Machine Learning for Matrix Elements

Fady Bishara

FH Pizza Seminar
25 September 2023

HELMHOLTZ

DESY.

- Need Monte Carlo events @ higher orders in α

- Higher-order matrix elements are slow to evaluate numerically

- Moreover, need to evaluate these matrix elements **many** times

For instance, time to generate 1 million events s.t. MC statistical error $1/\sqrt{N} \sim 10^{-3}$

time/point $[s]$	unwgt. efficiency	CPU time
1	100%	12 days
10	100%	116 days
10	1%	32 years
1000	10%	**317 years**
⋮		

This is not just about speeding up – **it's about making the impossible possible**

MATRIX
CPU budget (total runtime)

[Grazzini, Kallweit, MW '17]

process (${process_id})	LO runtime estimate for 10^{-3} uncertainty	NLO runtime estimate for 10^{-3} uncertainty	NNLO runtime estimate for 10^{-3} uncertainty
$pp \to H$ (pph21)	2 CPU seconds	1 CPU minute	19 CPU days
$pp \to Z$ (ppz01)	4 CPU seconds	1 CPU minute	11 CPU days
$pp \to W^-$ (ppw01)	2 CPU seconds	1 CPU minute	10 CPU days
$pp \to W^+$ (ppwx01)	5 CPU seconds	2 CPU minutes	11 CPU days
$pp \to e^-e^+$ (ppeex02)	28 CPU seconds	12 CPU minutes	22 CPU days
$pp \to \nu_e\bar{\nu}_e$ (ppnenex02)	1 CPU minute	4 CPU minutes	18 CPU days
$pp \to e^-\bar{\nu}_e$ (ppenex02)	1 CPU minute	16 CPU minutes	21 CPU days
$pp \to e^+\nu_e$ (ppexne02)	1 CPU minute	15 CPU minutes	24 CPU days
$pp \to \gamma\gamma$ (ppaa02)	1 CPU minute	19 CPU minutes	6 CPU days
$pp \to e^-e^+\gamma$ (ppeexa03)	9 CPU minutes	4 CPU hours	167 CPU days
$pp \to \nu_e\bar{\nu}_e\gamma$ (ppnenexa03)	1 CPU minute	1 CPU hour	17 CPU days
$pp \to e^-\bar{\nu}_e\gamma$ (ppenexa03)	13 CPU minutes	9 CPU hours	232 CPU days
$pp \to e^+\nu_e\gamma$ (ppexnea03)	17 CPU minutes	1 CPU day	443 CPU days
$pp \to ZZ$ (ppzz02)	1 CPU minute	4 CPU minutes	25 CPU days
$pp \to W^+W^-$ (ppwxw02)	1 CPU minute	3 CPU minutes	13 CPU days
$pp \to e^-\mu^-e^+\mu^+$ (ppemexmx04)	2 CPU minutes	20 CPU minutes	45 CPU days
$pp \to e^-e^-e^+e^+$ (ppeeexex04)	6 CPU minutes	1 CPU hour	193 CPU days
$pp \to e^-e^+\nu_\mu\bar{\nu}_\mu$ (ppeexnmnmx04)	3 CPU minutes	29 CPU minutes	31 CPU days
$pp \to e^-\mu^+\nu_\mu\bar{\nu}_e$ (ppemxnmnex04)	7 CPU minutes	3 CPU hours	119 CPU days
$pp \to e^-e^+\nu_e\bar{\nu}_e$ (ppeexnenex04)	10 CPU minutes	4 CPU hours	52 CPU days
$pp \to e^-\mu^-e^+\bar{\nu}_\mu$ (ppemexnmx04)	3 CPU minutes	26 CPU minutes	19 CPU days
$pp \to e^-e^-e^+\bar{\nu}_e$ (ppeeexnex04)	6 CPU minutes	1 CPU hour	39 CPU days
$pp \to e^-e^+\mu^+\nu_\mu$ (ppeexmxnm04)	4 CPU minutes	1 CPU hour	21 CPU days
$pp \to e^-e^+e^+\nu_e$ (ppeexexne04)	6 CPU minutes	3 CPU hours	44 CPU days

Higgs
DY

diphoton

Wγ

from seconds at LO
to minutes at NLO
to days at NNLO

(MATRIX not optimized for simple processes)

diphoton fastest NNLO process

Wγ slowest NNLO process

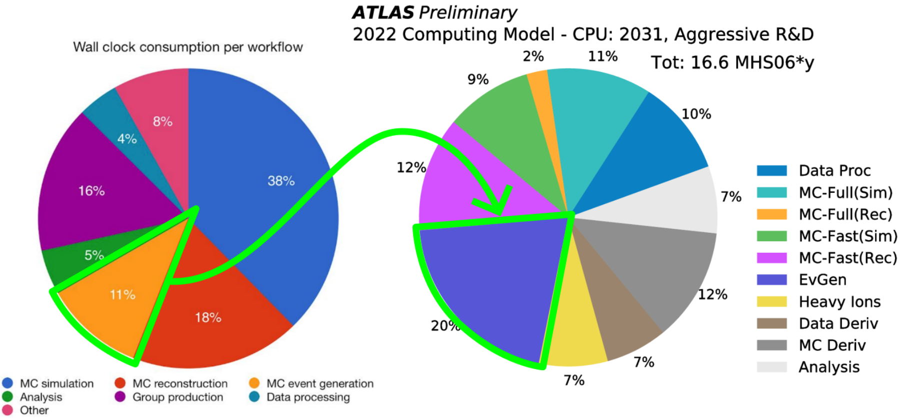
(dependents on fiducial cuts!)

off-shell diboson processes

from minutes at LO
to hours at NLO
to days at NNLO

Slide by Marius Wiesemann

ATLAS Computing Budget, e.g.



But remember the goal: impossible → possible

Machine Learning

- Universal Approximation Theorem: "…any multivariate continuous function can be represented as a superposition of one–dimensional functions" (Neural Networks/sigmoid) [From Braun, J. & Griebel, M. Constr Approx (2009)]

- In practice, convergence is non-trivial (and not guaranteed)

- ✔ Gradient boosting machines perform extremely well

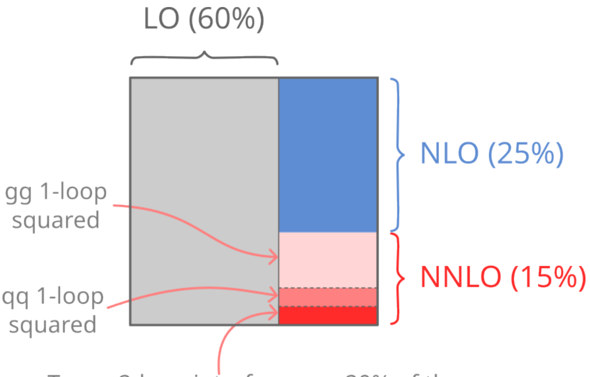
- ✔ Deep neural networks with special architectures do even better for higher dimensions

Why is it reasonable to approximate?

- There are many sources of error:
  - Experimental: statistics, JES/JER, tagging, …,
  - Theoretical: PDF, α_s, …,
  - Monte Carlo statistics $\sim \mathcal{O}(10^{-3})$,
- this guides the approximation precision requirement
- and …

process (${process_id})	σ_{LO}	σ_{NLO}	σ_{loop} ($\sigma_{\mathrm{loop}}/\Delta\sigma^{\mathrm{ext}}_{\mathrm{NNLO}}$)	$\sigma^{r_{\mathrm{cut}}}_{\mathrm{NNLO}}$	$\sigma^{\mathrm{extrapolated}}_{\mathrm{NNLO}}$	K_{NLO}	K_{NNLO}
$pp \to ZZ$ (ppzz02)	$9.845(1)^{+5.2\%}_{-6.3\%}$ pb	$14.10(0)^{+2.9\%}_{-2.4\%}$ pb	$1.361(1)^{+25\%}_{-19\%}$ pb (52.9%)	$16.68(1)^{+3.2\%}_{-2.6\%}$ pb	$16.67(1)^{+3.2\%}_{-2.6\%}$ pb	+43.3%	+18.2%
$pp \to W^+W^-$ (ppwxw02)	$66.64(1)^{+5.7\%}_{-6.7\%}$ pb	$103.2(0)^{+3.9\%}_{-3.1\%}$ pb	$4.091(3)^{+27\%}_{-19\%}$ pb (29.5%)	$117.1(1)^{+2.5\%}_{-2.2\%}$ pb	$117.1(1)^{+2.5\%}_{-2.2\%}$ pb	+54.9%	+13.4%

Grazzini, Kallweit, Wiesemann [1711.06631]



Tree x 2-loop interference ~20% of the NNLO contribution...

