Global fits and Machine Learning

Veronica Sanz

EXCELENCIA SEVERO OCHOA
IFIC INSTITUT DE FÍSICA CORPUSCULAR
CSIC
VNIVERSITAT DE VALÈNCIA

HEFT workshop '24 (Bologna)

Outline

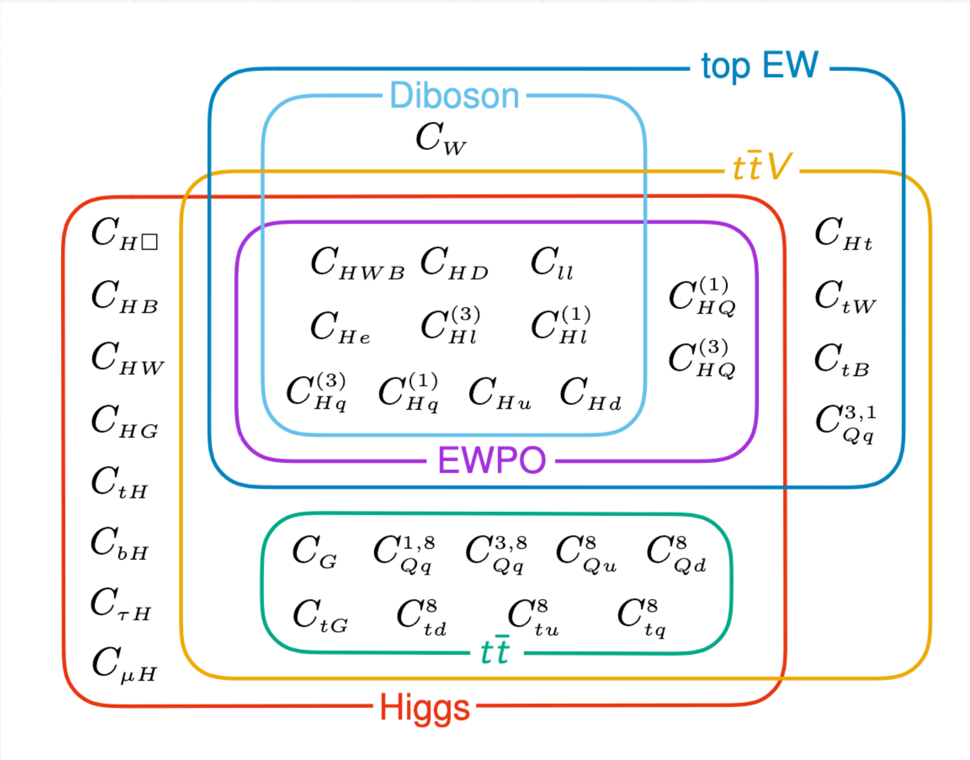
Key points on global fits for SMEFT

Do we *really* need ML?

An example of using ML to optimise global fits

Key points on SMEFT global analyses

State-of-the-art



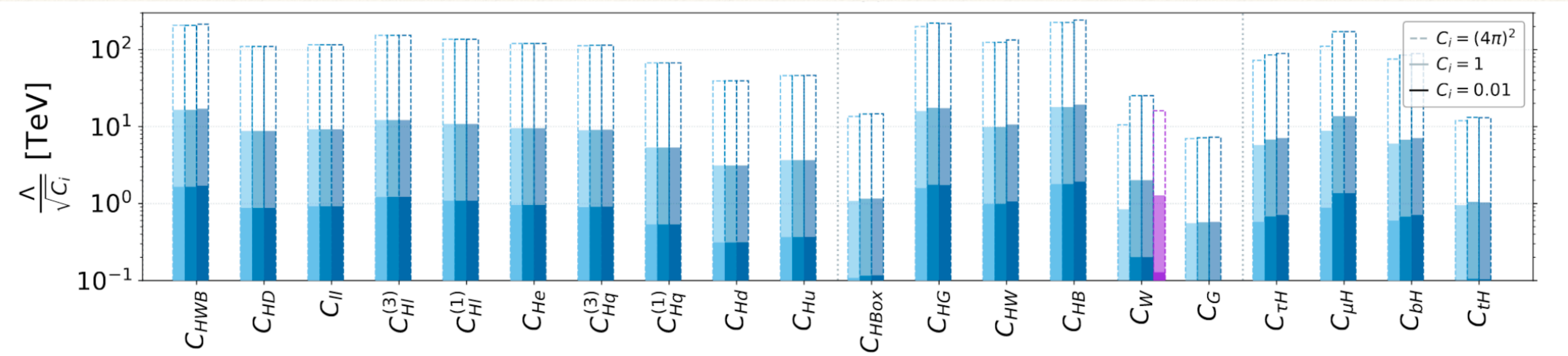
Global EFT analyses nowadays use EWPT, LEP WW, LHC di/tri-boson Higgs, Top, HTop, 4F from LEP, Tevatron, LHC Run1 and Run2 inclusive and differential and sometimes low-energy & flavour

