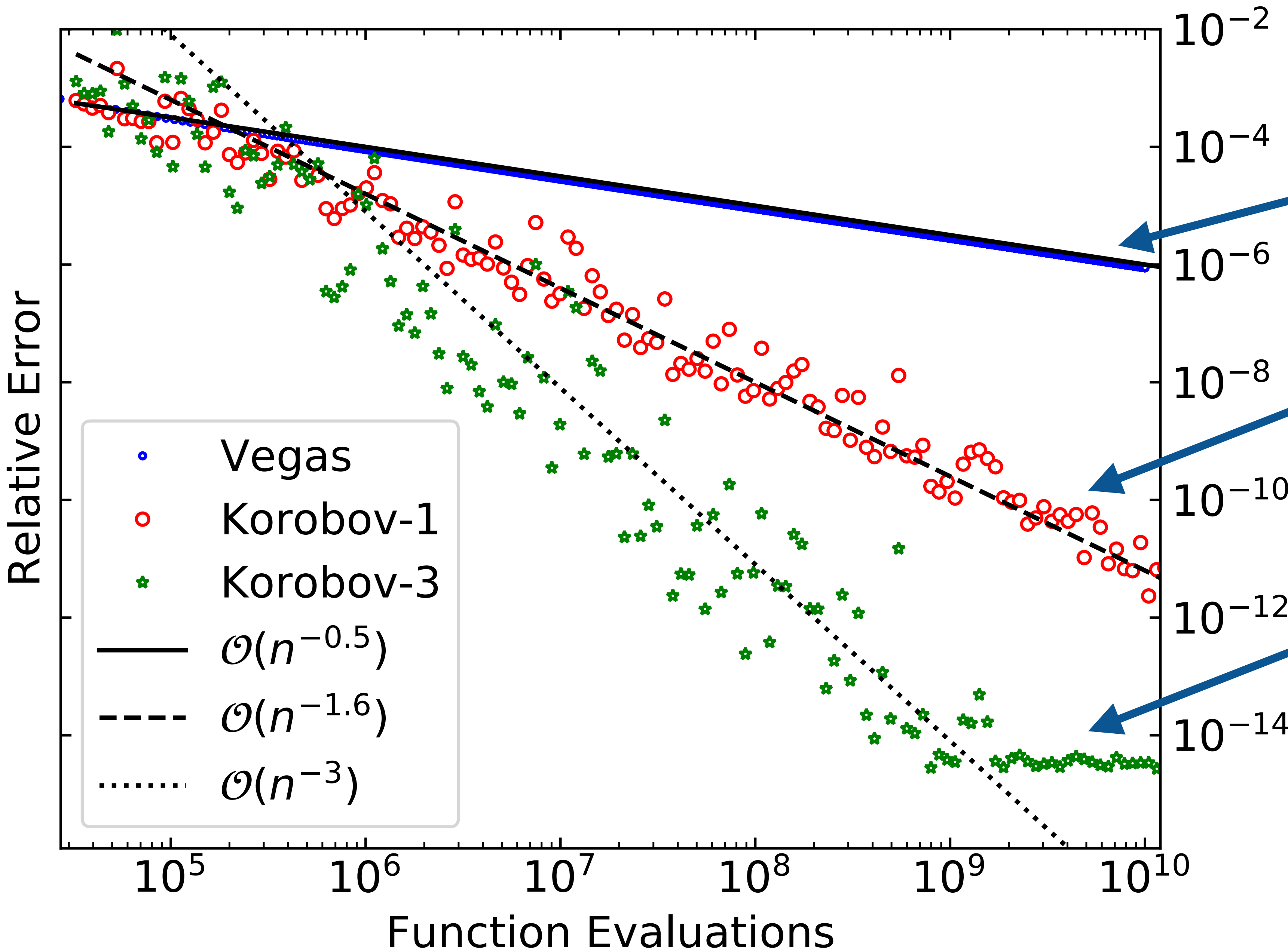
Scaling

Example:

Sector Decomposed HJ Integral

(Note: example stolen from previous calculation)