... but O(**100-1000**) times slower than 1-loop amplitudes

- *High-Precision Regressors for Particle Physics*
  F. Bishara, A. Paul, J. Dy. Paper submitted to Nature Scientific Reports for peer-review (now in 2nd round) [arXiv:2302.00753]

- *Skip Connections for High Precision Regressors*
  F .Bishara, A. Paul, J. Dy. Machine Learning and the Physical Sciences, Workshop at the 36th Conference on Neural Information Processing Systems (NeurIPS 2022)

- *Machine Learning Amplitudes for Faster Event Generation*
  F. Bishara and M. Montull. Phys. Rev. D 107 (2023) no.7, L071901 [arXiv:1912.11055]

Physics guidance and considerations

- Functions span many orders of magnitude, transform

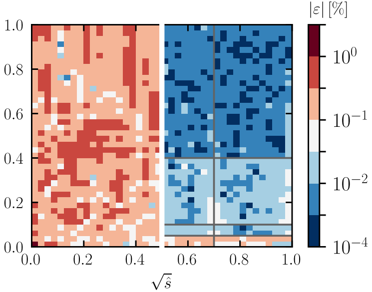
$$f(x) = \begin{cases} \log(1+x) & x > 0 \\ -\log(1-x) & x < 0 \end{cases}$$

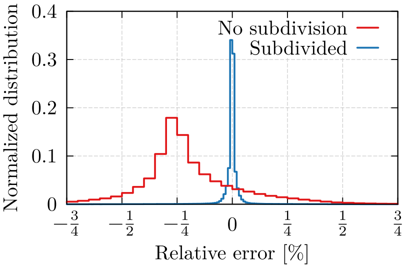
- Symmetries allow to reduce a set of 18 functions to just **4**

- Improve NN performance by constructing linear combinations of functions with nicer properties (e.g. symmetry)

- How to generate a training sample over the domain?
  - Generally, sample uniformly
  - Some variables need to be sampled log-uniformly – **need to invert transformation s.t. point density is uniform!**

A first example: 2d function with boosted decision tree



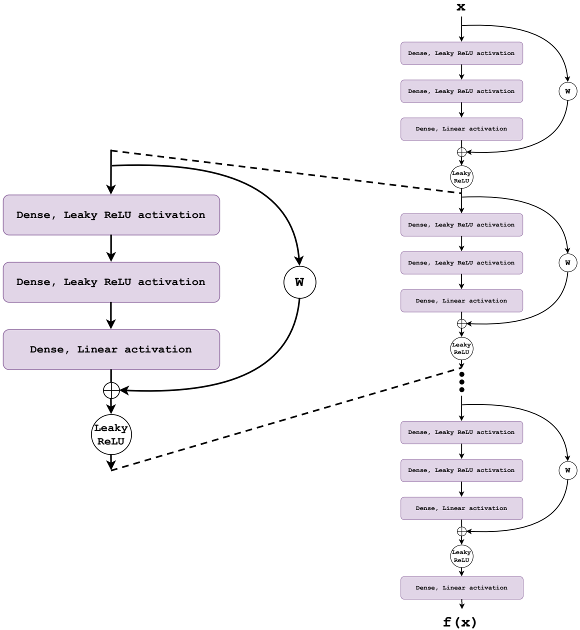


Exact: $\sim$ 16s / point

Approximate: $\sim$ 16s / **1M** points

Approximation is 10^6 times faster with relative error $< 10^{-3}$!

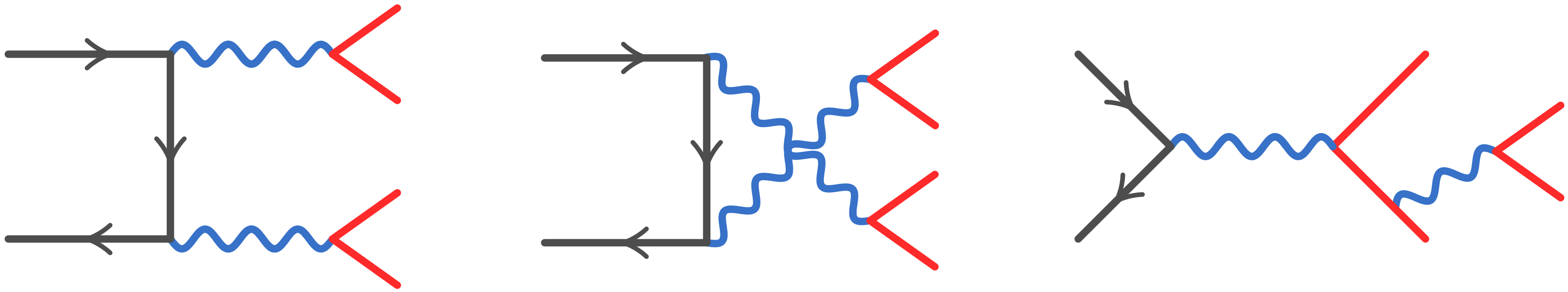
Neural network with skip connections



The Amplitudes

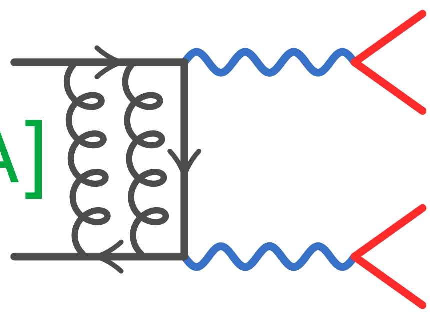
pp →4l @ NNLO (double virtual)

TL

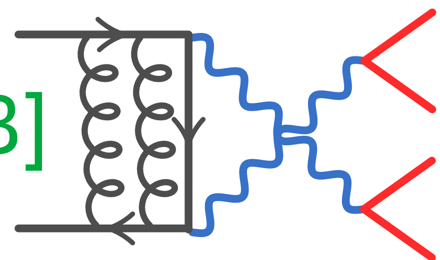


2L

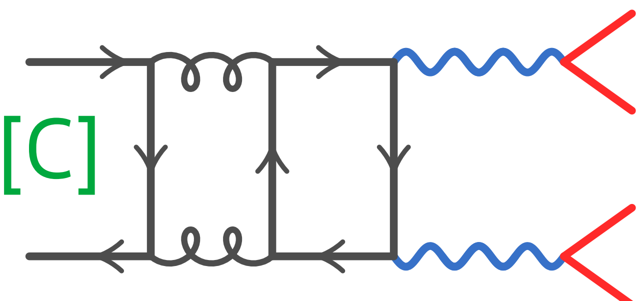
[A]



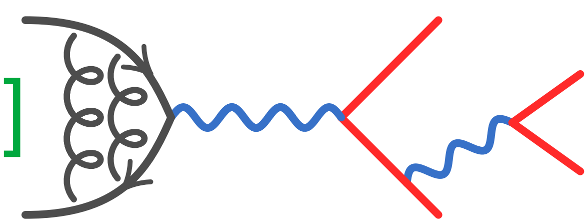
[B]



[C]

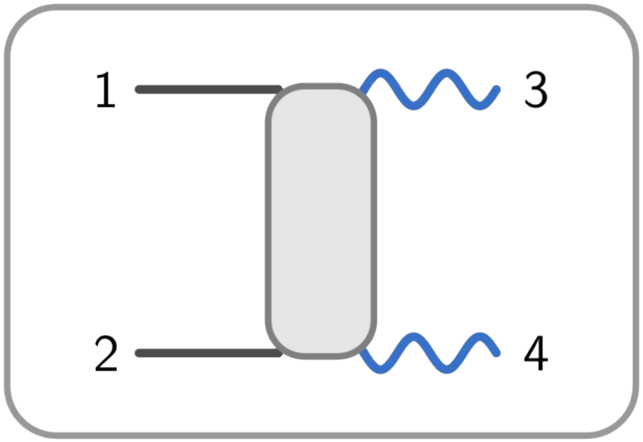


[F]



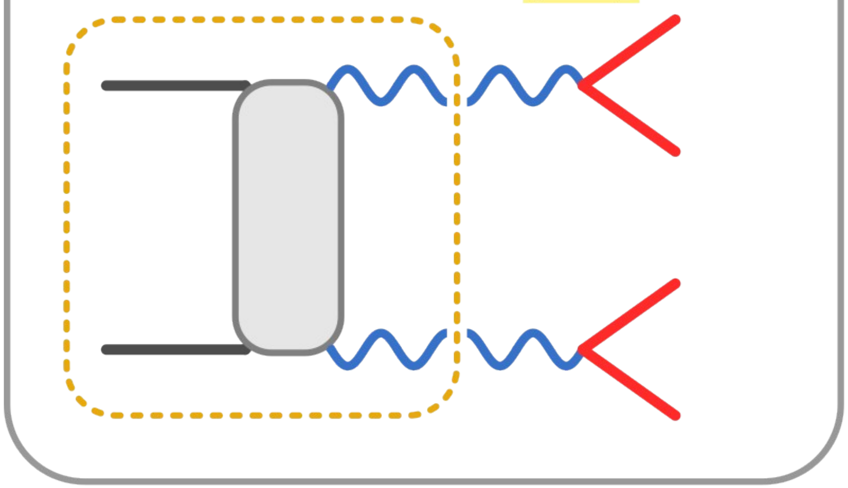
The phase-space

on-shell $2 \to 2$ [2d]



on-shell $2 \to 4$ [8d]

off-shell $2 \to 2$ [4d]



