(Machine) Learning Amplitudes for Faster Event Generation

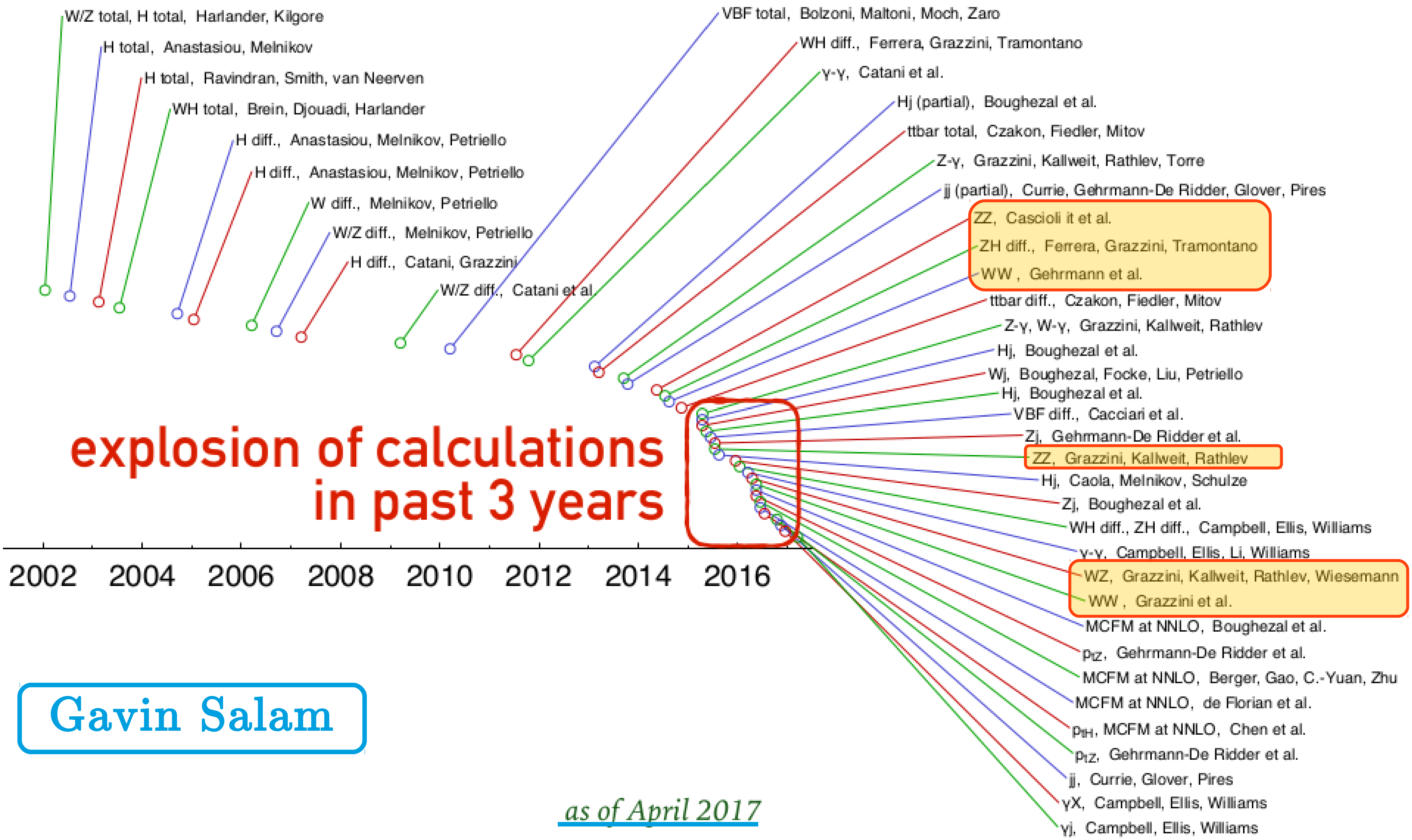
Fady Bishara

Cavendish-DAMTP Seminar
28.05.20

with Marc Montull [1912.11055] + w.i.p

HELMHOLTZ RESEARCH FOR GRAND CHALLENGES
European Research Council
Established by the European Commission
DESY.

Hard processes to NNLO in α_s



Gavin Salam

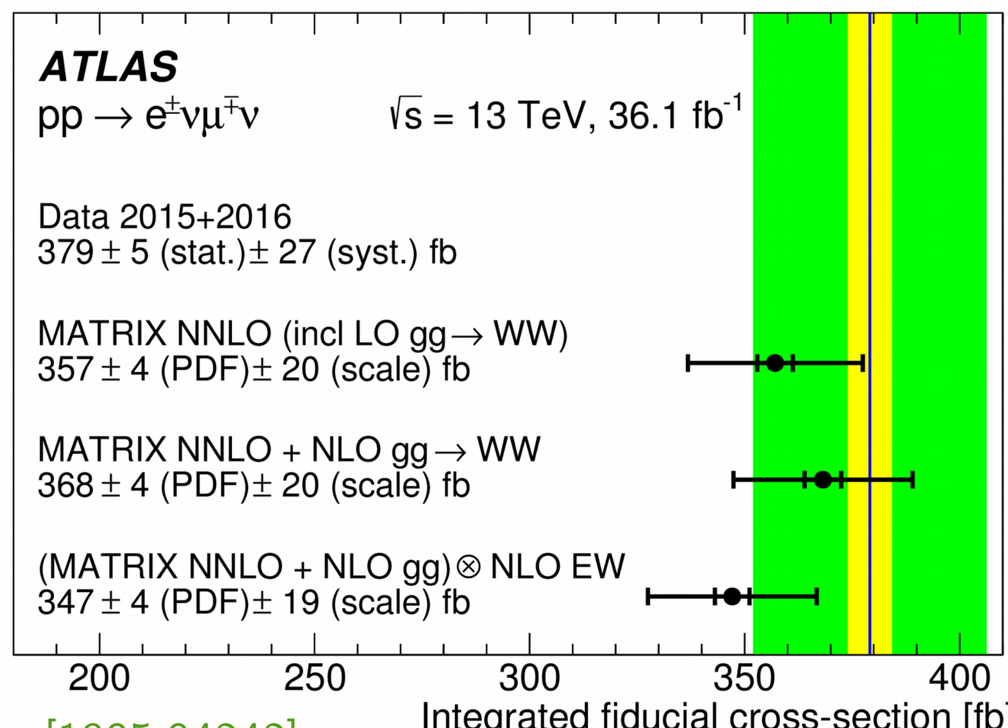
as of April 2017

Why NNLO?

process (${process_id})	$\sigma_{\rm LO}$	$\sigma_{\rm NLO}$	$\sigma_{\rm loop}$ ($\sigma_{\rm loop}/\Delta\sigma^{\rm ext}_{\rm NNLO}$)	$\sigma^{r_{\rm cut}}_{\rm NNLO}$	$\sigma^{\rm extrapolated}_{\rm NNLO}$	$K_{\rm NLO}$	$K_{\rm NNLO}$
$pp \to e^-\mu^+\nu_\mu\bar{\nu}_e$ (ppemxnmnex04)	$232.9(0)^{+6.6\%}_{-7.6\%}$ fb	$236.1(1)^{+2.8\%}_{-2.4\%}$ fb	$26.93(1)^{+27\%}_{-19\%}$ fb (94.3%)	$264.7(1)^{+2.2\%}_{-1.4\%}$ fb	$264.6(2)^{+2.2\%}_{-1.4\%}$ fb	+1.34%	+12.1%

Grazzini, Kallweit, Wiesemann [1711.06631]

- Dibosons processes interesting from BSM perspective (@ high p_T)
- NNLO corrections are large
- K-factors are in general not flat → depend on kinematics so best to have fully differential prediction



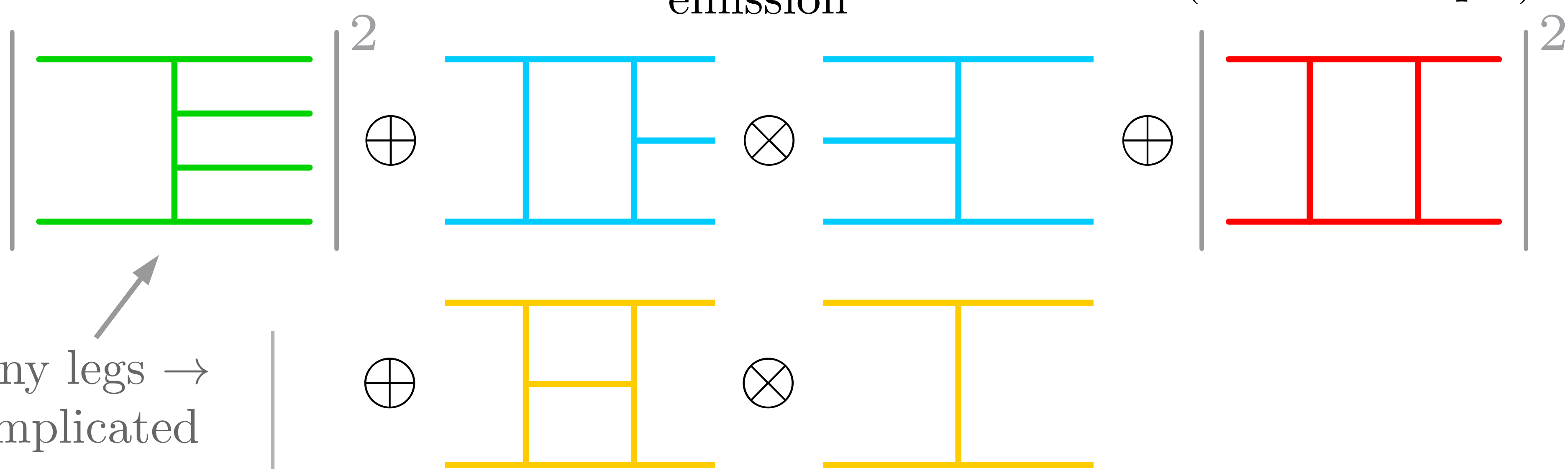
[1905.04242]