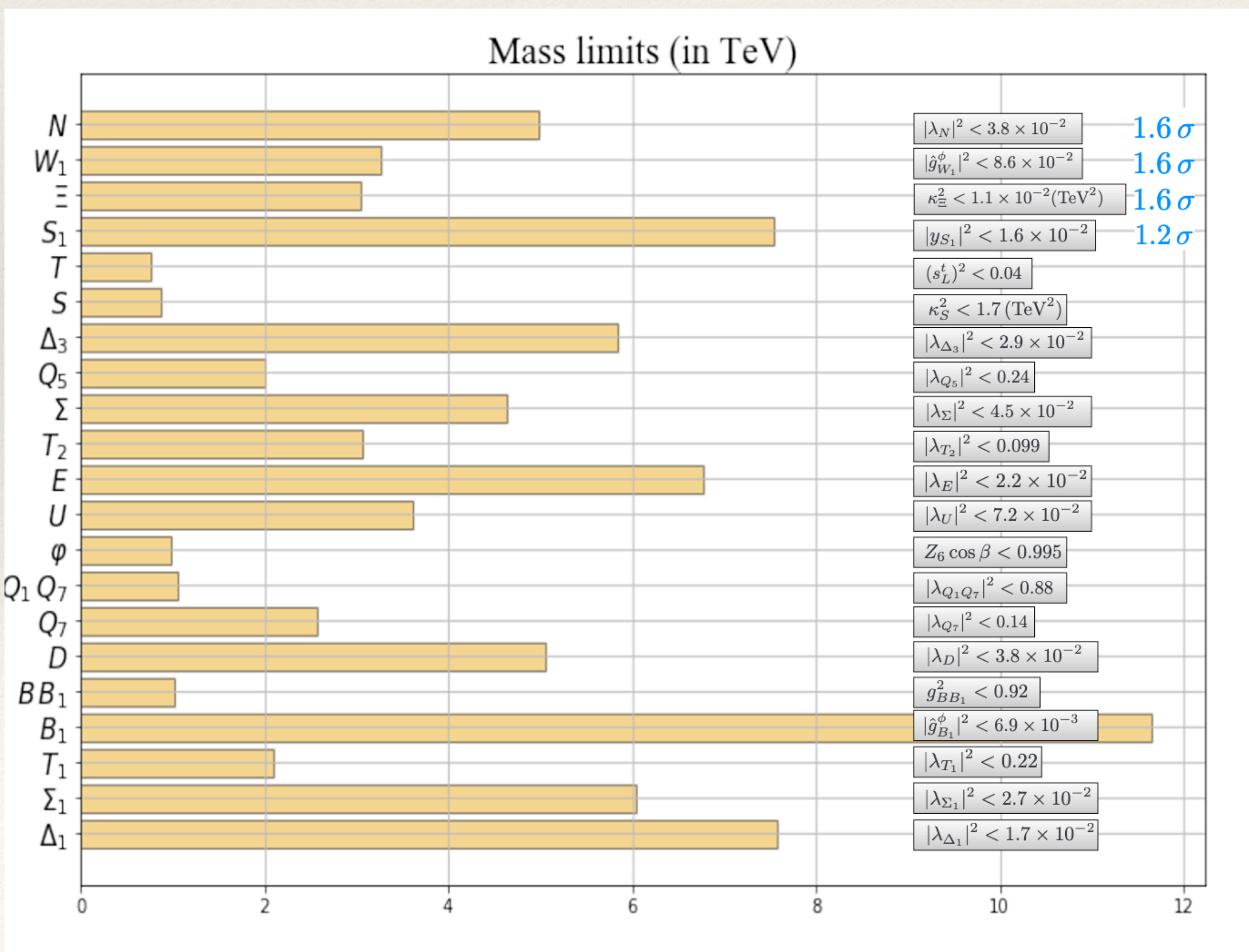
So it's a game of matching hundreds of observables with a very large parameter space, and give a **consistent view** when all EFT directions are taken into account

This is very tricky, theoretically and experimentally
e.g. *Fitmaker*: ~350 obs, ~20 EFT coeffs