- The resonant-propagator numerators can be rewritten as

$$\Delta_{\mu\nu} = -g_{\mu\nu} + (1-\xi)\frac{q_\mu q_\nu}{q^2 - m^2} \xrightarrow{\text{e.o.m.}} -g^{\mu\nu} = \sum_\lambda \epsilon_\mu^\lambda \epsilon_\nu^{\lambda *}$$

Two-loop matrix element

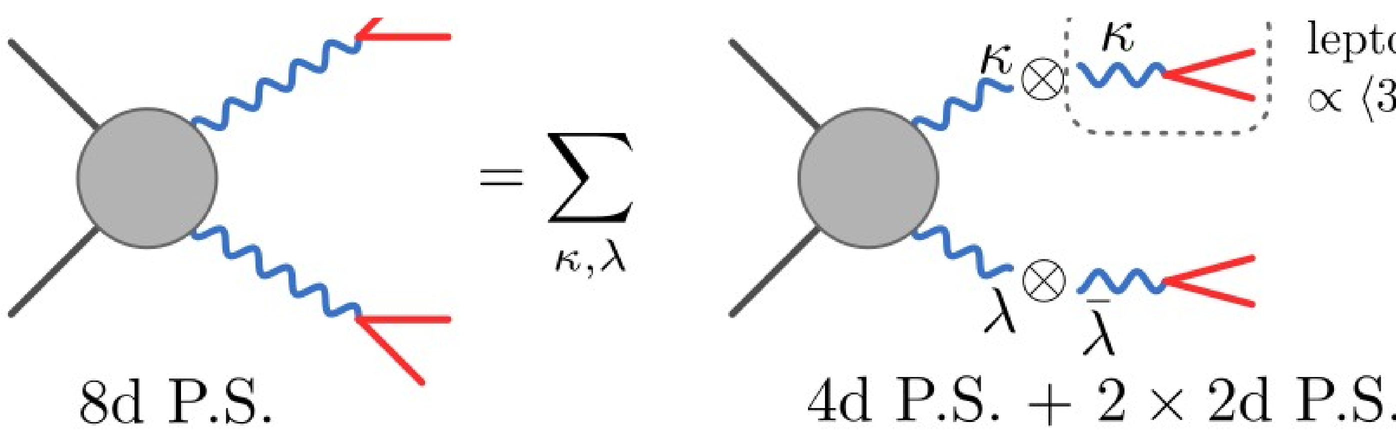
10 tensor structures
@LO only 4: $T_7, \ldots, T_{10}$

form factors = scalar functions of kinematic vars.

Squared amplitude:

$$\mathcal{T}\left(s,t,p_3^2,p_4^2\right) = \sum_{\text{pols,cols}} \left| \sum_{j=1}^{10} A_j\left(s,t\right) T_j^{\mu\nu} \varepsilon_\lambda^\mu\left(p_3\right) \varepsilon_{\lambda'}^\nu\left(p_4\right) \right|^2$$

$$\mathcal{T}^{(2)}(s,t,p_3^2,p_4^2) = 2\Re\left(\langle\mathcal{M}^{(0)}|\mathcal{M}^{(2)}\rangle\right) + \langle\mathcal{M}^{(1)}|\mathcal{M}^{(1)}\rangle$$



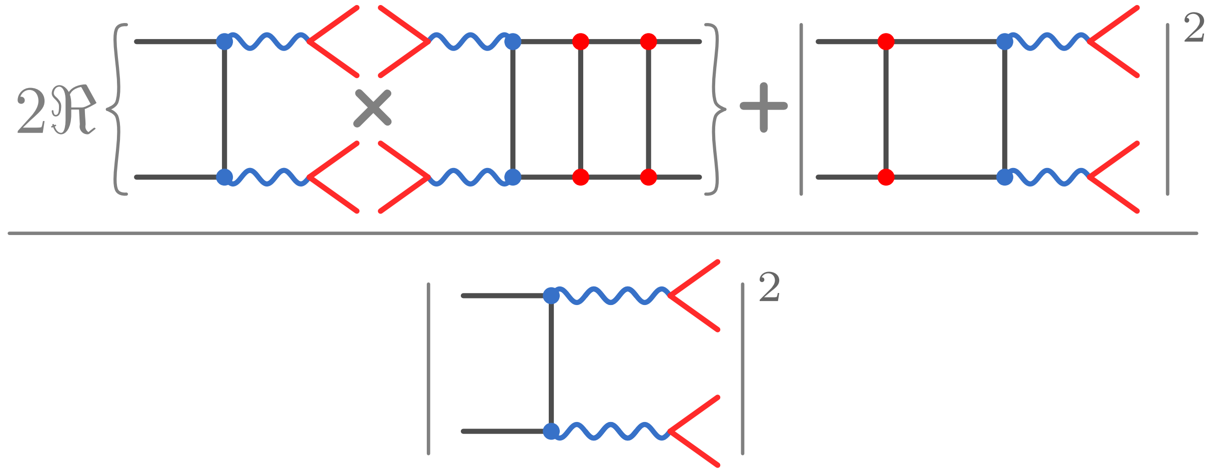
lepton amps. are simple: $\propto \langle 3\gamma_\mu 4]$ and $[3\gamma_\mu 4\rangle$

Form factors computed using VVAMP
[Gehrmann, von Manteuffel, Tancredi [1503.04812]]

Two paths to approximate

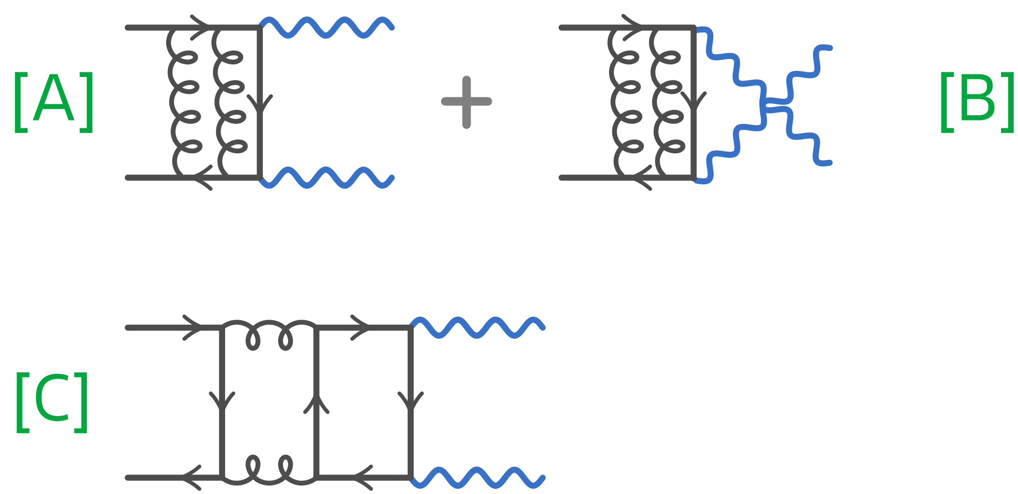
k-factor

- Couplings cannot be factored out (frozen-in)
- Phase-space is 8-dim.
- Can sum over helicities → only one function per process



(sub)-amplitudes

- Couplings can be factored out (at least when sum of Qi=0)
- Phase-space is 4-dim.
- Can be recycled for different vector boson combinations



[Work in progress with Ayan Paul]

- Goal: **implement into MC generators**, many details to consider
  - want functions that can be recycled $\rightarrow$ couplings factored out
  - and for this, approximate **amplitudes** (i.e. not squared)
  - amplitudes are complex objects $f : \mathbb{R}^d \mapsto \mathbb{C}$
  - want V_1 and V_2 **off-shell** but don't want leptons so **4d**

- Therefore, have $2 \times 3 \times 3 = 18$ amplitudes in principle
  - ✔ Amplitudes have symmetries $\rightarrow$ reduced set

- Nice choice of reference momenta $\rightarrow$ more symmetry
  - simultaneous light-cone decomposition of p_3 and p_4 leads to

$$\epsilon^-_{3,\mu} = \frac{\langle 4\gamma_\mu 3]}{\sqrt{2}\langle 43\rangle}, \quad \epsilon^+_{3,\mu} = \frac{\langle 3\gamma_\mu 4]}{\sqrt{2}[34]}, \quad \epsilon^-_{4,\mu} = \frac{\langle 3\gamma_\mu 4]}{\sqrt{2}\langle 34\rangle}, \quad \epsilon^+_{4,\mu} = \frac{\langle 4\gamma_\mu 3]}{\sqrt{2}[43]}$$

  - in C.M. frame with p_3 and p_4 pointed along $\pm\hat{z}$ direction and with appropriate choice of spinor phases, $\langle 34\rangle = [43]$