Anatomy of NNLO

many legs → large # of diagrams and high dim. phase space

1-loop ⊗ 1-real emission

1-loop squared, e.g. $gg \to ZZ$ contribution to diboson at NNLO (a first example)



many legs → complicated phase-space, inefficient unweighting

see e.g., Bendavid [1707.00028]; Klimek, Perelstein [1810.11509]; Gao, Isaacson, Krause [arXiv:2001.05486]; Gao, Höche, Isaacson, Krause, Schulz [2001.10028]

2-loop ⊗ born → polylogarithms numerically expensive to evaluate

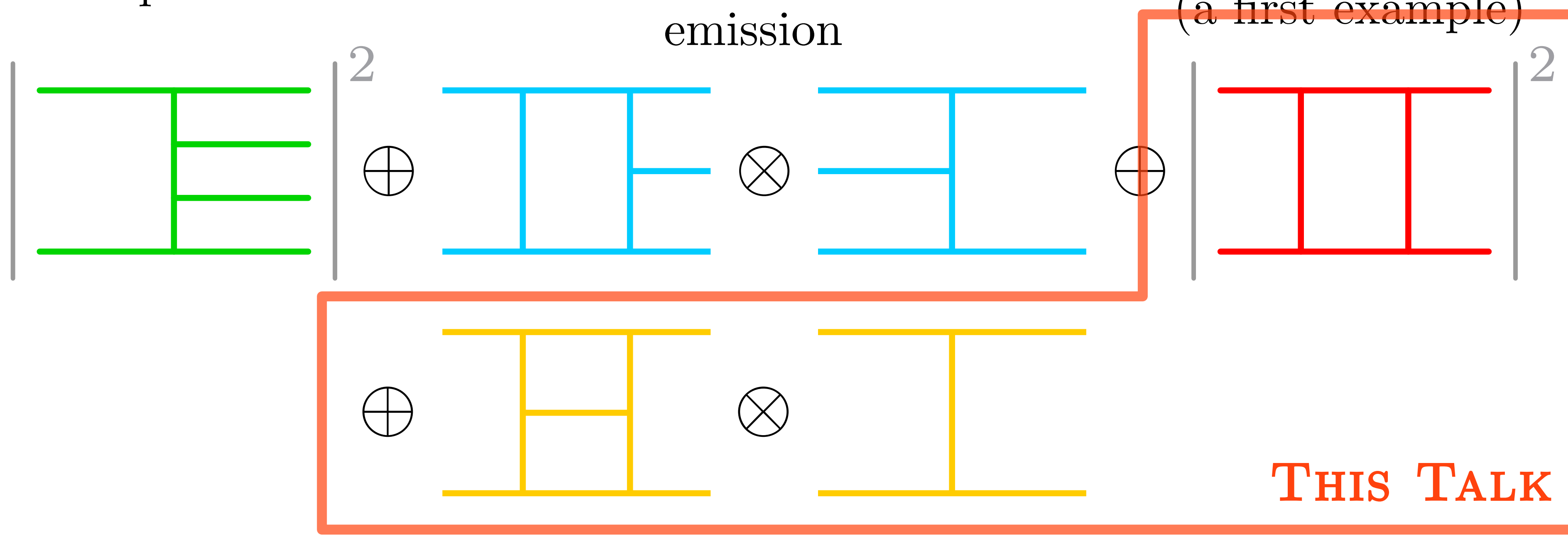
$\mathcal{O}(\alpha_s^2)$ correction $qq \to ZZ$ (a second example)

Anatomy of NNLO

many legs → large # of diagrams and high dim. phase space

1-loop ⊗ 1-real emission

1-loop squared, e.g. $gg \to ZZ$ contribution to diboson at NNLO (a first example)



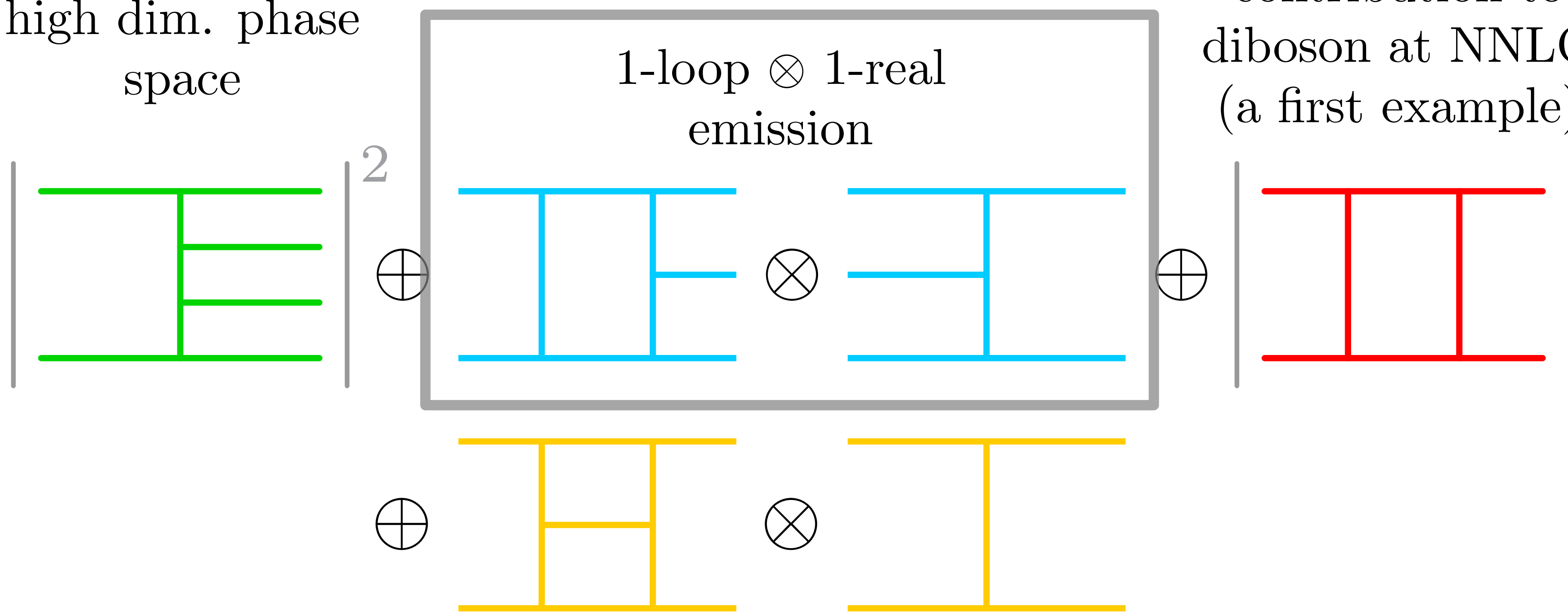
2-loop ⊗ born → polylogarithms numerically expensive to evaluate

Anatomy of NNLO

many legs → large # of diagrams and high dim. phase space

FUTURE WORK

1-loop squared, e.g. $gg \to ZZ$ contribution to diboson at NNLO (a first example)



2-loop ⊗ born → polylogarithms numerically expensive to evaluate

In practice...