Current SMEFT constraints reach the TeV for most of t he param space

Ellis, Madigan, Mimasu, VS, You
2012.02779, JHEP



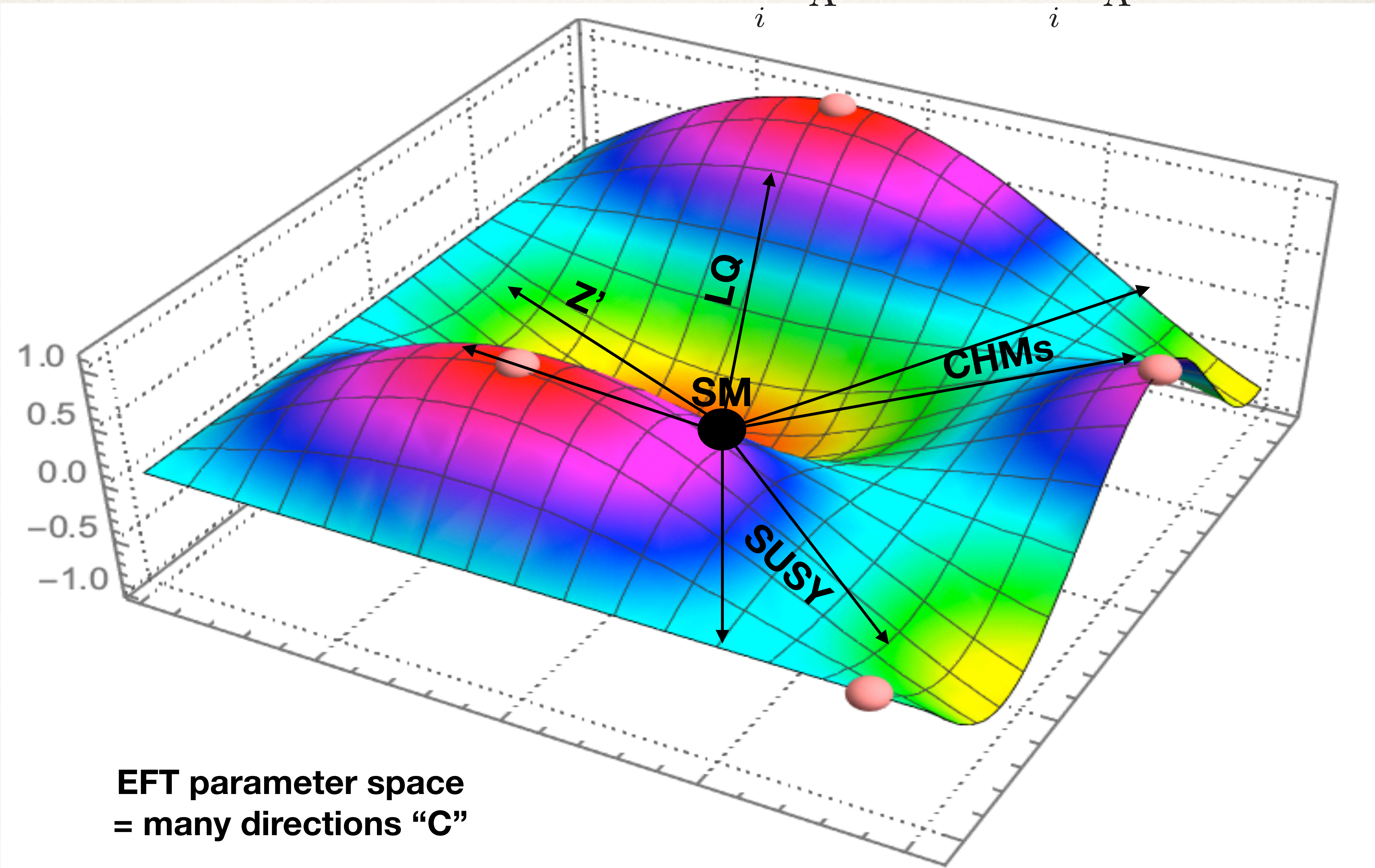


And when translated into vanilla extensions of the SM, the mass limits are also probing the TeV scale

Here: *Fitmaker*
but also *SMEFit*, *TopFitter*, *SFitter*, *HEPFit*…

SMEFT = model-independent (?)

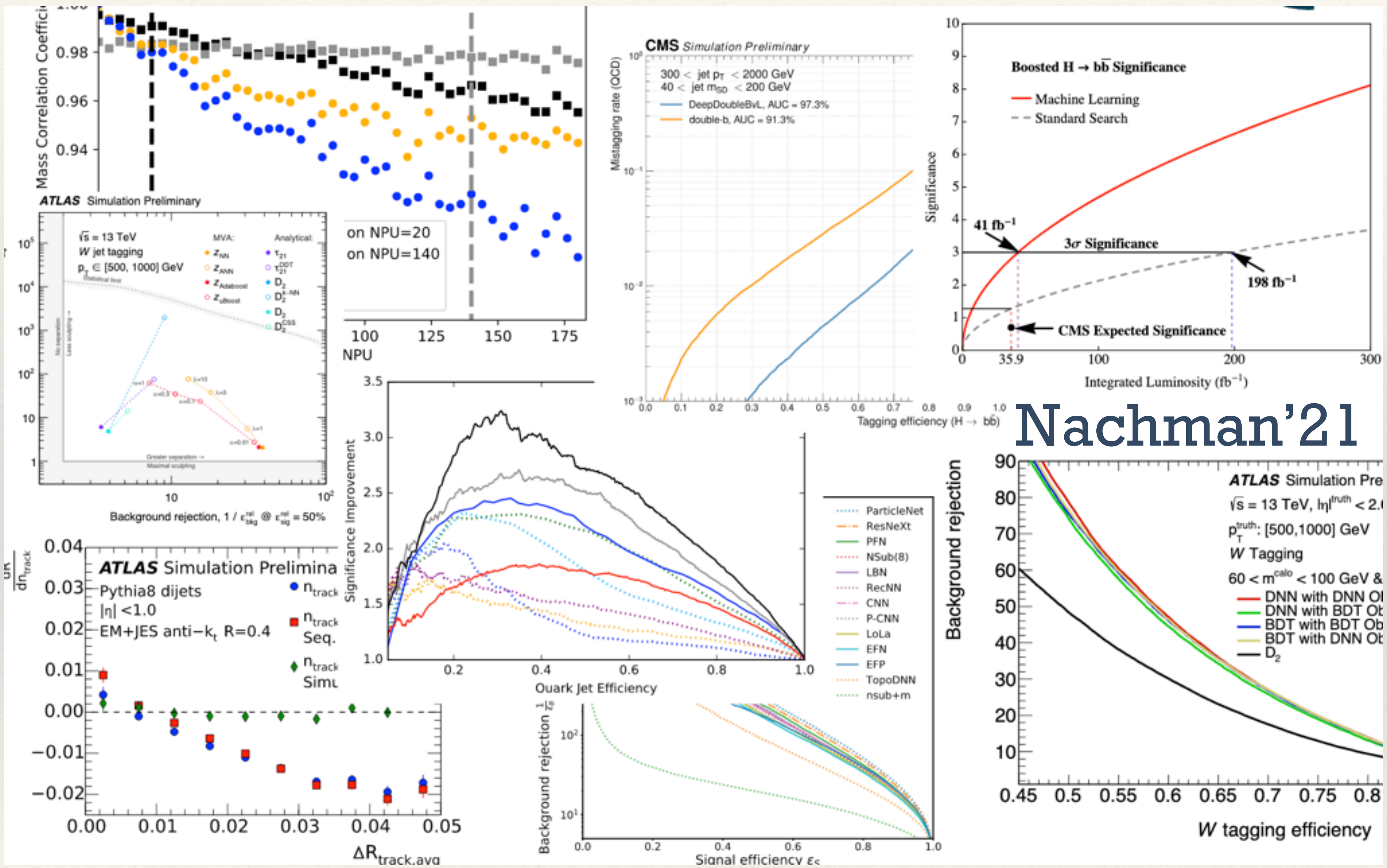
$$\mathcal{L} = \mathcal{L}_{SM} + \sum_i \frac{c_{6,i}}{\Lambda^2}\,\mathcal{O}^{d=6} + \sum_i \frac{c_{8,i}}{\Lambda^4}\,\mathcal{O}^{d=8} + \ldots$$