- Only **4** / 18 amplitudes can generate the full set

- $\Re$ and $\Im$ parts of the amplitudes are correlated $\rightarrow$ natural to output them together (trivial for NNs)

- In the future, could be a good application for complex activation functions

- For now, ignore complications that arise if the two pairs of leptons have the same flavor

Populating the Phase-Space

$qq \to Z^* Z^*$

[FB & A. Paul [work in progress]]

- Map full phase-space to unit hypercube

$$\sqrt{s_{12}} \in [m_{34} + m_{56}, 14\ \mathrm{TeV}] \mapsto [0,1]$$
$$\cos\theta^* \in [-1,1] \mapsto [0,1]$$
$$m_{34}, m_{56} \in [50, 130]\ \mathrm{GeV} \mapsto [0,1]$$

  - otherwise **no cuts** on P.S.!
  - two options to extend m_{ij} even up to 14 TeV to cover W boson

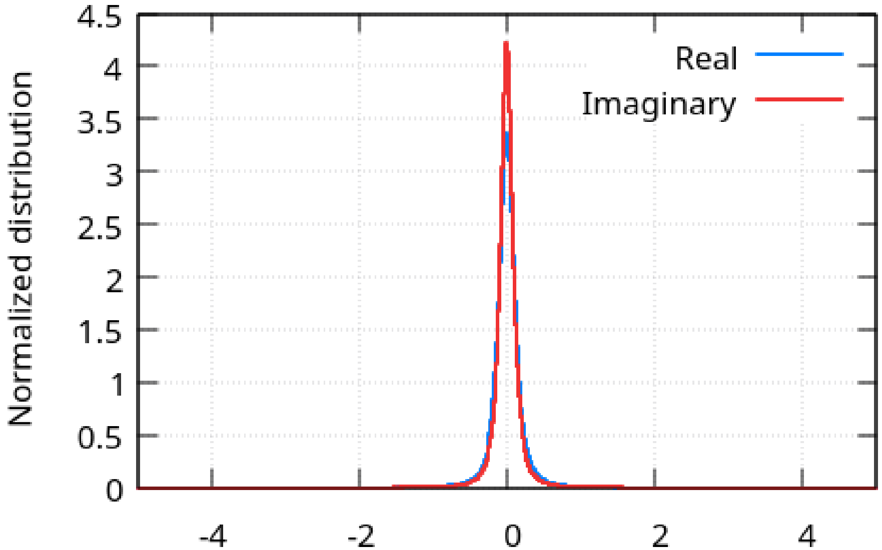
- The scattering angle of $Z(p_{34})$ is defined as

$$\cos\theta^* = \frac{t-u}{s\,\lambda}$$

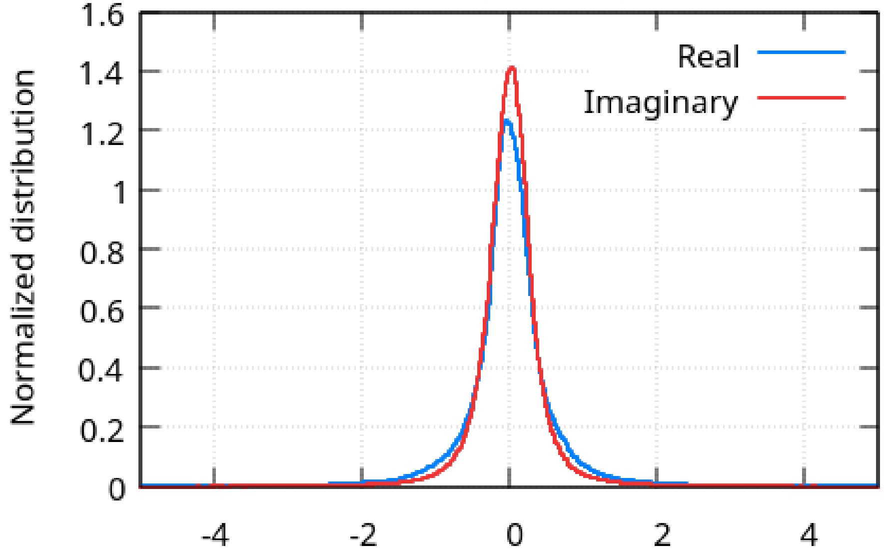
where $s \equiv s_{12}$ and λ is the Källén function $\lambda(1, \frac{m_{34}}{\sqrt{s_{12}}}, \frac{m_{56}}{\sqrt{s_{12}}})$