Marius Wiesemann

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

MATRIX CPU budget (total runtime)

[Grazzini, Kallweit, MW '17]

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


QUIET NAIL.
WAIT! HAMMER NO!
ML

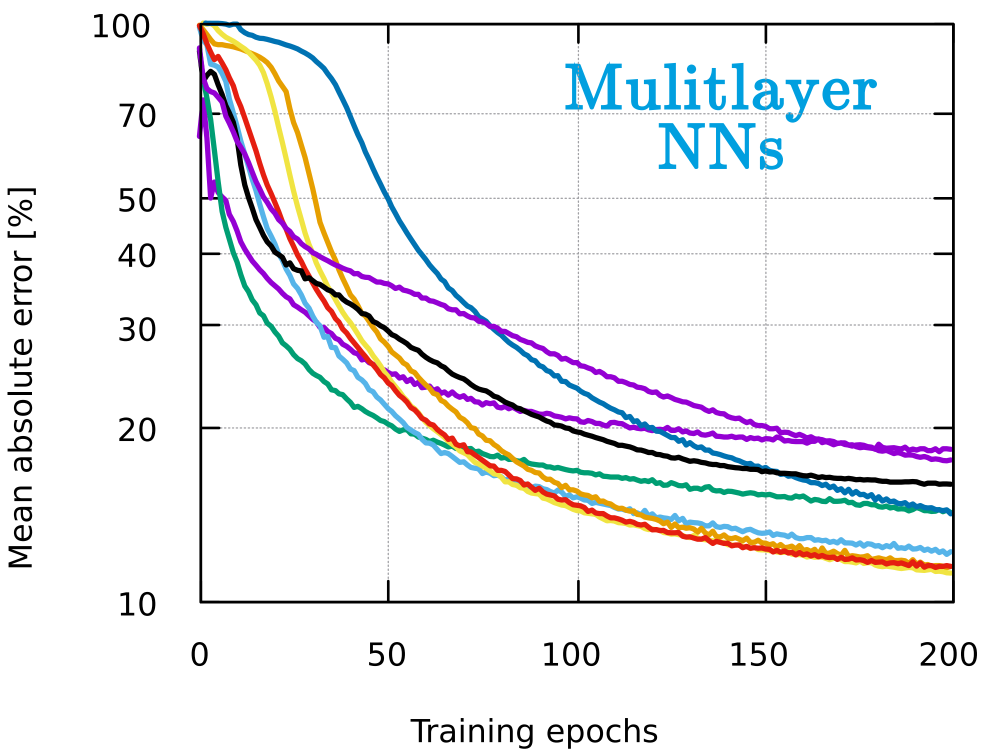
Universal Approximation Theorem

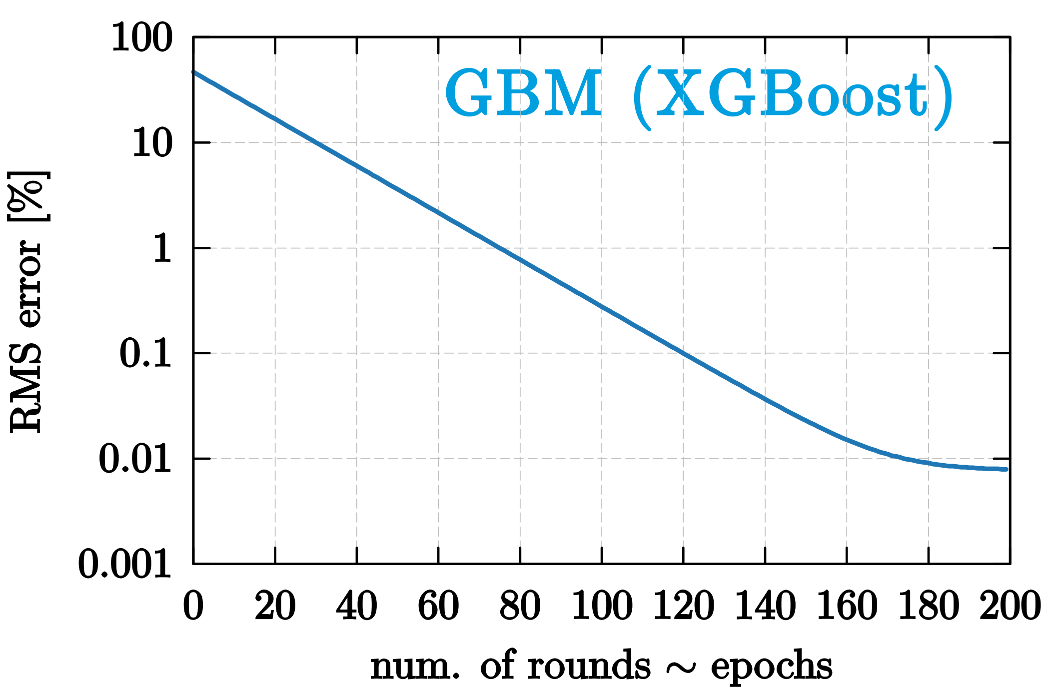
- "...any multivariate continuous function can be represented as a superposition of one–dimensional functions,"

From Braun, J. & Griebel, M. Constr Approx (2009)

GBM vs. DNN

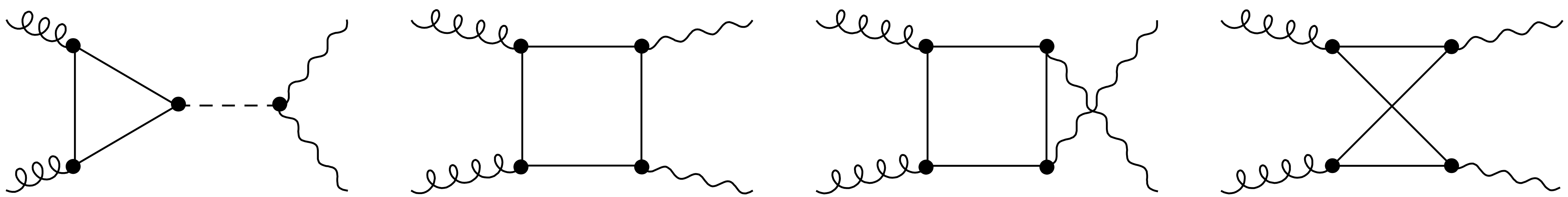
- Nevertheless, in practice converging to the parameters is non-trivial
- We tried with limited success.
- Gradient boosting machines – where no such proof exists (AFAIK) – however, performed extremely well





A first example: gg→ZZ

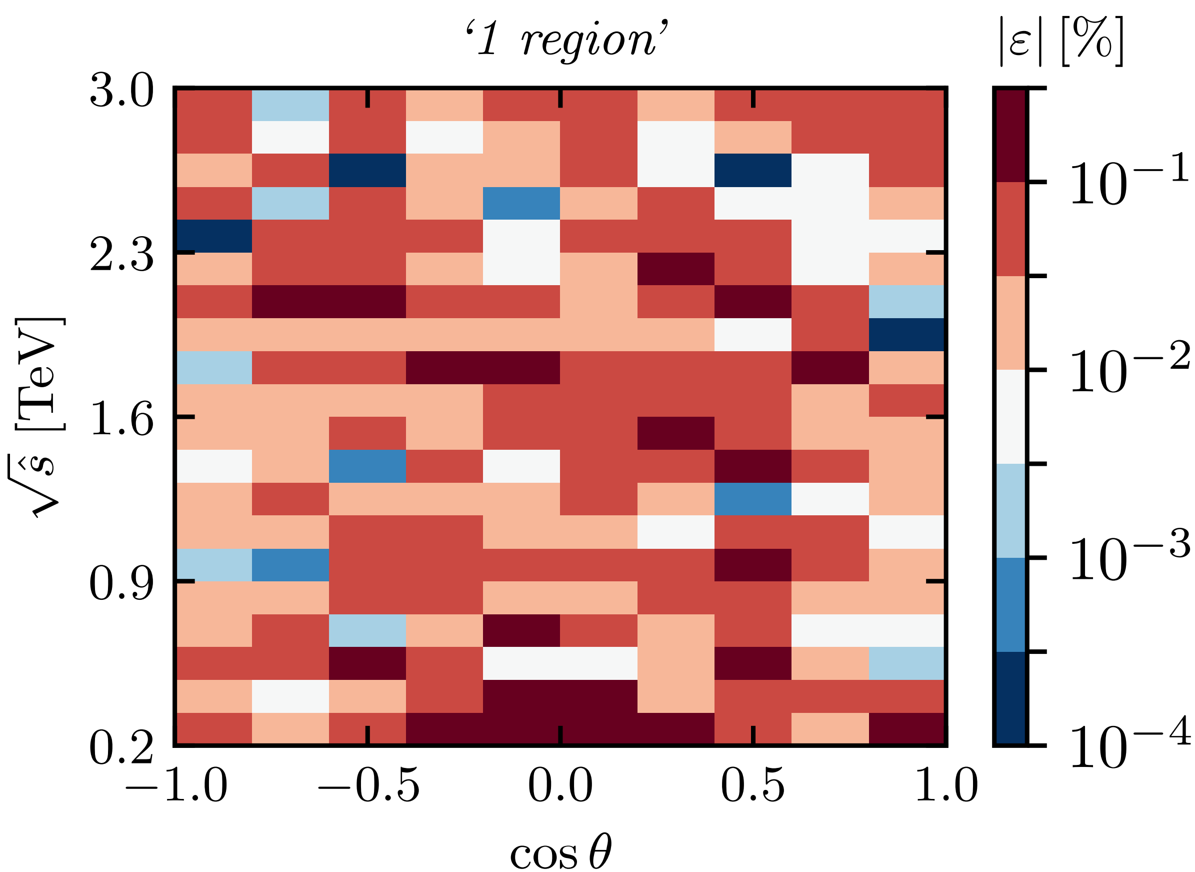
FB, Montull [1912.11055]



- Accounts for the bulk of the NNLO correction to $pp \to ZZ$
- Very simple process $\Rightarrow$ a good test bed for the idea
  - For unpolarized beams, no azimuthal dependence
  - Polar angle distribution is symmetric around $\cos\theta = 0 \Rightarrow$ train on half the distribution
  - No sharp peaks in m_{ZZ} spectrum