EFT parameter space
= many directions "C"

Do we *really* need Machine Learning?

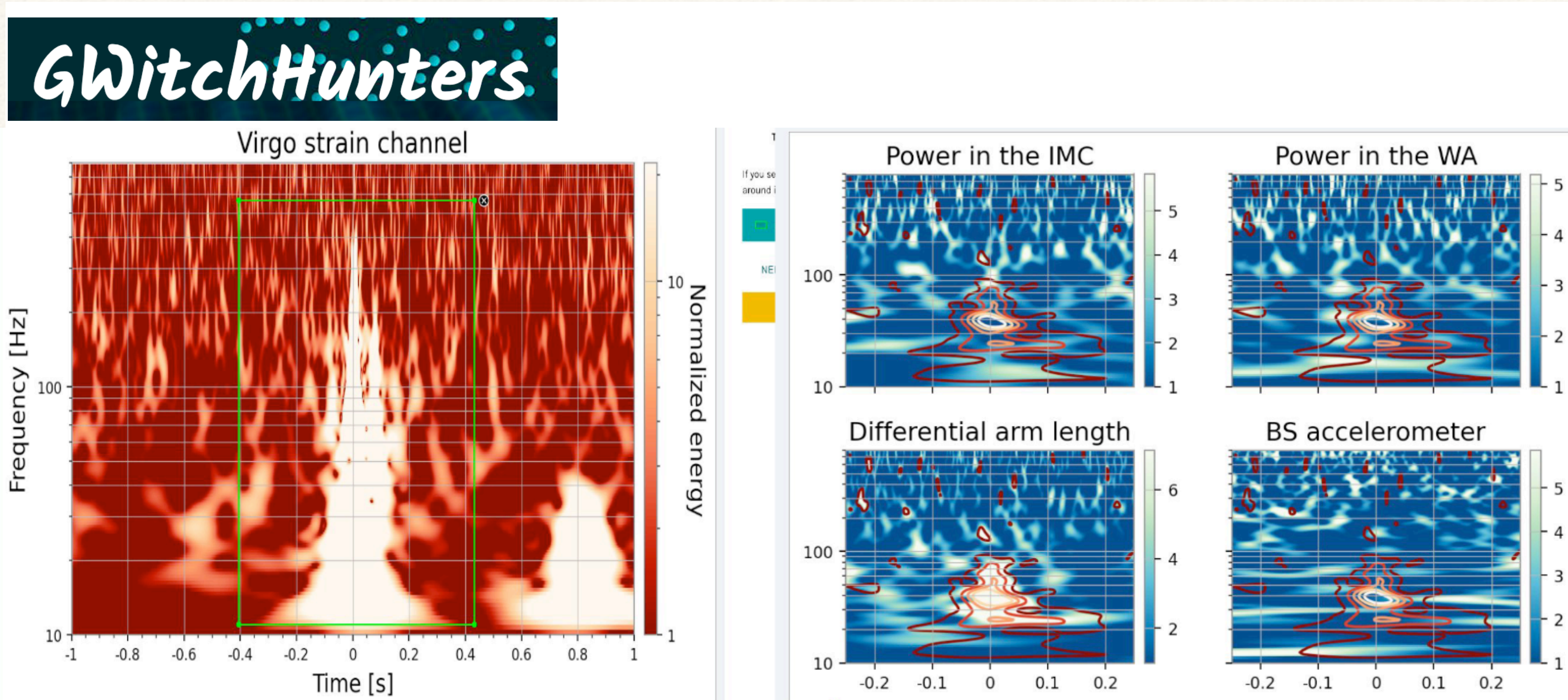
Sure, ML is useful to squeeze the last drops