Earlier results: uniform up to 500 GeV

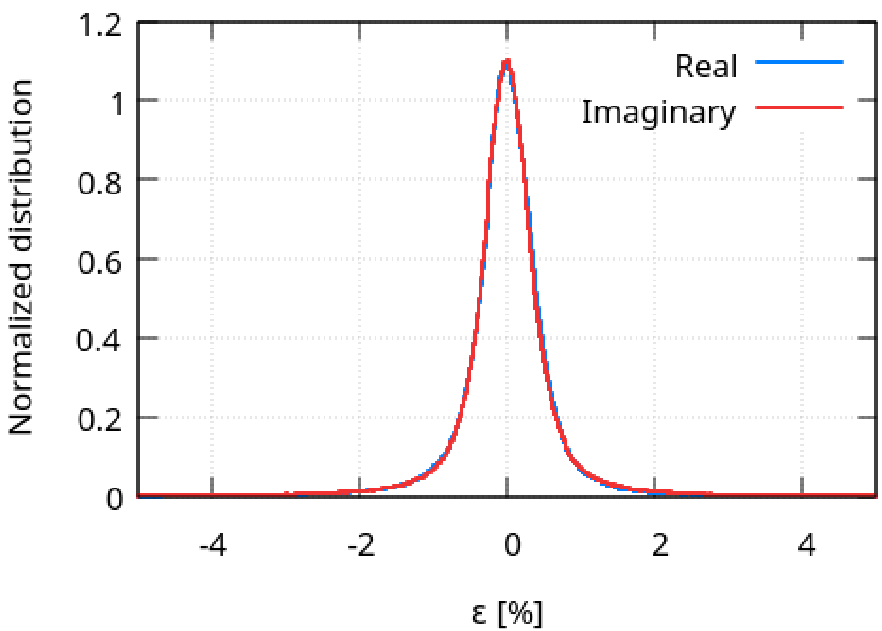
$q_L\ (-,-)$



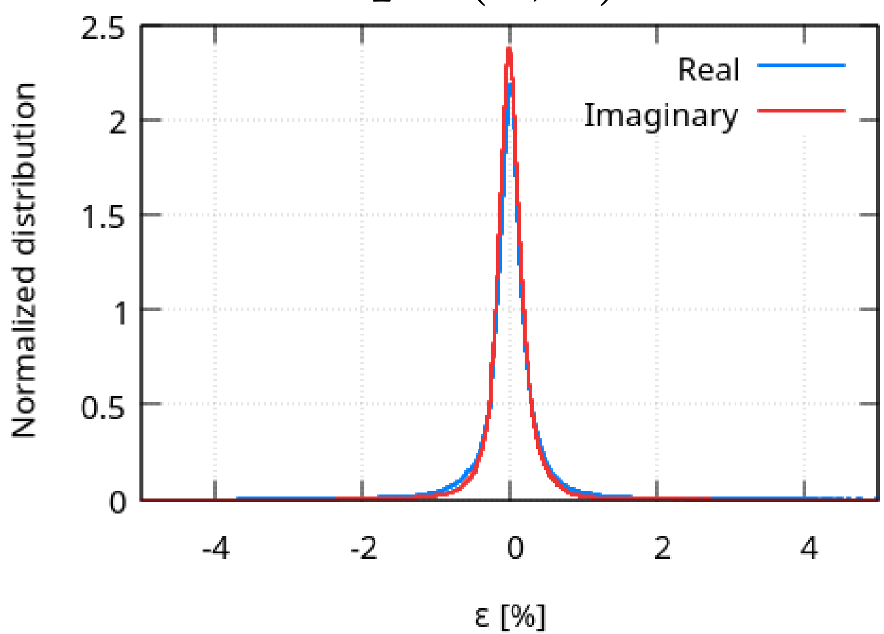
$q_L\ (-,0)$



$q_L\ (-,+)$



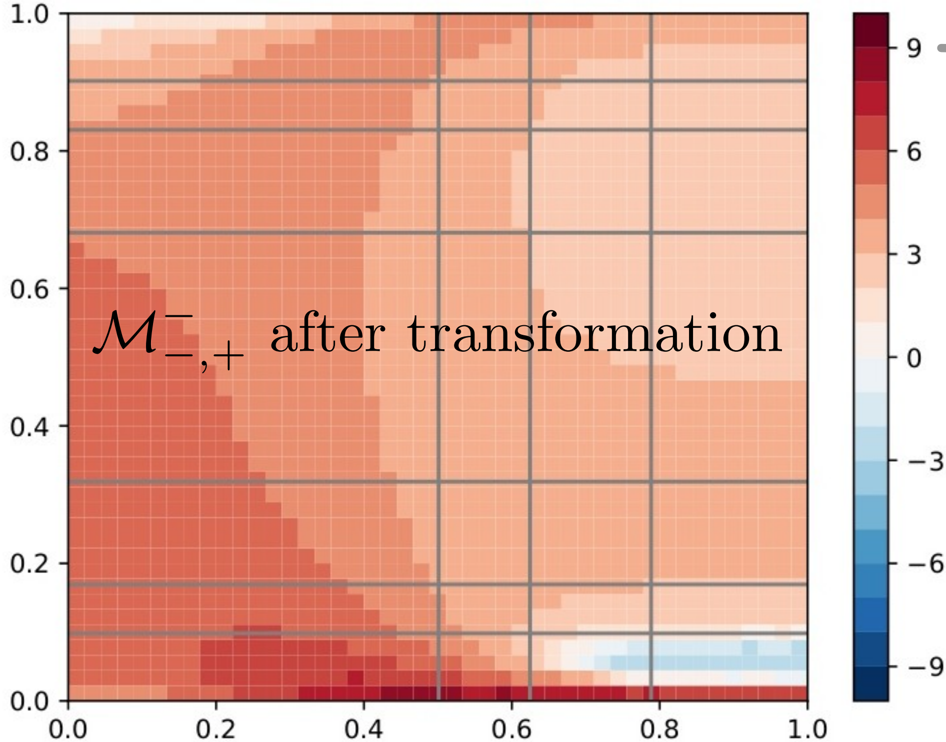
$q_L\ (0,0)$



$qq \to Z^* Z^*$ [FB & A. Paul [work in progress]]

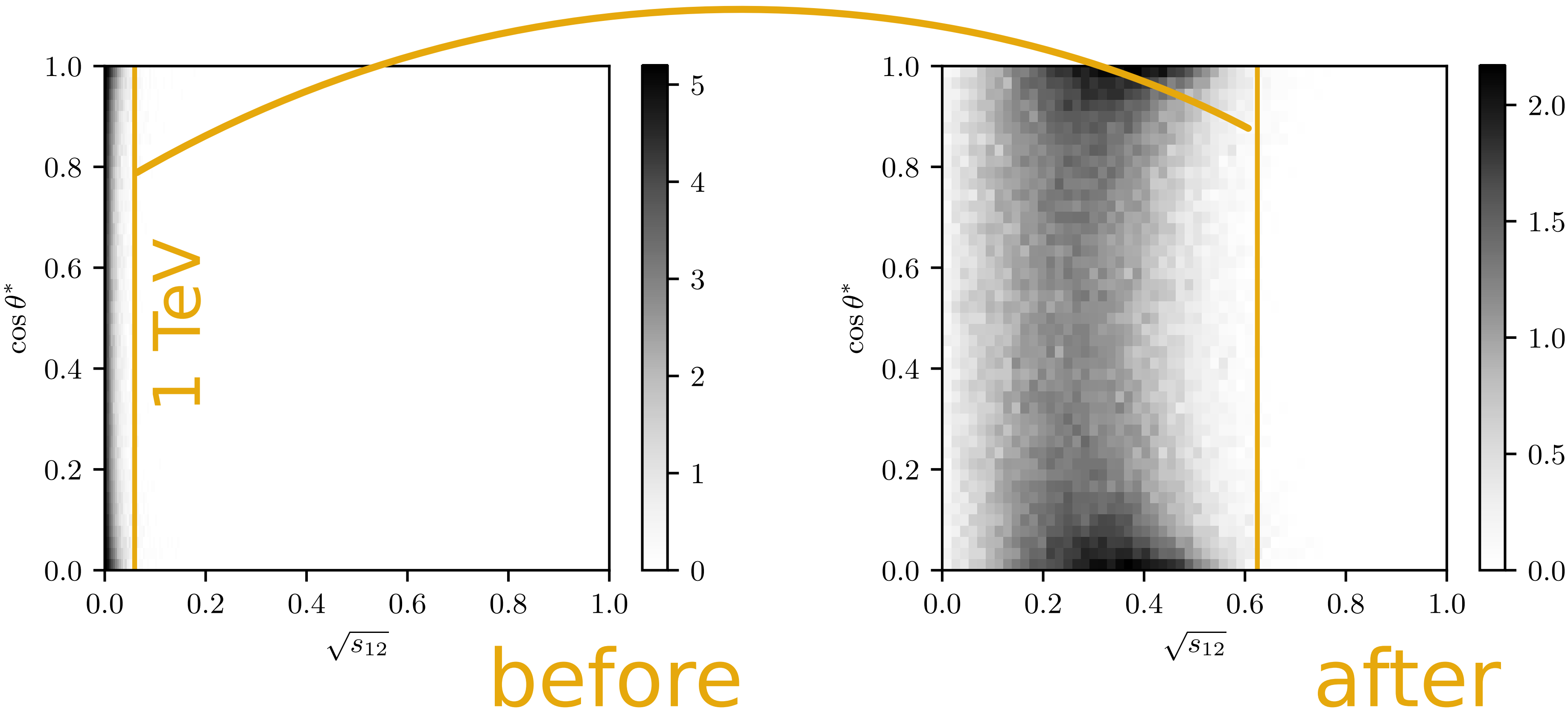
- Amplitudes span many order of magnitude (and can be negative of course) $\to$ transform according to

$$f(x) = \begin{cases} \log(1+x) & x > 0 \\ -\log(1-x) & x < 0 \\ 0 & \text{otherwise} \end{cases}$$



$e^9 \sim \mathcal{O}(10^4)$

- Training the network is done on the full phase-space, uniformly populated except for s_{12} because...