Results

- Train 2 models over $m_{ZZ} \in [2m_Z, 3\,\text{TeV}]$ and $[3, 14]$ TeV
- $\varepsilon = 1 - \texttt{approx./exact}$

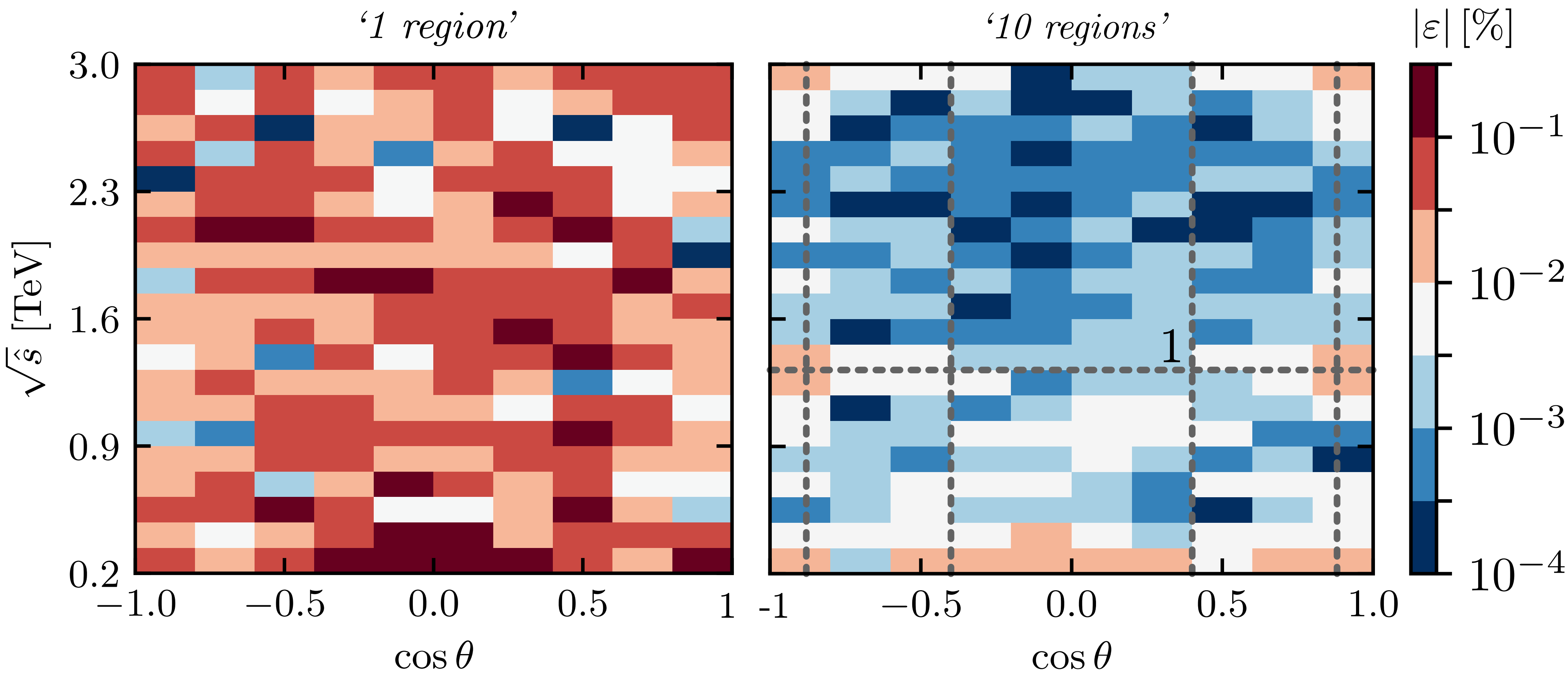


- Use $\cos\theta \to -\cos\theta$ symmetry to train on $\cos\theta \in [-1, 0]$
- Combine models (C libraries) into one driver code and remap $[0, 1]^2 \to$ PS
- A simple tuning of hyperparameter gives excellent results

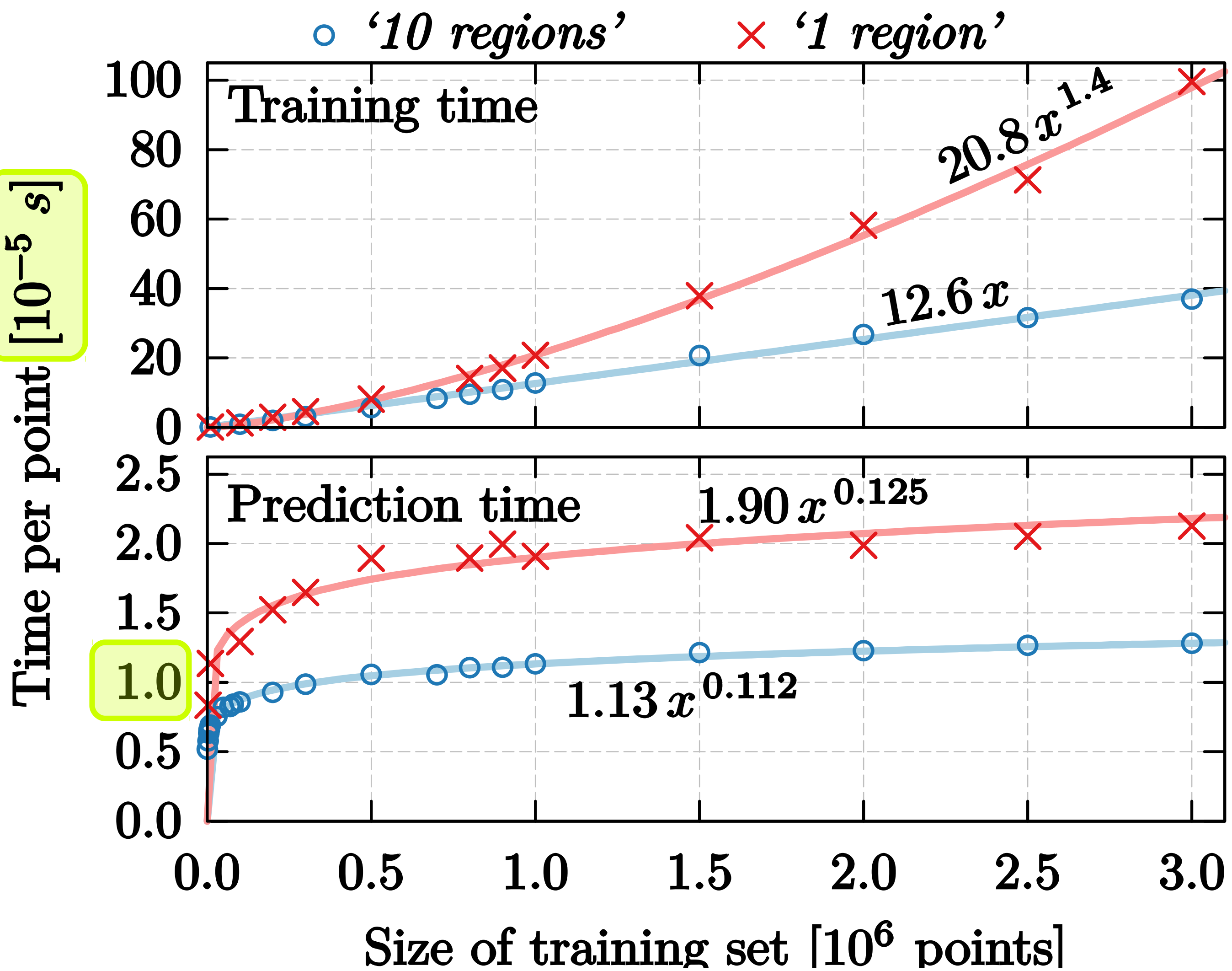
Subdividing the phase space

bin size = 140 [GeV] $\times$ 0.2

$$\varepsilon = 1 - \frac{\texttt{approx.}}{\texttt{exact}}$$



Prediction time



Compare with `OpenLoops` 8.7×10^{-3} [s/point] $\Rightarrow$ 1000-fold speedup!

Prediction time

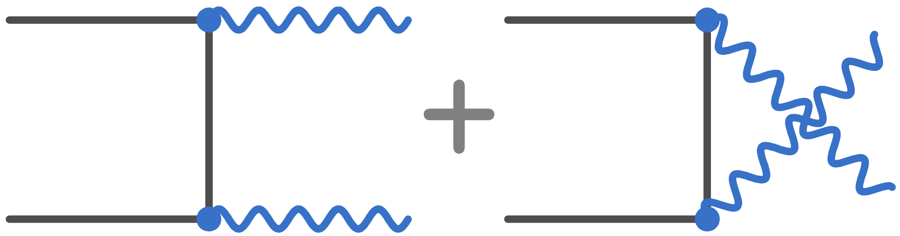
	'1 region'	'10 regions'	
$\lvert\varepsilon^{\text{bins}}_{min}\rvert$ [%]	$7 \cdot 10^{-5}$	$3 \cdot 10^{-5}$	Fig. 3
$\lvert\varepsilon^{\text{bins}}_{max}\rvert$ [%]	0.3	0.03	Fig. 3
$t^{(1\text{ core})}_{\text{predict}}$ [s/point]	$2 \cdot 10^{-5}$	10^{-5}	Fig. 6
$t^{(1\text{ core})}_{\text{train}}$ [s]	977	390	Fig. 2
Size [Mb]	4.8	28	

Table I. Main characteristics of the two ML-regressors trained on 3M points and predicting on 15M. The relative errors $\varepsilon^{\text{bins}}_{max}$ and $\varepsilon^{\text{bins}}_{min}$ stand for the relative errors in the bins of size 140 $\times$ 0.2 ($\sqrt{\hat{s}}$ [GeV], $\cos\theta$).

beyond loop-induced...