In LHC physics gains in ID-ing phenomena are typically in the range of 5%-30%

for tricky environments:
difference between discovery or not

…or modelling truly difficult things

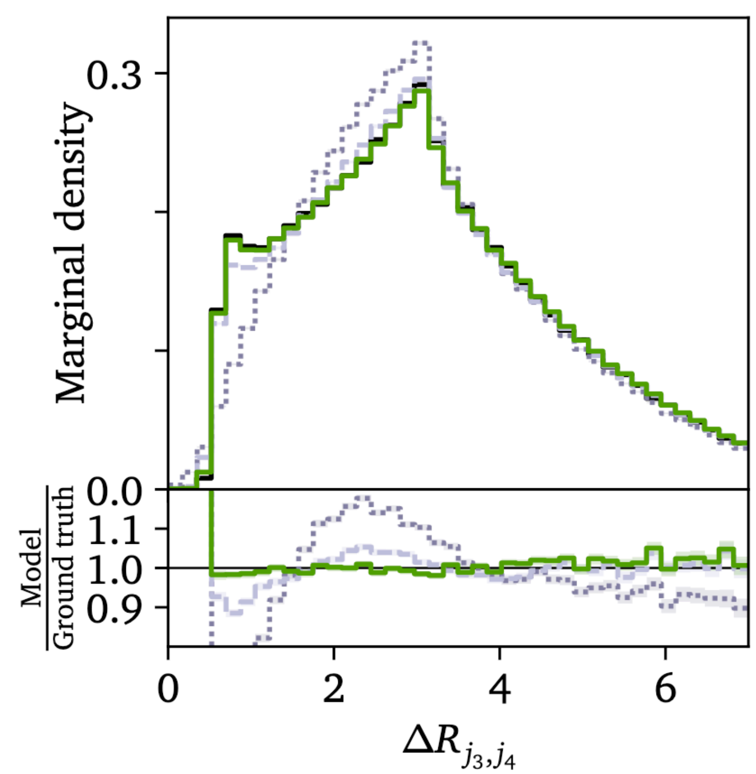


from Elena Cuoco's talk'23

Gravitational waves

HEP high-multiplicity

$$pp \to t\bar{t} + 4\text{jets}$$



Spinner, Breso et al. 2405.14806

ML+SMEFT seems an overkill

This is not so *complex*

we understand the **structure** of the SMEFT effects

$$\mathcal{L} = \mathcal{L}_{SM} + \sum_i \frac{c_{6,i}}{\Lambda^2} \mathcal{O}^{d=6} + \sum_i \frac{c_{8,i}}{\Lambda^4} \mathcal{O}^{d=8} + \ldots$$

it is an expansion, which should be well-controlled e.g,

$$f(\vec{x}, \vec{c}) = \frac{1}{\sigma_{fid}(\vec{c})} \frac{d\sigma(\vec{x}, \vec{c})}{d\vec{x}}$$

where x= (pT, mtt, costheta…) and c=EFT coefficients

$$f(\vec{x}, \vec{c}) = f(\vec{x}, 0) + \sum_i^{n_{eft}} f^{(i)}(\vec{x})\, c_i + \sum_{j,k}^{n_{eft}} f^{j,k)}(\vec{x})\, c_j c_k + \ldots$$