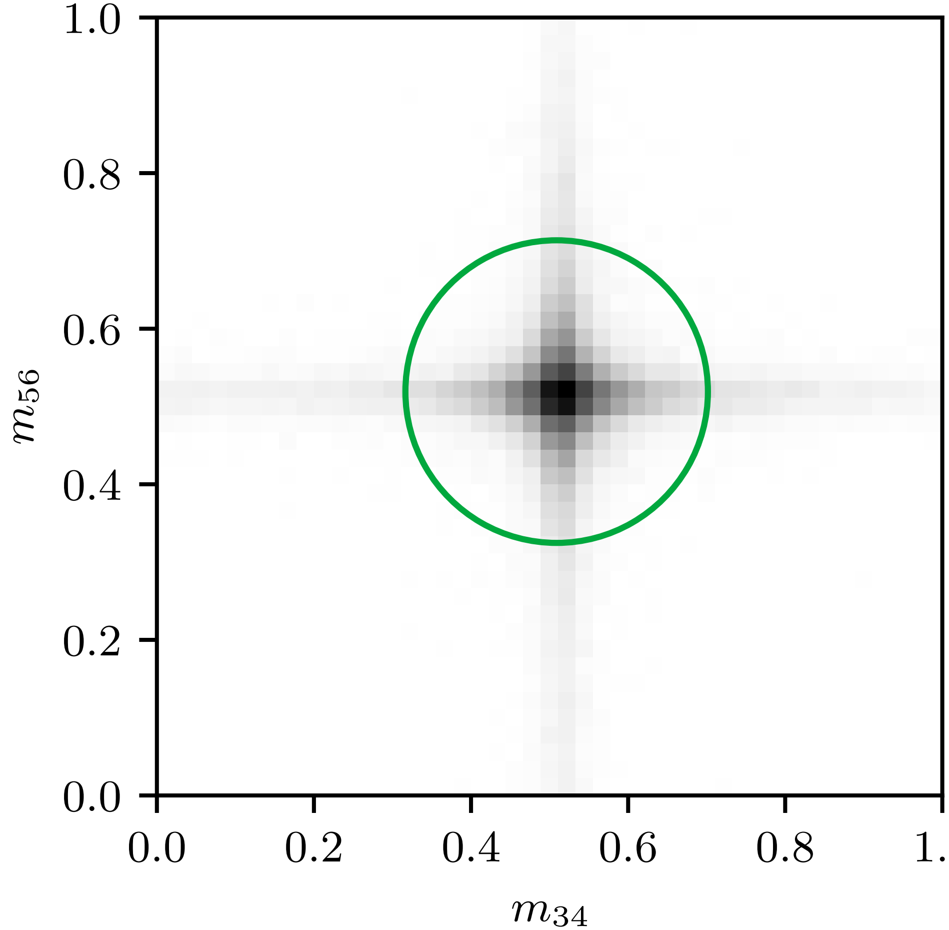


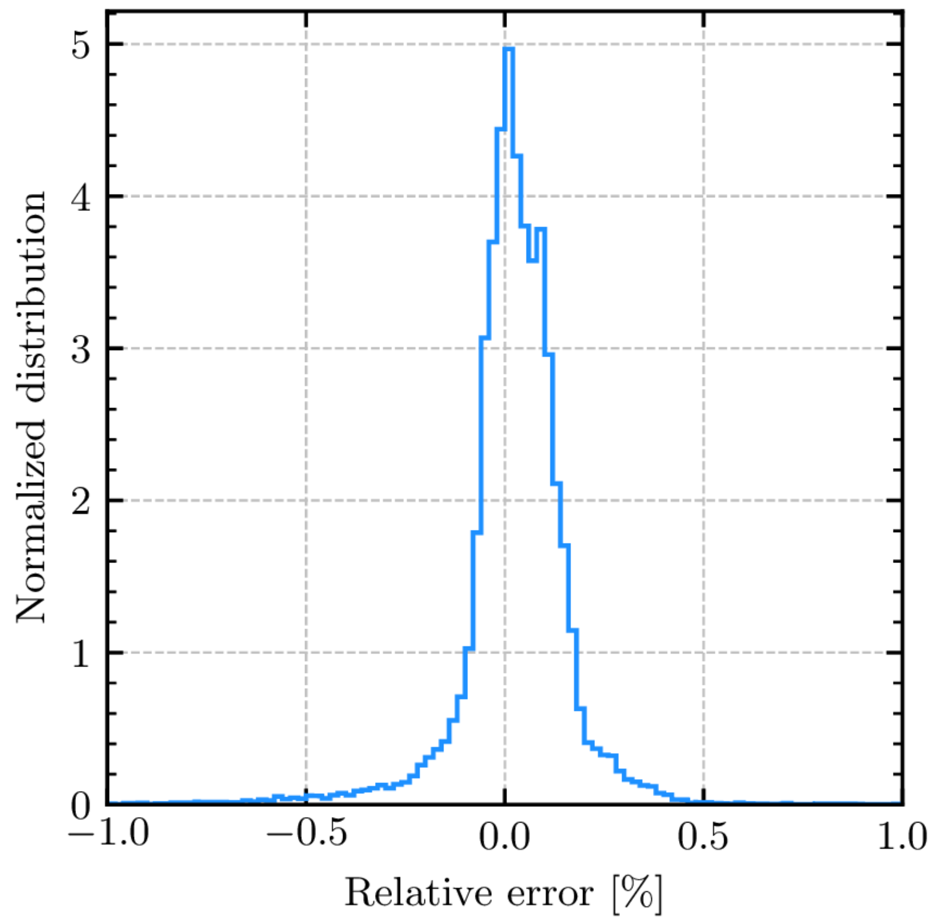
- Populate s_{12} log-uniformly

$$\mathsf{CDF}(x) = \frac{1}{p} \log\left\{1 + x\left(e^{p} - 1\right)\right\}$$

* PS from unweighted LO events generated with MadGraph

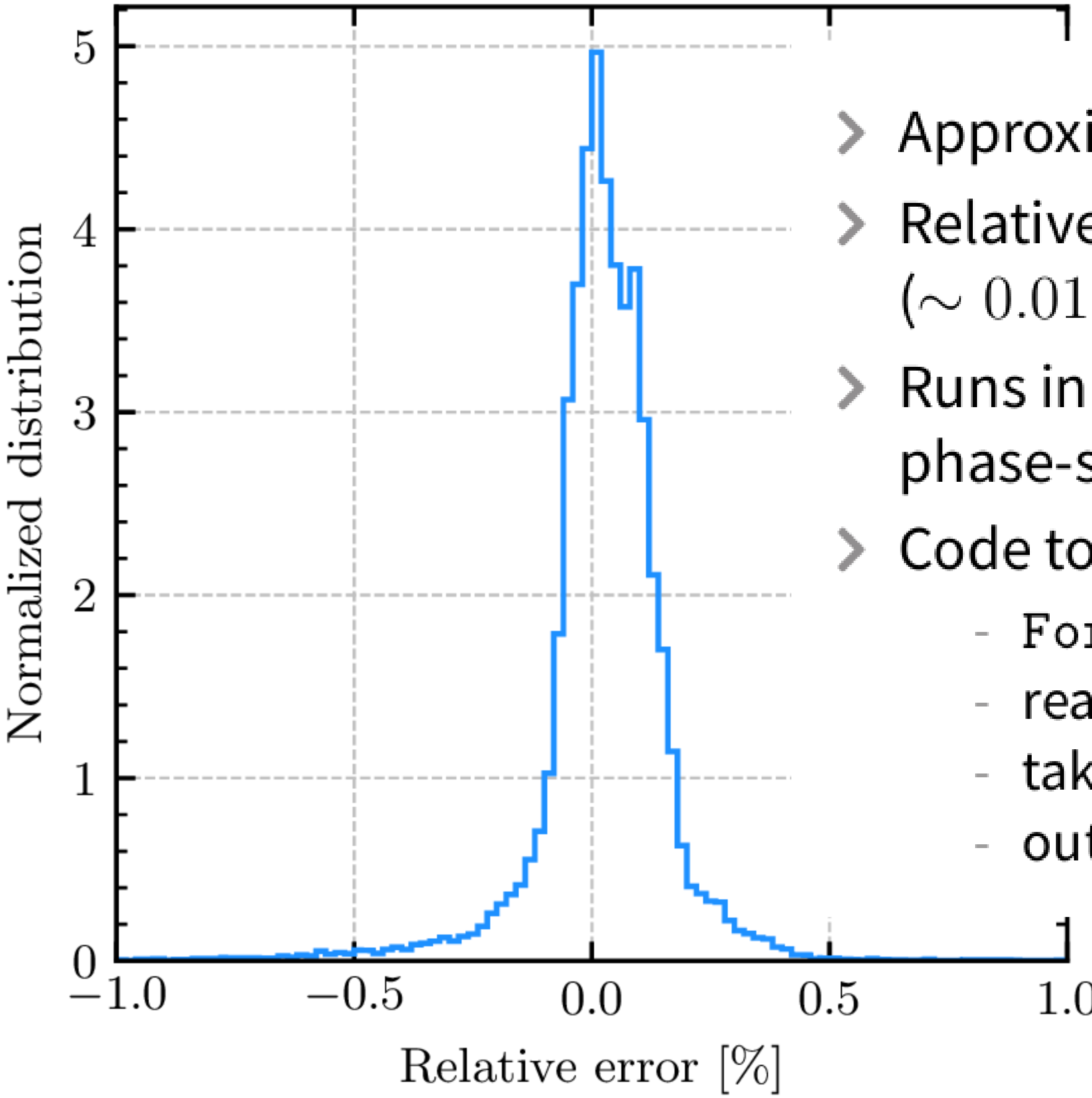
- The $m_{34} - m_{56}$ is also clearly sparse if PS is uniformly populated but, for now, keep it as is





Trained on uniformly populated masses predictions in a very small region of PS!

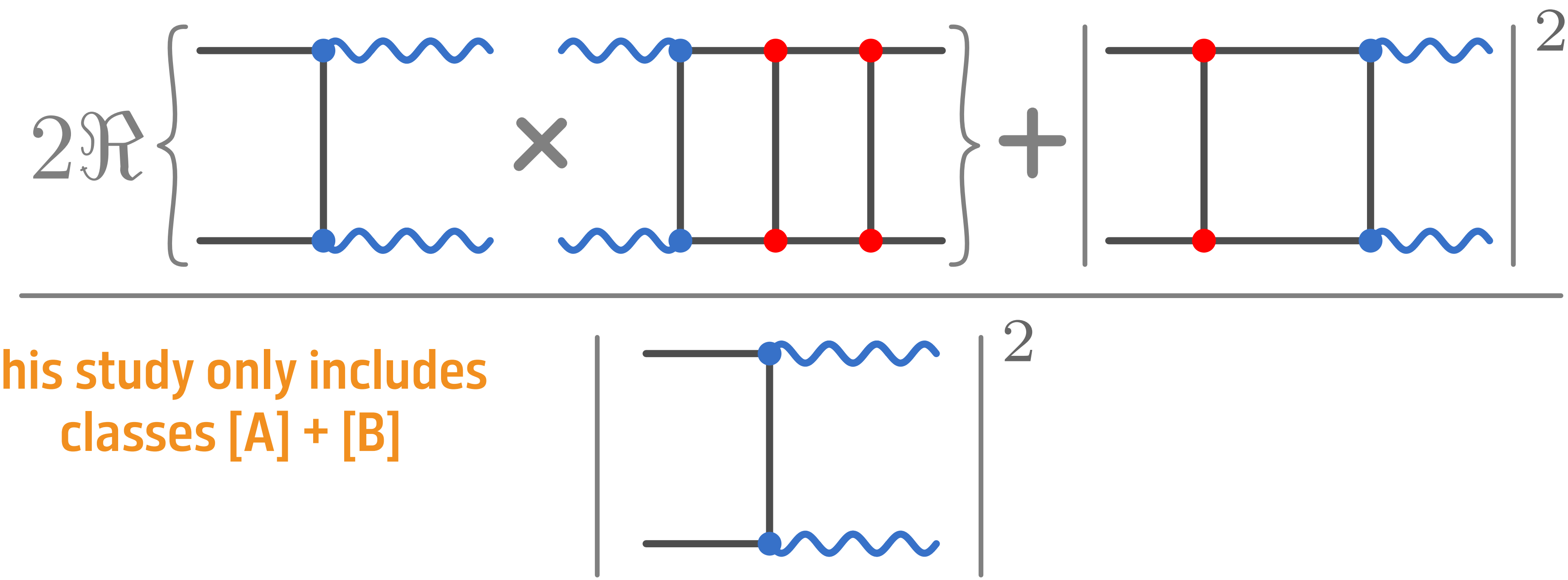
Of course can/should improve this i.e. by distributing according to a wide Cauchy dist.