Monte Carlo scaling

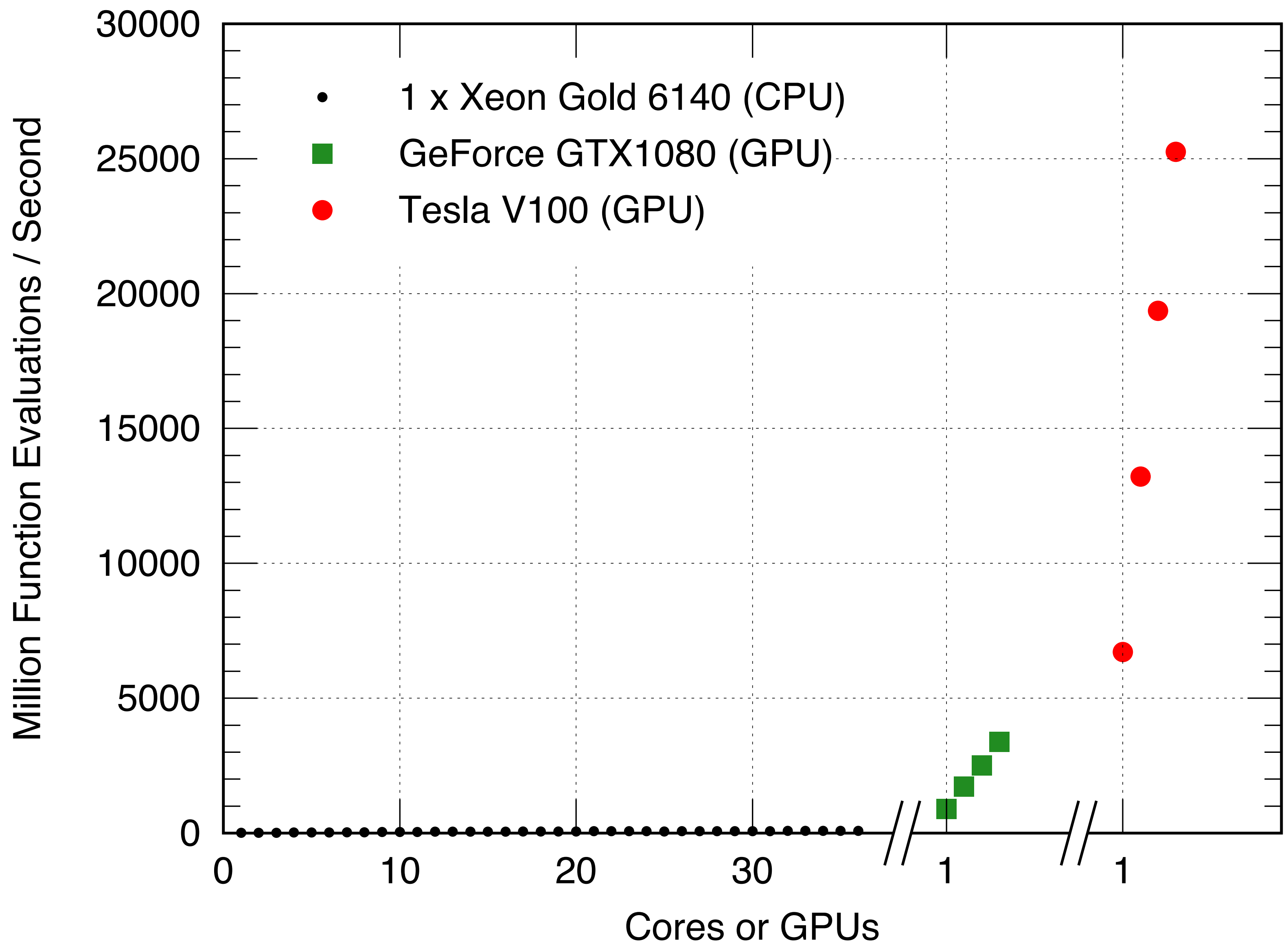
Better than ``guaranteed” n^{-1} scaling

Limited by machine precision (double)

qmc: Performance

Accuracy limited by number of function evaluations

Can accelerate this using Graphics Processing Units (GPUs)



Device	M Func. Evals/s	Speedup
Xeon 6140	80.6	-
GTX1080	897	11x
Tesla V100	6710	83x

Note: Performance gain highly dependent on integrand & hardware!

Still room for further optimisations (both for CPU and GPU)