plus we have excellent simulation tools

BUT…

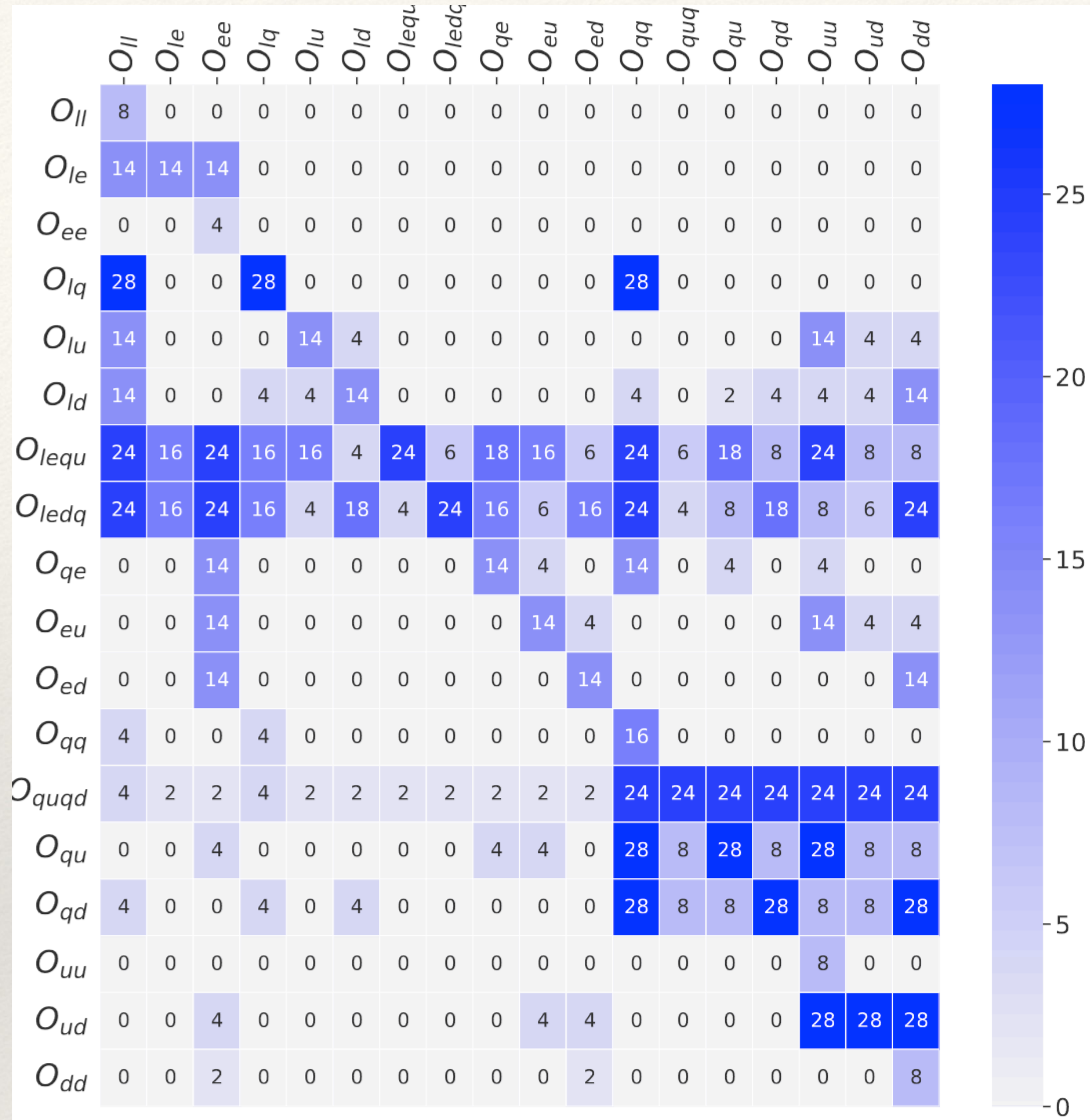
UV models populate the SMEFT

Model	C_{HD}	C_{ll}	C_{Hl}^3	C_{Hl}^1	C_{He}	$C_{H\Box}$	$C_{\tau H}$	C_{tH}	C_{bH}
S						$-\frac{1}{2}$			
S_1		1							
Σ			$\frac{1}{16}$	$\frac{3}{16}$			$\frac{y_\tau}{4}$		
Σ_1			$-\frac{1}{16}$	$-\frac{3}{16}$			$\frac{y_\tau}{8}$		
N			$-\frac{1}{4}$	$\frac{1}{4}$					
E			$-\frac{1}{4}$	$-\frac{1}{4}$			$\frac{y_\tau}{2}$		
Δ_1					$\frac{1}{2}$		$\frac{y_\tau}{2}$		
Δ_3					$-\frac{1}{2}$		$\frac{y_\tau}{2}$		
B_1	1					$-\frac{1}{2}$	$-\frac{y_\tau}{2}$	$-\frac{y_t}{2}$	$-\frac{y_b}{2}$
Ξ	-2					$\frac{1}{2}$	y_τ	y_t	y_b
W_1	$-\frac{1}{4}$					$-\frac{1}{8}$	$-\frac{y_\tau}{8}$	$-\frac{y_t}{8}$	$-\frac{y_b}{8}$
φ							$-y_\tau$	$-y_t$	$-y_b$
$\{B, B_1\}$						$-\frac{3}{2}$	$-\ y_\tau$	$-\ y_t$	$-\ y_b$
$\{Q_1, Q_7\}$								y_t	
Model	C_{Hq}^3	C_{Hq}^1	$(C_{Hq}^3)_{33}$	$(C_{Hq}^1)_{33}$	C_{Hu}	C_{Hd}	C_{tH}	C_{bH}	
U	$-\frac{1}{4}$	$\frac{1}{4}$	$-\frac{1}{4}$	$\frac{1}{4}$			$\frac{y_t}{2}$		
D	$-\frac{1}{4}$	$-\frac{1}{4}$	$-\frac{1}{4}$	$-\frac{1}{4}$				$\frac{y_b}{2}$	
Q_5						$-\frac{1}{2}$		$\frac{y_b}{2}$	
Q_7					$\frac{1}{2}$		$\frac{y_t}{2}$		
T_1	$-\frac{1}{16}$	$-\frac{3}{16}$	$-\frac{1}{16}$	$-\frac{3}{16}$			$\frac{y_t}{4}$	$\frac{y_b}{8}$	
T_2	$-\frac{1}{16}$	$\frac{3}{16}$	$-\frac{1}{16}$	$\frac{3}{16}$			$\frac{y_t}{8}$	$\frac{y_b}{4}$	
T			$-\frac{1}{2}\frac{M_T^2}{v^2}$	$\frac{1}{2}\frac{M_T^2}{v^2}$			$y_t\frac{M_T^2}{v^2}$		