- Approximation of $2\Re\{\mathcal{M}^{(0)}\mathcal{M}^{(2)*}\}$
- Relative error is **sub-percent** ($\sim 0.01\%$ on total)
- Runs in < 2 **milliseconds** per phase-space point
- Code to produce this:
  - `Fortran` prog. (no ext. dep.)
  - reads parameter files (a few MB)
  - takes in phase-space coords
  - outputs helicity amplitudes

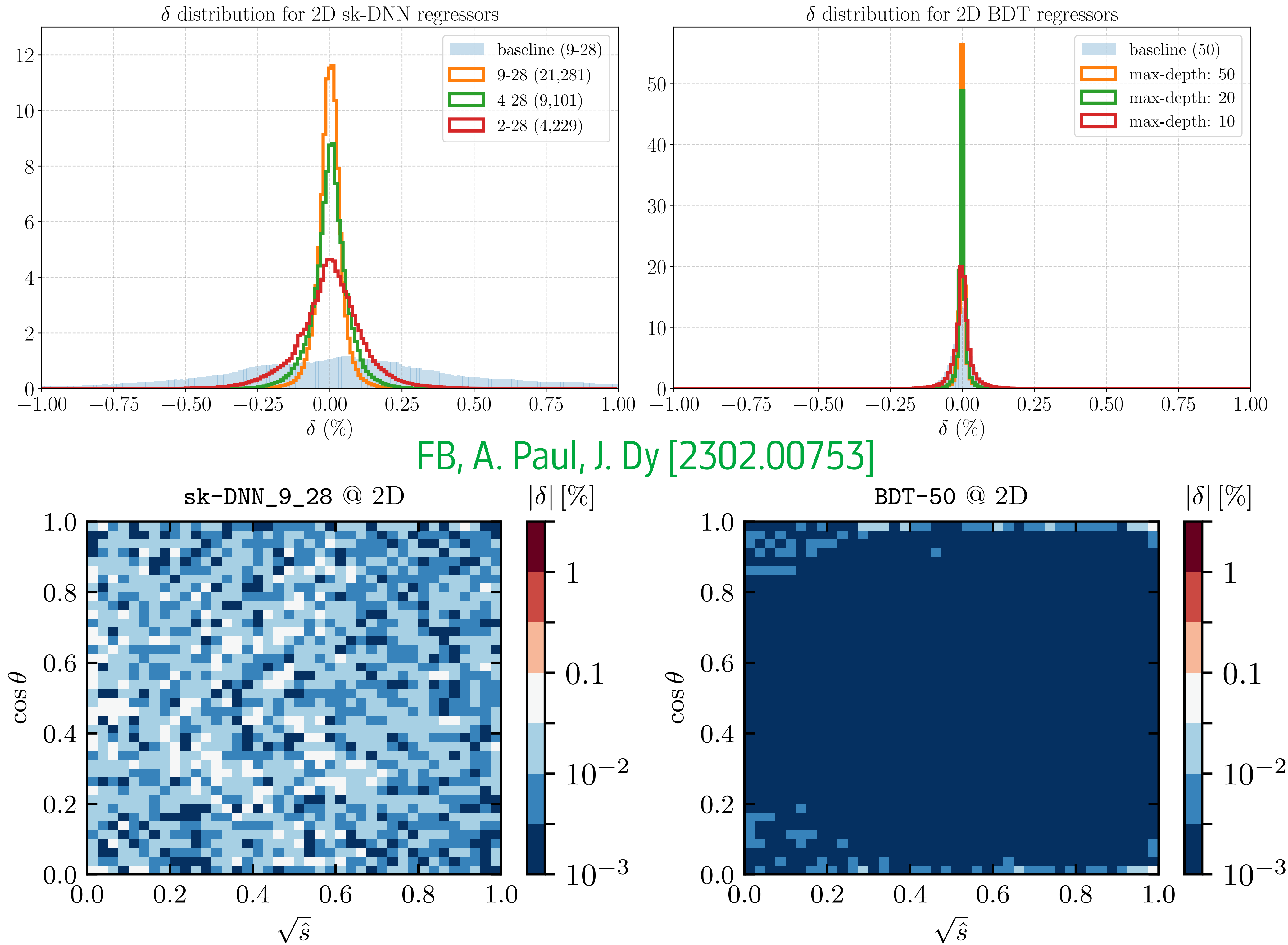
K-Factors

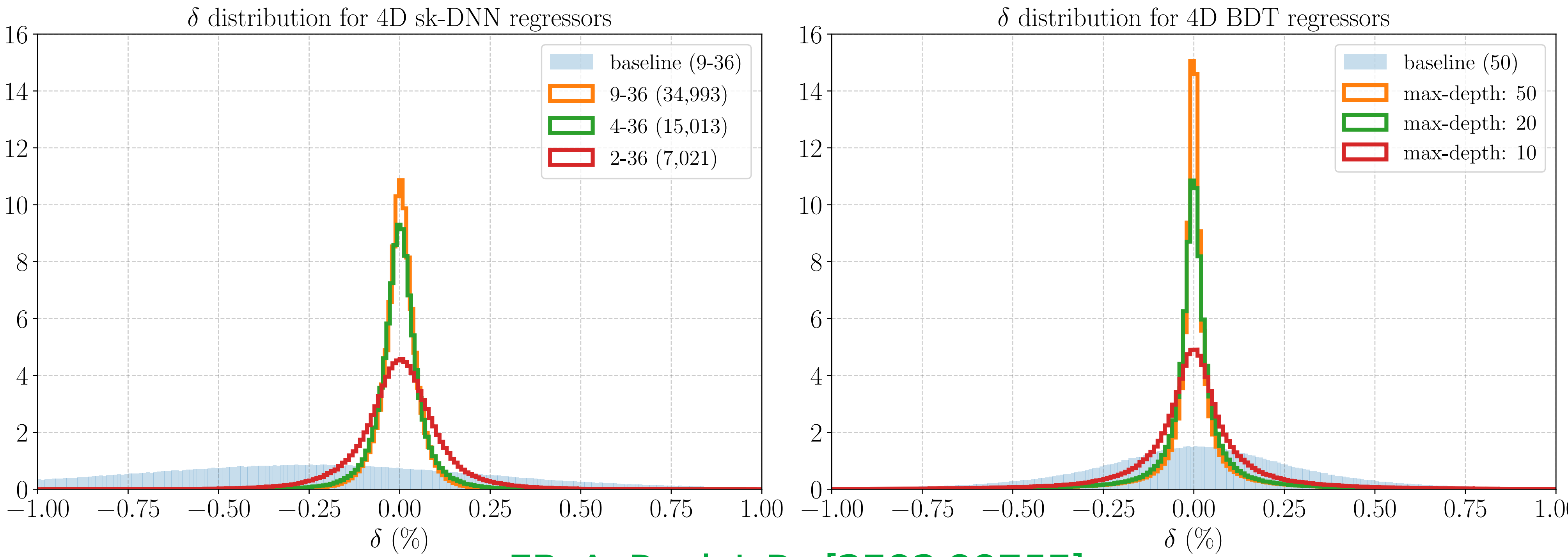
- “Toy” process just to establish generalization to higher dims.
  [FB, Ayan Paul, Jennifer Dy; `https://ml4physicalsciences.github.io/2022/files/NeurIPS_ML4PS_2022_164.pdf`]
  [FB, Ayan Paul, Jennifer Dy; [2301.XXXXX]]