ZZ production at hadron colliders

$q\bar{q} \to ZZ$ @ L.O.



Amplitude: $\sum_{j=1}^{10} A_j\left(s,t\right) T_j^{\mu\nu} \varepsilon_{\lambda}^{\mu}\left(p_3\right) \varepsilon_{\lambda'}^{\nu}\left(p_4\right)$

form factors
= scalar fns.
of invariants

in general, 10 tensor structures
@ L.O. only 4: $T_7, \ldots, T_{10}$

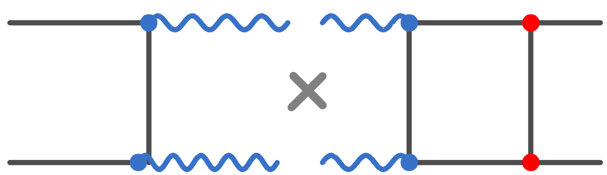
$$T_7^{\mu\nu} = \bar{u}\left(p_2\right) \gamma^{\nu} u\left(p_1\right) p_1^{\mu}$$

$$\mathcal{T}\left(s,t,p_3^2,p_4^2\right) \xrightarrow[\text{shell}]{\text{on}} \mathcal{T}(s,t) \equiv \sum_{\text{pols,cols}} \left|S_{\mu\nu}\varepsilon_{\lambda}^{\mu}\left(p_3\right) \varepsilon_{\lambda'}^{\nu}\left(p_4\right)\right|^2$$

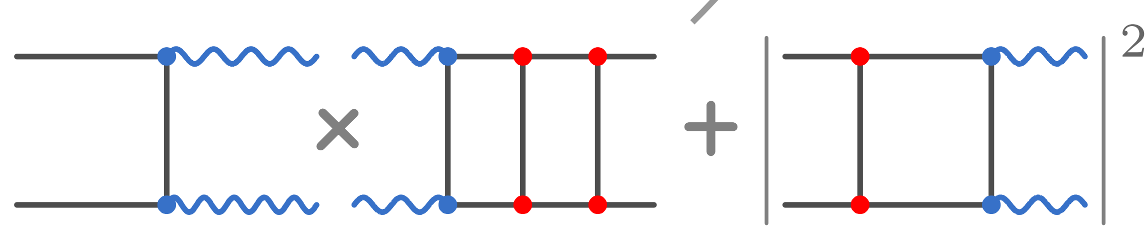
* Following the notation of [1503.04812]

qq→ZZ up to NNLO in α_s

$$\mathcal{T} = (4\pi\alpha)^2 \left[\mathcal{T}^{(0)}(s,t) + \left(\frac{\alpha_s}{2\pi}\right) \mathcal{T}^{(1)} + \left(\frac{\alpha_s}{2\pi}\right)^2 \mathcal{T}^{(2)} \right]$$



$$\mathcal{T}^{(1)} = 2\Re\left\{\left\langle \mathcal{M}^{(0)} | \mathcal{M}^{(1)} \right\rangle\right\}$$



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

- Extract A_i from `VVAMP` (computed to $\mathcal{O}(\alpha_s^2)$)
  Gehrmann, von Manteuffel, Tancredi [1503.04812], https://vvamp.hepforge.org
- Compute and fit $\mathcal{T}^{(2)} / \langle \mathcal{M}^0 | \mathcal{M}^0 \rangle$

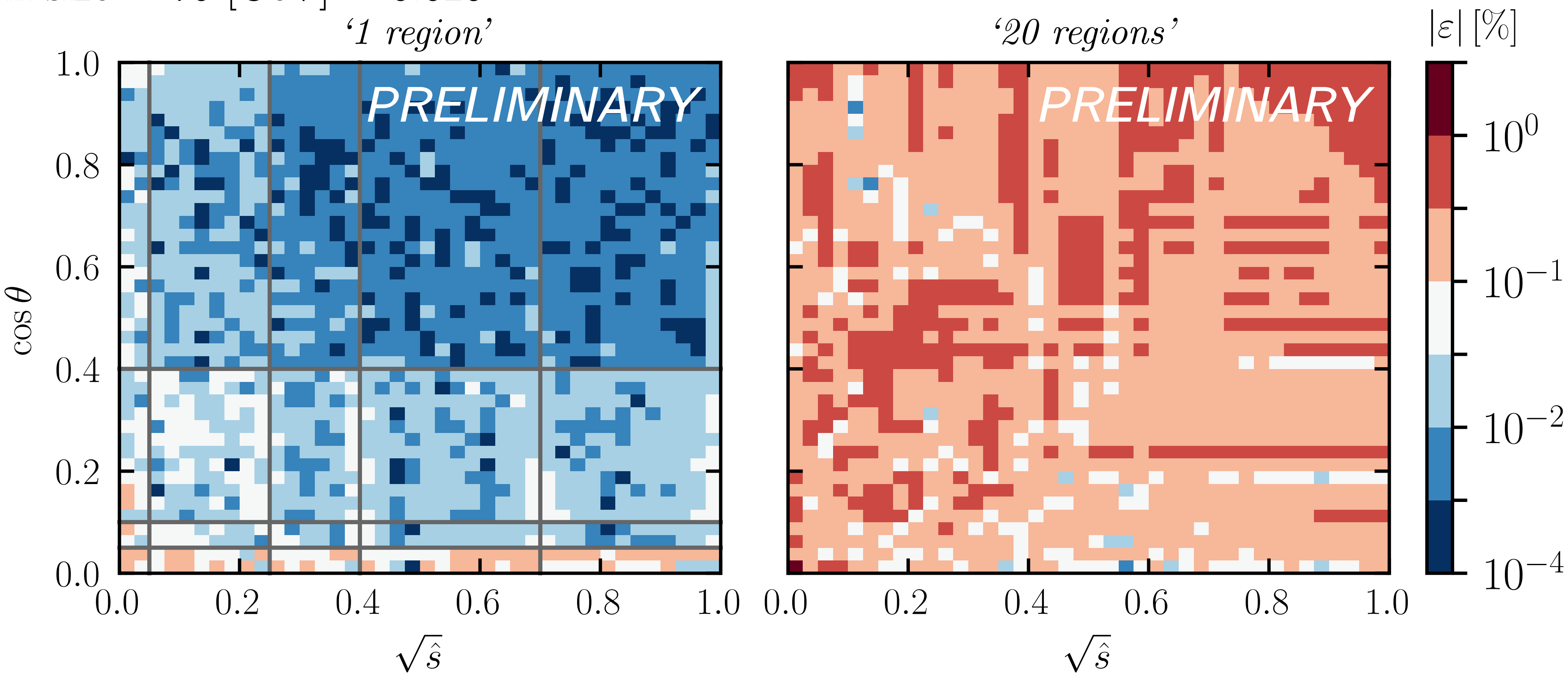
Datasets for fitting and testing

- Phase space is simple (2-dim.): $\{s, t\}$ or equiv. $\{\sqrt{s}, \cos\theta\}$
- $|\mathcal{M}|^2$ symmetric under $\cos\theta \to -\cos\theta$
- Generate 1M random points *uniformly* on the unit square and compute finite part of A_i with `VVAMP` in q_T scheme with $\mu = \sqrt{s}$; [uniform dataset]
- Compute $\mathcal{T}^{(2)}$ and sample another 1M points on the unit square from the 2d histogram of $\mathcal{T}^{(2)}$; [weighted dataset]