Granada dictionary

Single-particle extensions

Ellis, Madigan, Mimasu, VS, You
2012.02779, JHEP



Loop UV models with DM

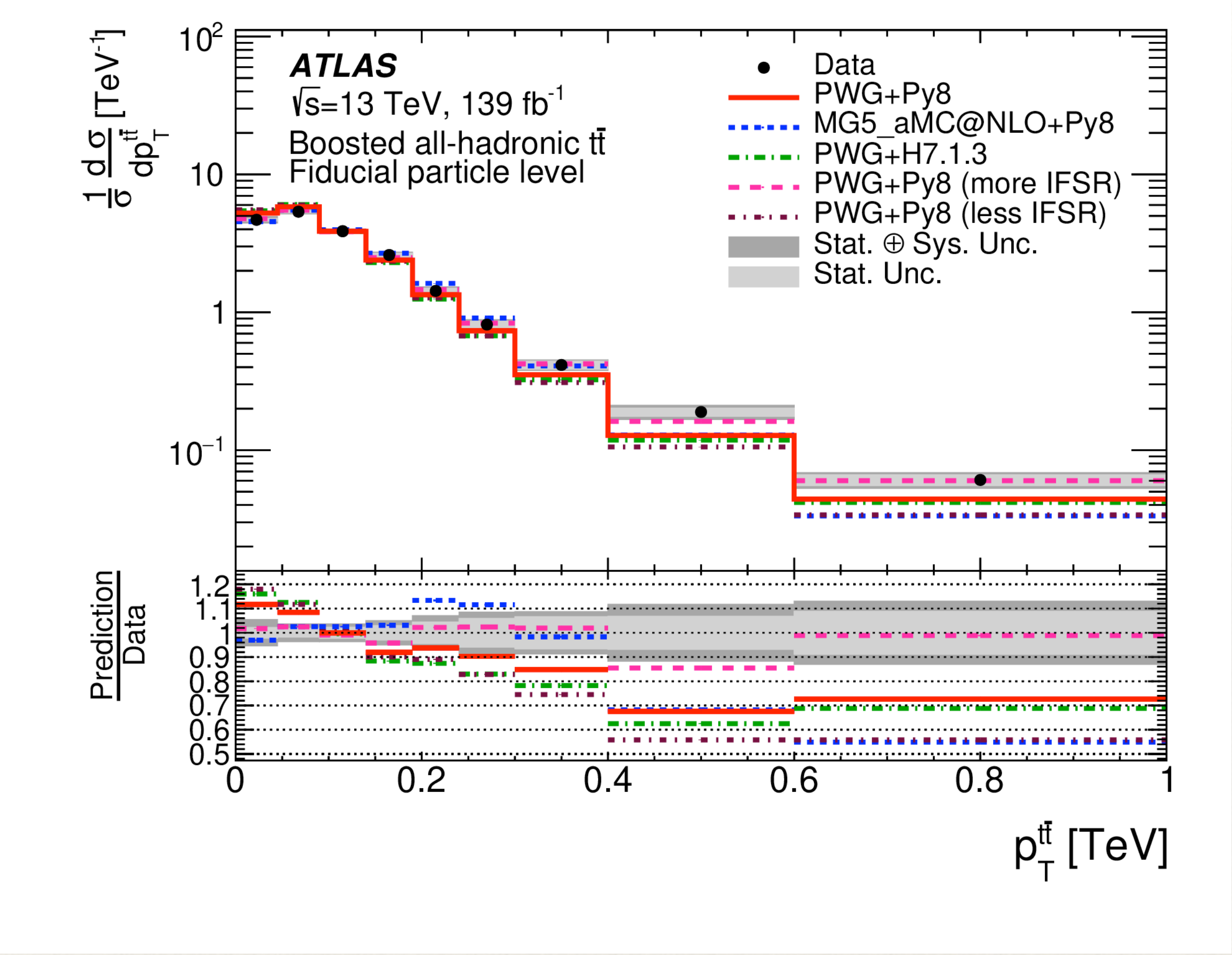
Cepedello, Esser, Hirsch, VS,
2302.03485, JHEP

see also talks by *Loise* (FN) and *Wilsch* (MSSM)

EFT fits are truly multidimensional

If we want to reach highest precision, we have to face some challenges

1. Models do push towards multidimensionality in terms of parameters



2. Experimentally, the LHC has a large kinematic regime, getting larger as we gather more statistics (more info in observables)

3. Things we have neglected so far, e.g. systematics / correlations… will have to be dealt with

So far, traditional searches for minima worked,
but we will hit the **curse of dimensionality**
and that is where ML is key

An example of using ML to optimise global fits

Gomez-Ambrosio, ter Hoeve, Madigan, Rojo, VS,
JHEP (2023)

Unbinned analysis

In the SMEFT global fits we use inclusive and differential distributions

But, instead of binning the data (Nev in Nbins)
one can do an unbinned analysis

$$\mathcal{D} = \{\boldsymbol{x}_i\} \qquad \boldsymbol{x}_i = (x_{i,1}, x_{i,2}, \ldots, x_{i,n_k}) \ , \qquad i = 1, \ldots, N_{\rm ev} \ ,$$

where **x_**i is a vector with kinematic features of event i

we can build a likelihood from the binned or unbinned dataset in terms of EFT coefficients

$$\mathcal{L}(\boldsymbol{c}) = P\left(\mathcal{D}|\mathcal{T}(\boldsymbol{c})\right) \ .$$

We use the fact that SMEFT is an expansion, use the parametrisation

$$r_\sigma(\boldsymbol{x}, \boldsymbol{c}) \equiv \frac{f_\sigma(\boldsymbol{x}, \boldsymbol{c})}{f_\sigma(\boldsymbol{x}, \boldsymbol{0})} = 1 + \sum_{j=1}^{n_{\rm eft}} r_\sigma^{(j)}(\boldsymbol{x}) c_j + \sum_{j=1}^{n_{\rm eft}} \sum_{k \geq j}^{n_{\rm eft}} r_\sigma^{(j,k)}(\boldsymbol{x}) c_j c_k \ ,$$

Optimisation with ML

1. Generate training datasets, with labels **c**, separated between linear and quadratic contributions

2. We separately train NNs with a binary cross-entropy output (=a value from 0 to 1)

$$\mathcal{D}_{\rm eft}(\boldsymbol{c} = (0, \ldots, 0, c_j^{(\rm tr)}, 0, \ldots, 0))$$

$$r_\sigma(\boldsymbol{x}, c_j^{(\rm tr)}) = 1 + c_j^{(\rm tr)} {\rm NN}^{(j)}(\boldsymbol{x})$$

$$\mathcal{D}_{\rm eft}(\boldsymbol{c} = (0, \ldots, 0, c_j^{(\rm tr)}, 0, \ldots, 0, c_k^{(\rm tr)}, 0, \ldots))$$

$$r_\sigma(\boldsymbol{x}, c_j^{(\rm tr)}, c_k^{(\rm tr)}) = 1 + c_j^{(\rm tr)} c_k^{(\rm tr)} {\rm NN}^{(j,k)}(\boldsymbol{x})$$