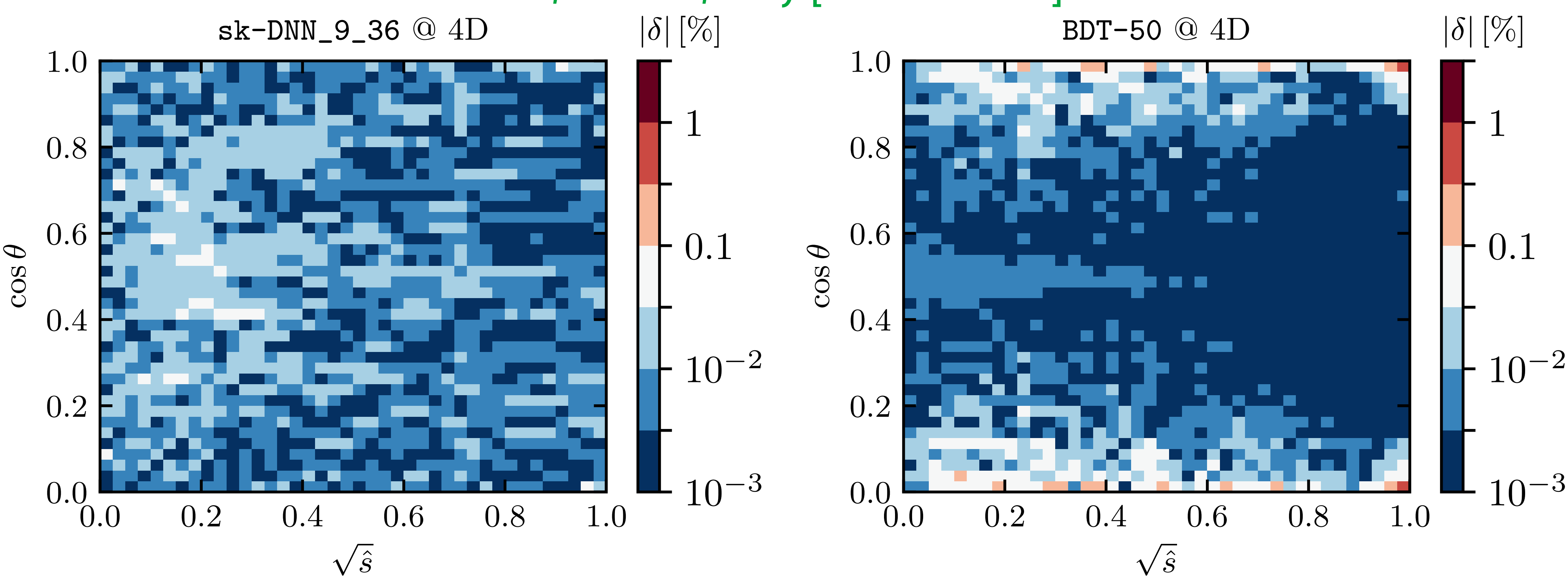
This study only includes classes [A] + [B]

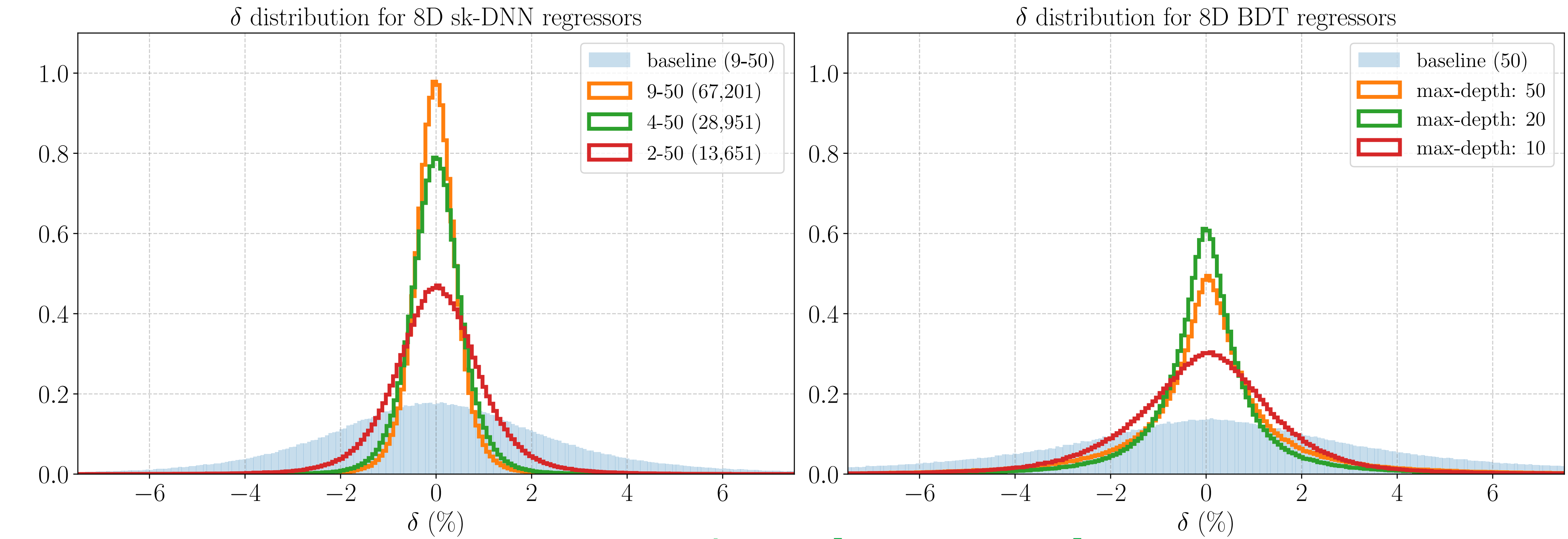
- Compute $\langle |\mathcal{M}|^2 \rangle$ for $qq \to ZZ(\to 4\ell)$ using `VVAMP`
  - Training: 4.8M points
  - Validation: 3.2M
  - Testing: 2M


δ distribution for 2D sk-DNN regressors
baseline (9-28)
9-28 (21,281)
4-28 (9,101)
2-28 (4,229)
δ (%)
δ distribution for 2D BDT regressors
baseline (50)
max-depth: 50
max-depth: 20
max-depth: 10
δ (%)
FB, A. Paul, J. Dy [2302.00753]
sk-DNN_9_28 @ 2D
$|\delta|$ [%]
$\cos\theta$
$\sqrt{\hat{s}}$
BDT-50 @ 2D
$|\delta|$ [%]
$\cos\theta$
$\sqrt{\hat{s}}$

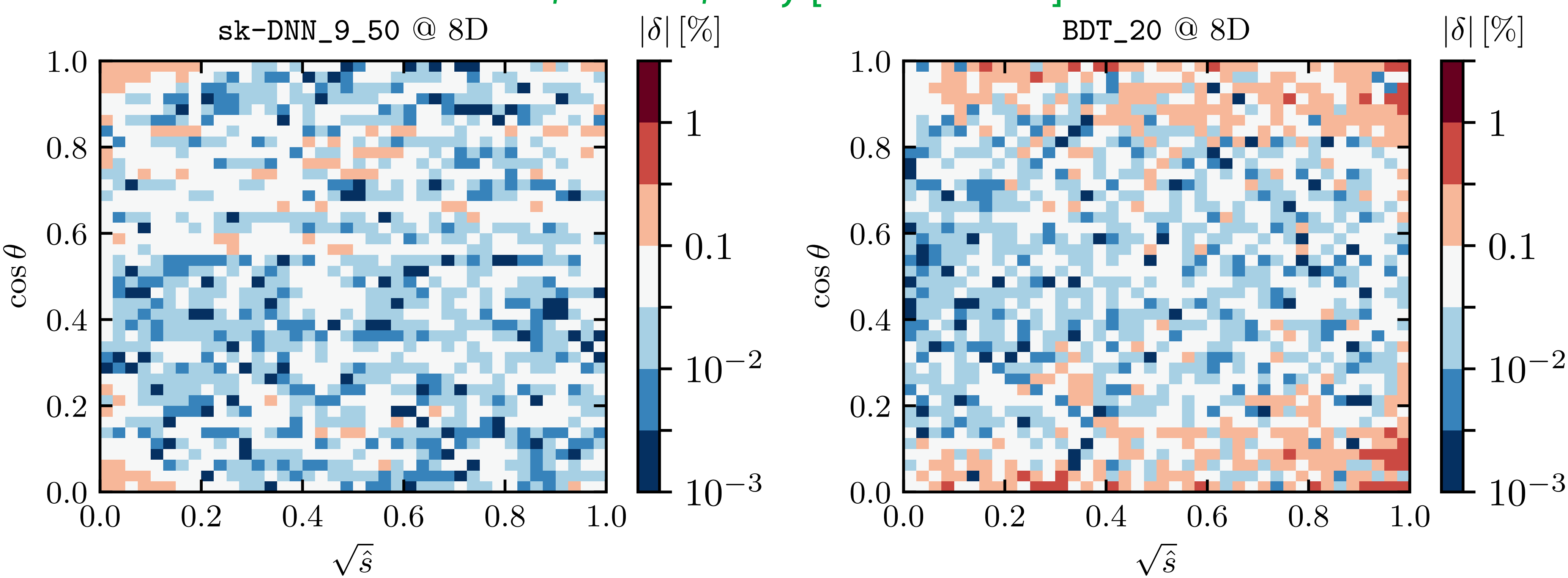


FB, A. Paul, J. Dy [2302.00753]





FB, A. Paul, J. Dy [2302.00753]



Summary and outlook

- Approximate double-virtual amplitudes can leapfrog MC generation times for some processes
- Implementation soon in `MCFM`, then in `GENEVA` and hopefully also `MATRIX`
- Many many future directions and application to other amplitudes, e.g., including gluon-induced di-bosons @NLO, top mass in the loop, 5-point 3-photon two loop amplitude, etc.