Train on **weighted** dataset; predict on the **uniform**

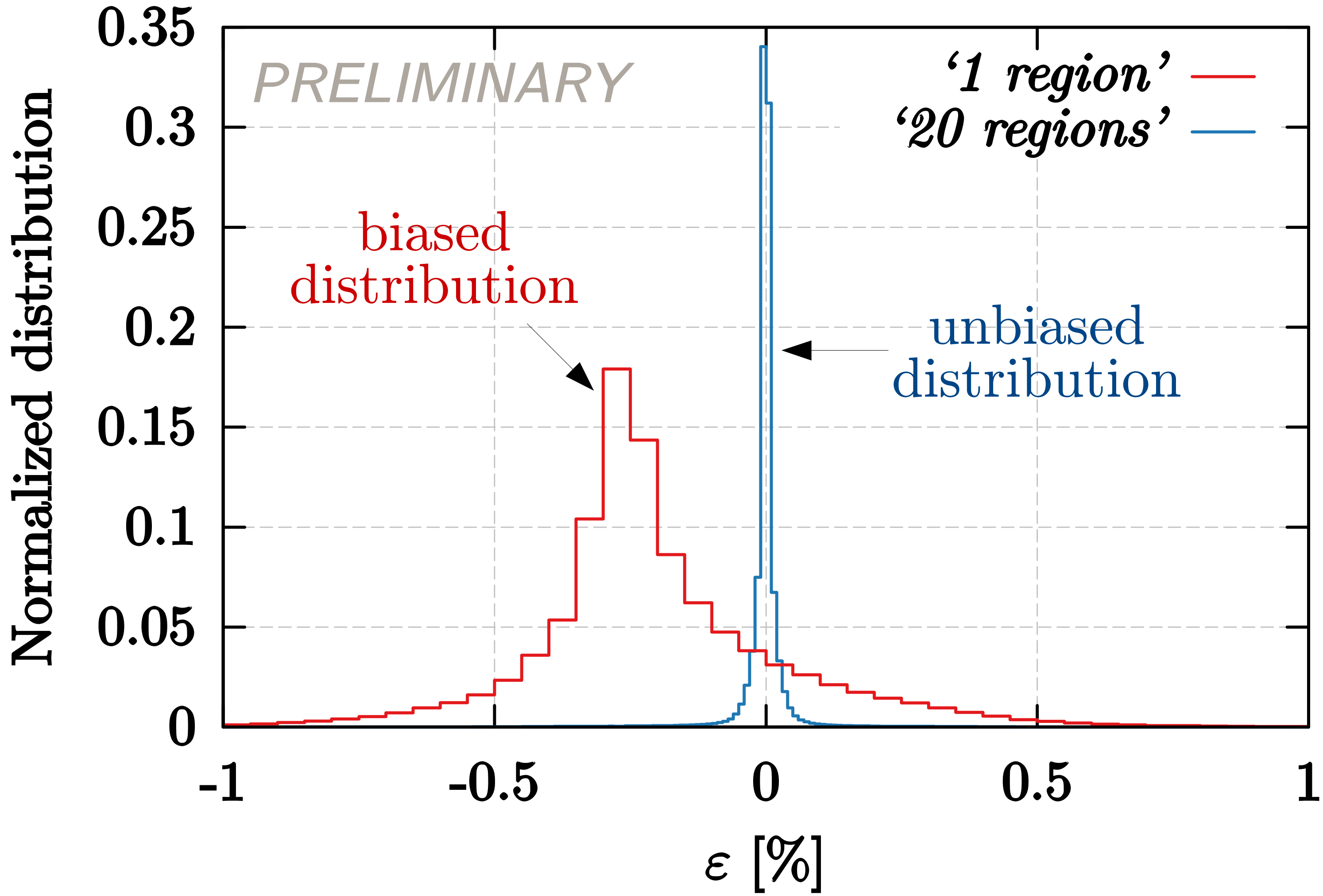
Double differential distributions

bin size = 70 [GeV] $\times$ 0.025



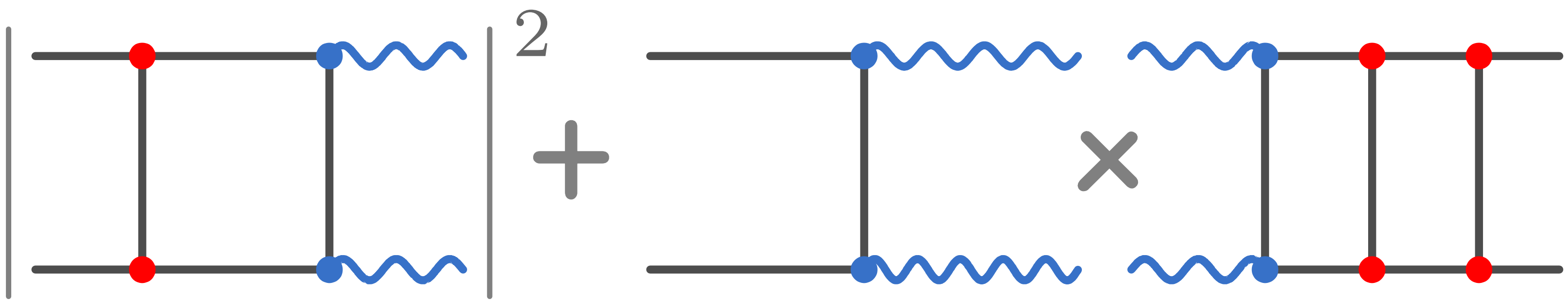
- Region boundaries delineated by dashed grey lines
- Subdividing phase-space reduces the approximation error by $\sim$ an order of magnitude! And ...

Approximation error distribution



Subdividing phase-space makes the distribution of errors unbiased (i.e., centered around zero) → integrating will reduce errors

Timing



`VVAMP`: $\sim$ 16s / point
`XGBoost`: $\sim$ 16s / **1M** points

Speedup gain of 10^6 over evaluation of exact $|\mathcal{M}|^2$!

Summary

- ✔ Proof of principle (loop-induced) performs extremely well: 10^3 speedup and with errors well below 0.1%
- ✔ Fitting the $\mathcal{O}(\alpha_s^2)$ Born $\times$ 2-loop and (1-loop)^2 also works extremely well: 10^6 speedup
- ✔ For $gg \to ZZ$, we now have a standalone fully functional library
- ✔ Standalone library for $qq \to ZZ$ on `HEPForge` or `git` soon
- ✔ Include $qq \to V_1 V_2$ with $V_{1,2} = W, Z, \gamma$ next

Outlook

- In principle, all slow $2 \to 1$ and $2 \to 2$ can be done
- ☐ Ideally include mixed QCD/EW corrections since important for BSM VV
- ☐ Also working on more complicated processes, e.g., $2 \to 3$ loop-tree interference
- ☐ Systematically report approx. errors for every point
- ☐ Report 7 scale variation

> We approximated squared matrix elements with gradient boosting machines. Speedup gain of 1000 over exact function @ 1-loop and 10^6 @ 2-loops ; relative errors $\lesssim \mathcal{O}(10^{-3})$!

Thank you!

Universal approx. theorem

From Braun, J. & Griebel, M. Constr Approx (2009);

- "...any multivariate continuous function can be represented as a superposition of one–dimensional functions,"

Kolmogorov, Dokl. Akad. Nauk USSR 14(5), 953–956 (1957)

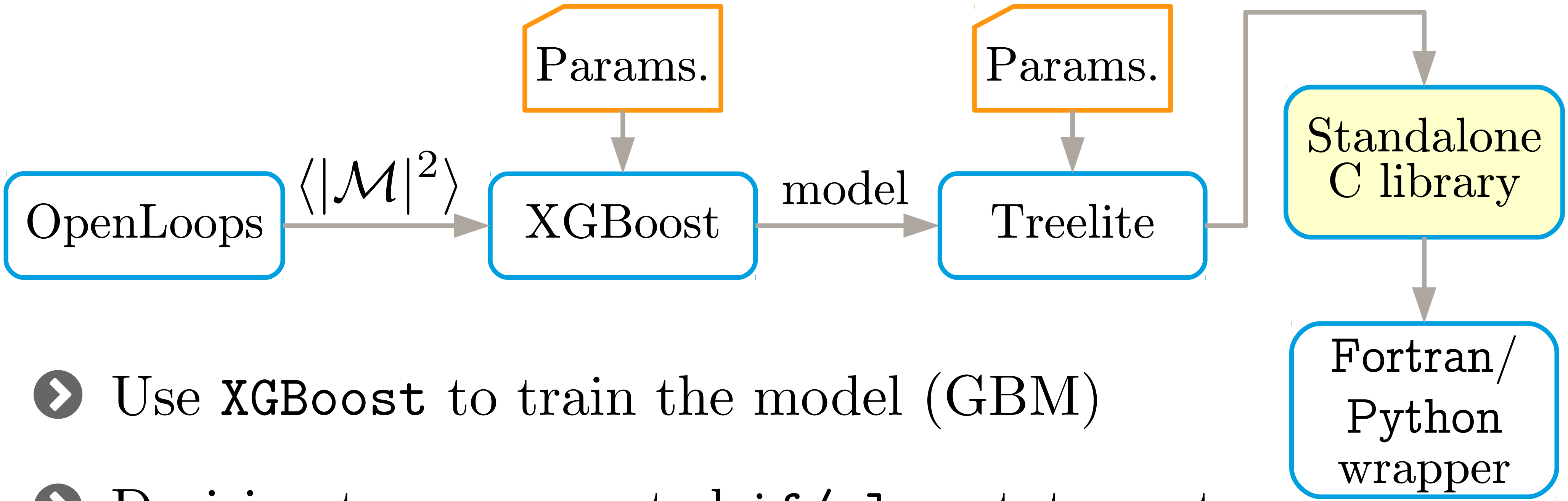
$$f(x_1, \dots, x_n) = \sum_{q=0}^{2n} \Phi_q \left(\sum_{p=1}^{n} \psi_{q,p}(x_p) \right)$$

- Later, constructive proof by Cybenko (1989), for $\varphi : \mathbb{R} \to \mathbb{R}$, $N \in \mathbb{Z}_+$, v_i, $b_i \in \mathbb{R}$, $w_i \in \mathbb{R}^n$ define

$$F(x) = \sum_{i=1}^{N} v_i \varphi \left(w_i^T x + b_i \right)$$

as a representation of $f : \mathbb{R} \to \mathbb{R}$ such that $|F(x) - f(x)| < \varepsilon$ for $x \in [0,1]^n$ and any $\varepsilon > 0$

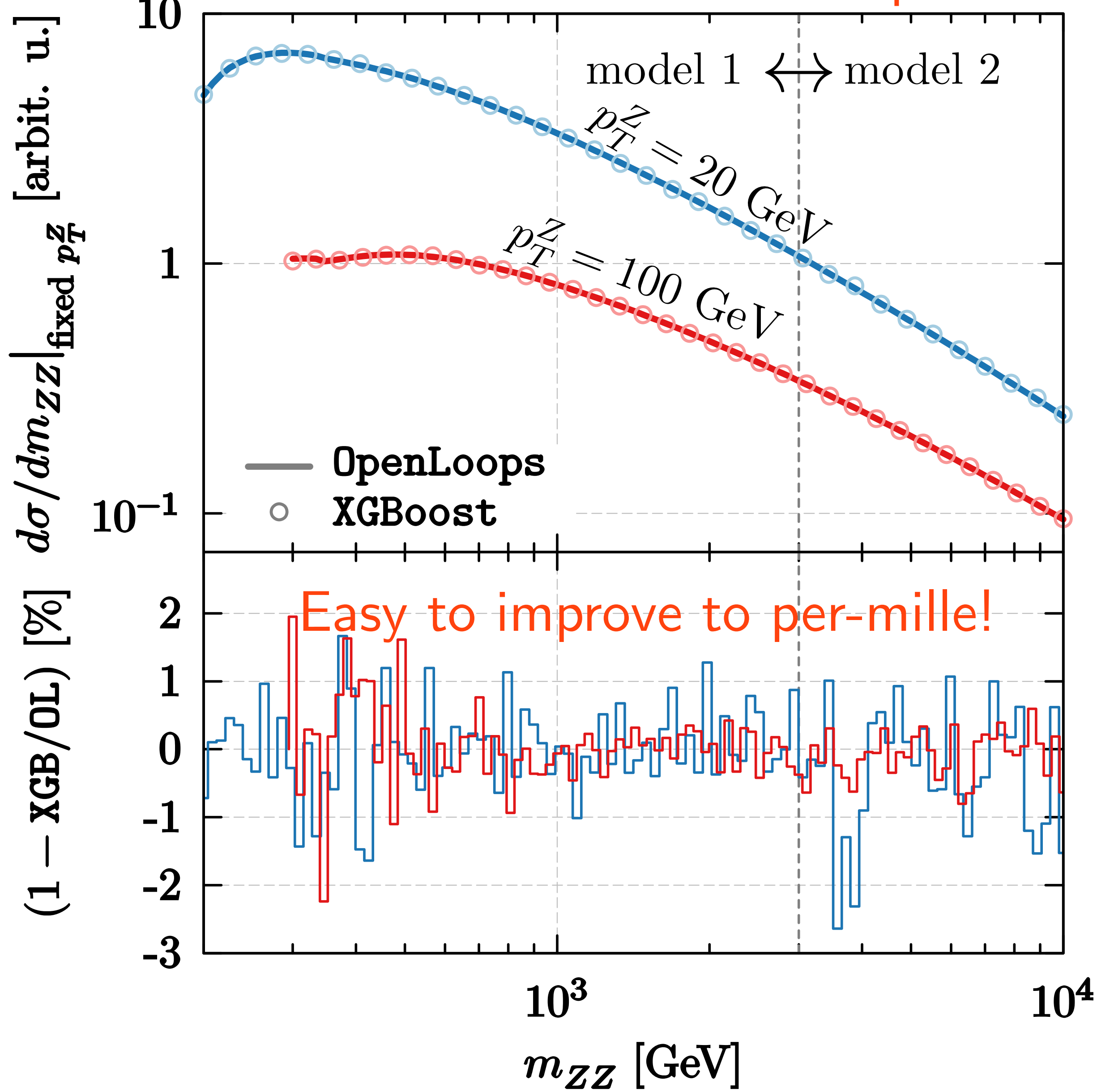
Implementation



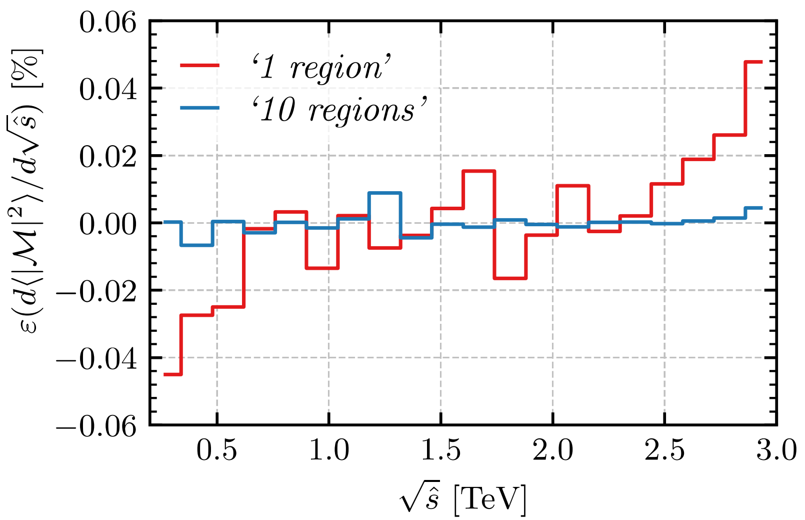
- Use `XGBoost` to train the model (GBM)
- Decision trees are nested `if/else` statements $\Rightarrow$ models are easy to turn into standalone libraries
- Use `Treelite` for further optimization (floating point $\rightarrow$ integer split thresholds, CPU optimization) **and** to output the standalone C library
- Successfully wrapped in `Fortran` and `Python`

Singly-differential distribution gg→ZZ

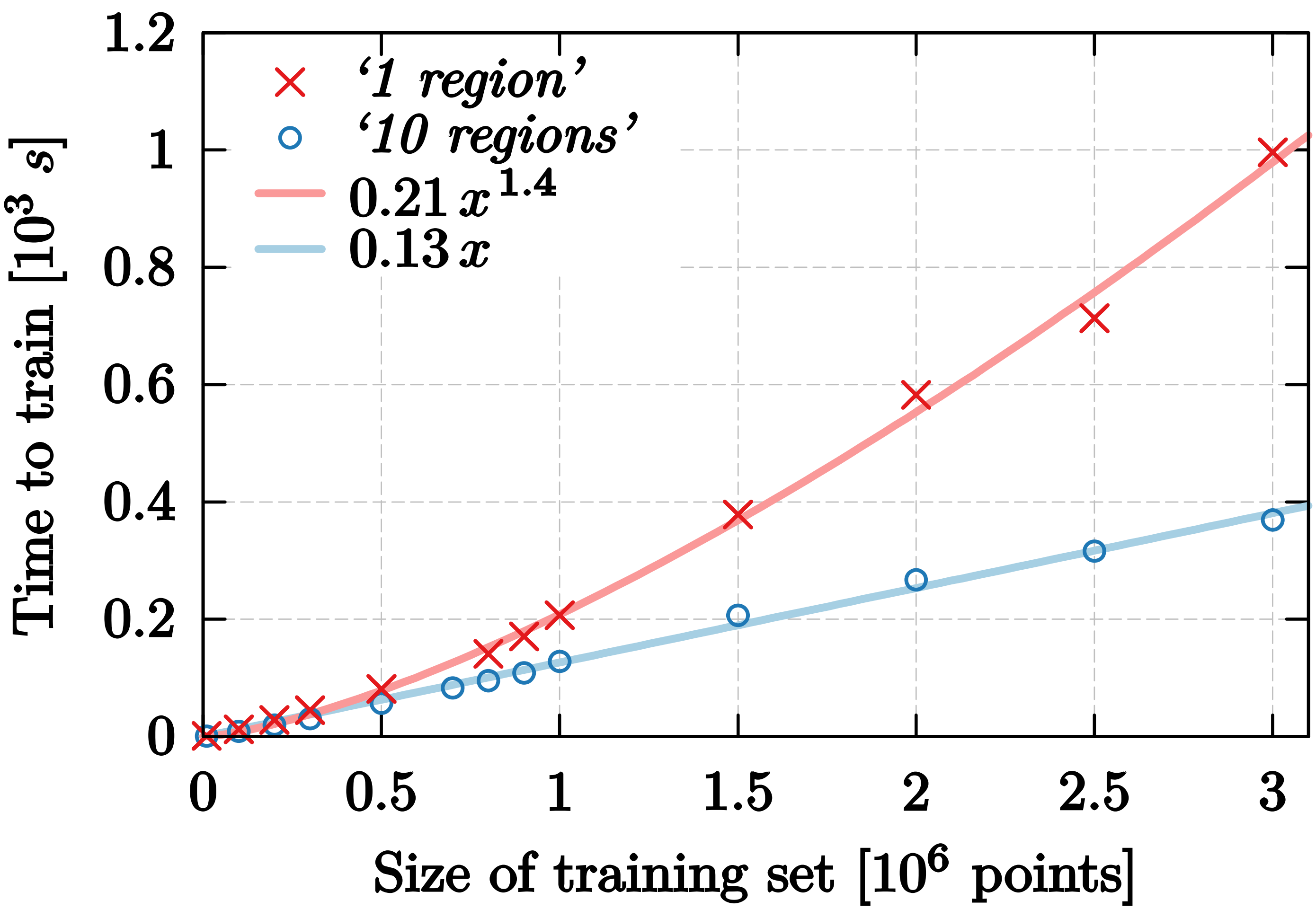
Each model trained on 1M points



Easy to improve to per-mille!

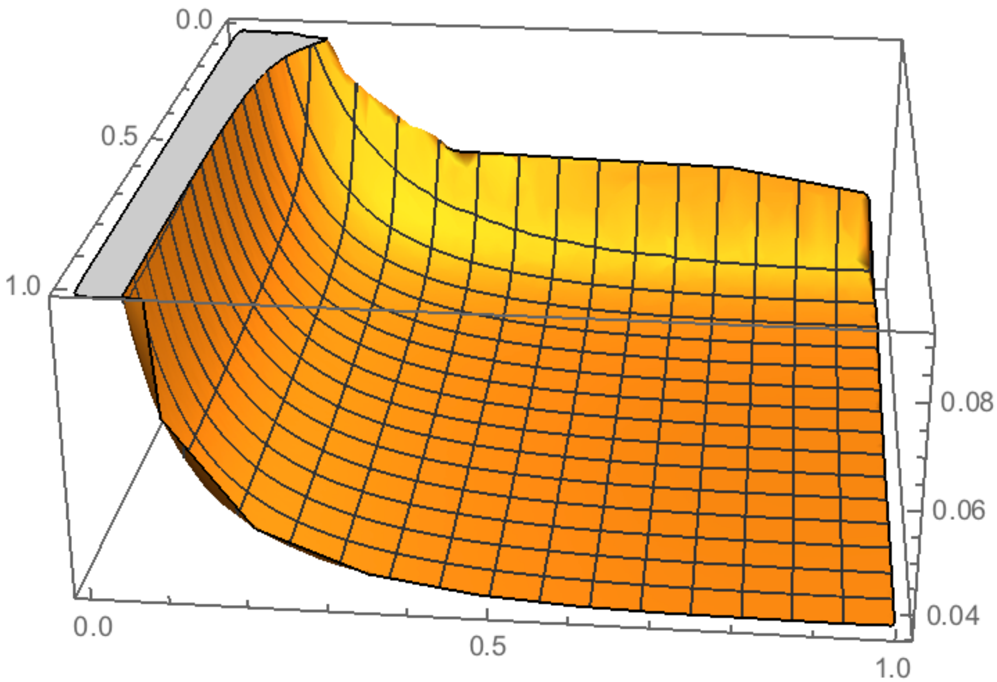


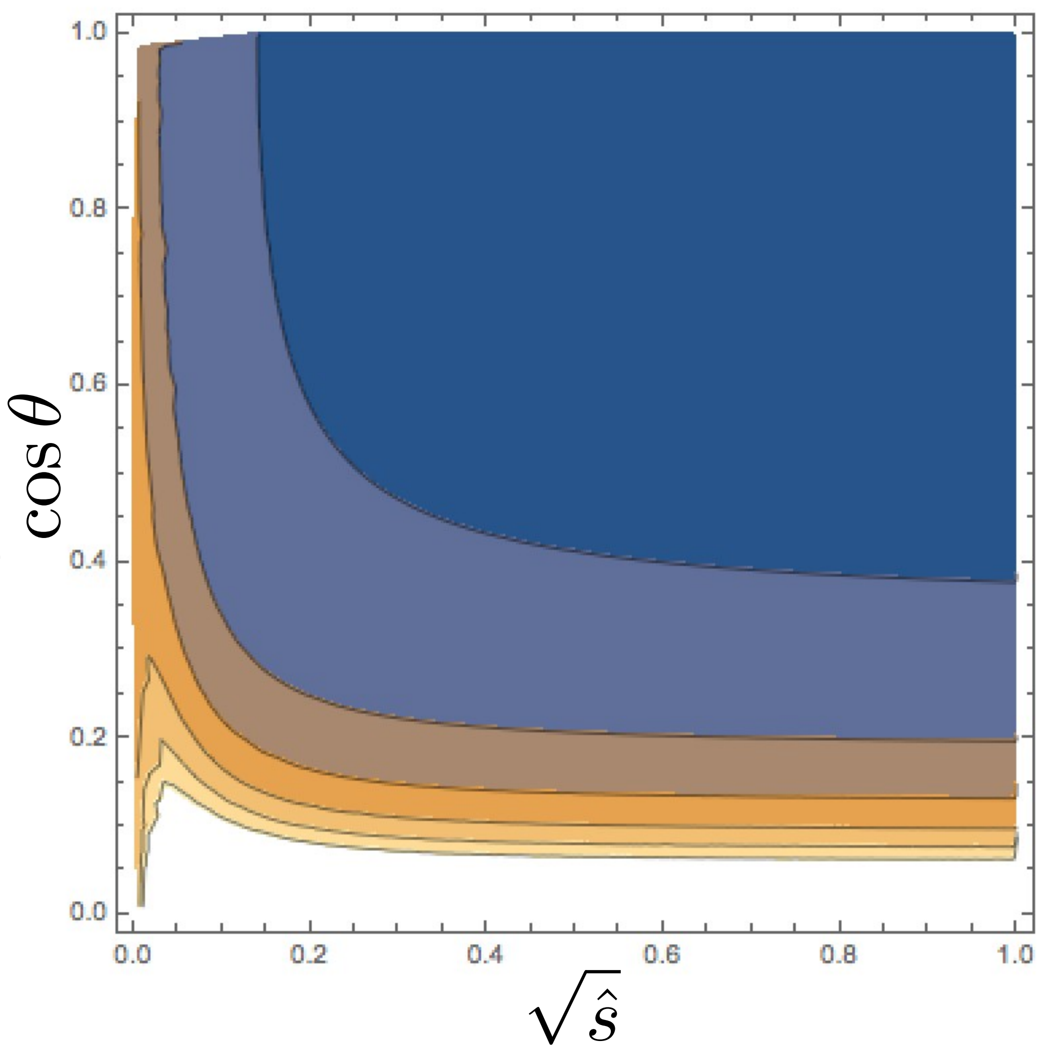
Trainng time on 1 CPU core



$T^{(2)}$ for qq→ZZ

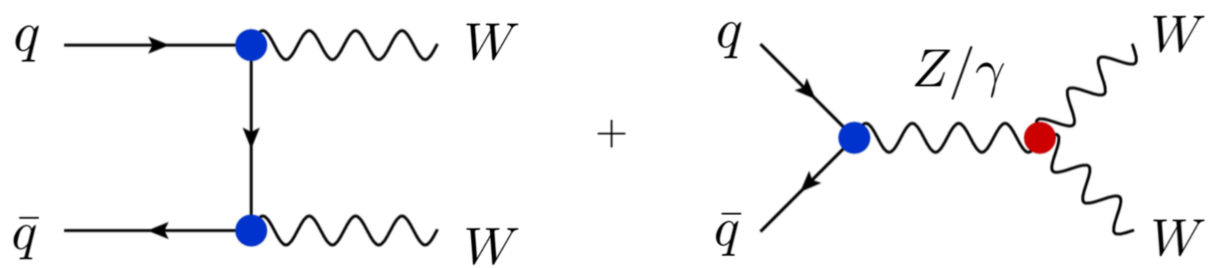
$\cos\theta \in [-1, 0]$ and normalized to $[0, 1]$



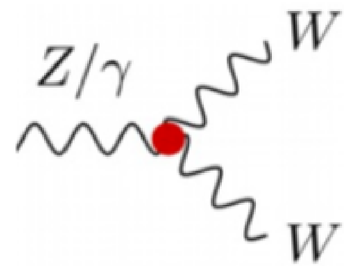


WW production

[Butter et al.: 1604.03105]
[Azatov et al.: 1707.08060]
[Grojean et al.: 1810.05149]
[+ more]



Improving LEP-2 bounds on
anomalous Triple Gauge Couplings



Bounds on aTGC

Butter et al 1604.03105

	LHC Run I				LEP			
	68 % CL	Correlations			68 % CL	Correlations		
Δg_1^Z	0.010 ± 0.008	1.00	0.19	−0.06	$0.051^{+0.031}_{-0.032}$	1.00	0.23	−0.30
$\Delta\kappa_\gamma$	0.017 ± 0.028	0.19	1.00	−0.01	$-0.067^{+0.061}_{-0.057}$	0.23	1.00	−0.27
λ	0.0029 ± 0.0057	−0.06	−0.01	1.00	$-0.067^{+0.036}_{-0.038}$	−0.30	0.27	1.00

Per mille at LHC !!

Percent at LEP

Slide from Marc Montull