After training, we have

$$\{\mathrm{NN}^{(j)}(\boldsymbol{x})\} \quad \text{and} \quad \{\mathrm{NN}^{(j,k)}(\boldsymbol{x})\}\,, \quad j,k = 1,\ldots,n_{\mathrm{eft}}\,, \quad k \geq j\,.$$

with scaling with number of training EFT coefficients

$$n_{\mathrm{eft}} \quad + \quad n_{\mathrm{eft}}(n_{\mathrm{eft}}+1)/2 \quad \xrightarrow{\text{parallel}} \quad n_{\mathrm{eft}}^2/n_{\mathrm{proc}}$$

NN approximates the true ratio r

$$\hat{r}_\sigma(\boldsymbol{x},\boldsymbol{c}) = 1 + \sum_{j=1}^{n_{\mathrm{eft}}} \mathrm{NN}^{(j)}(\boldsymbol{x})c_j + \sum_{j=1}^{n_{\mathrm{eft}}}\sum_{k\geq j}^{n_{\mathrm{eft}}} \mathrm{NN}^{(j,k)}(\boldsymbol{x})c_jc_k\,,$$

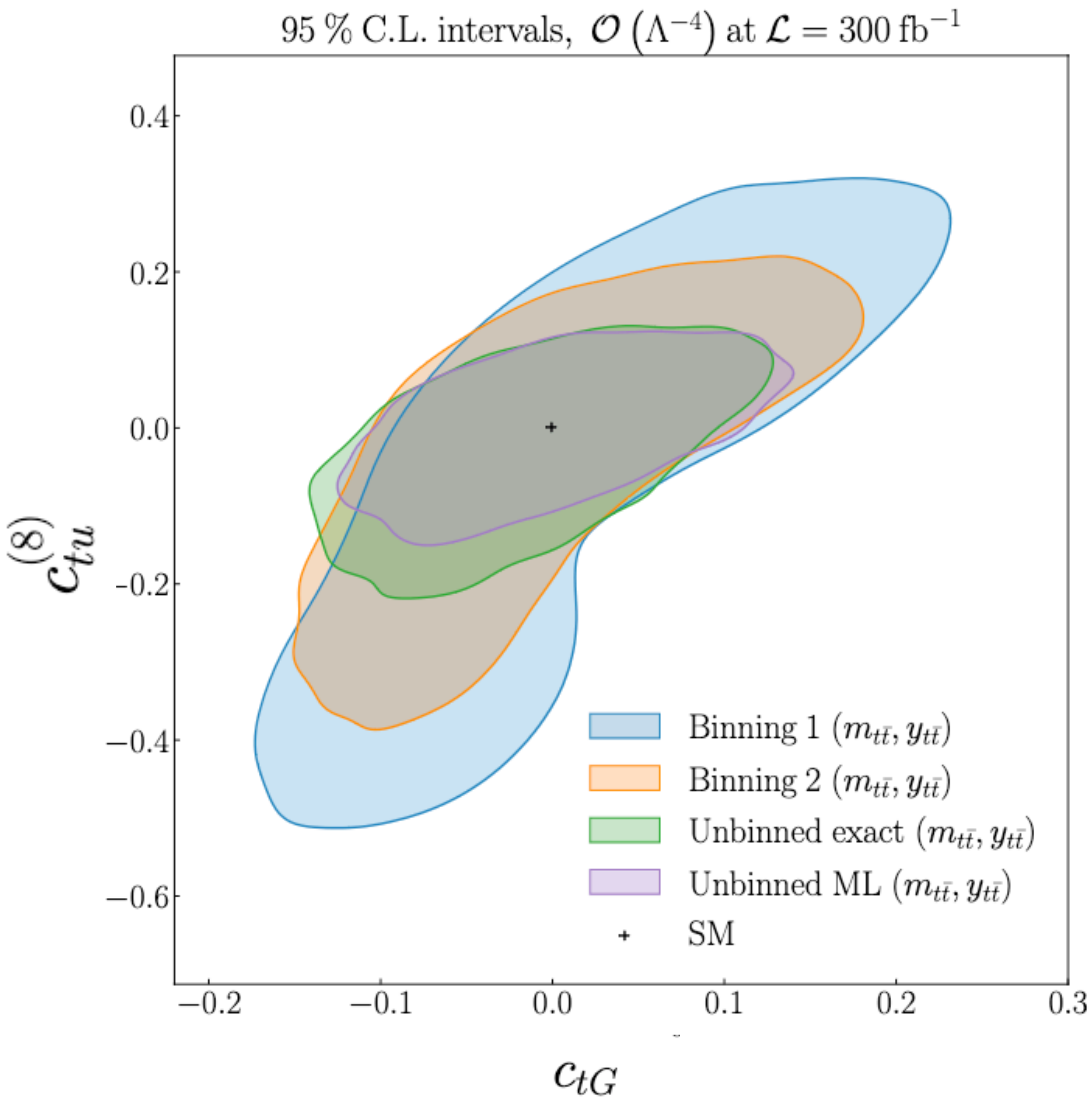
which we can use to build a LLR as a function of **c** and obtain limits

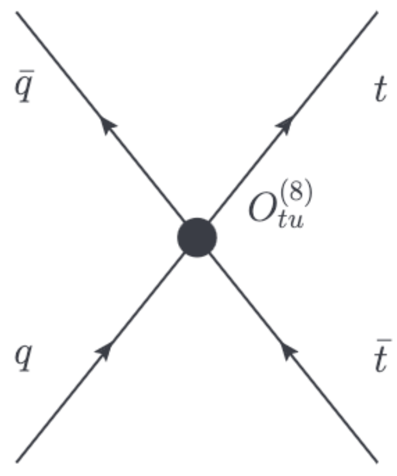
finally, to estimate the ML training error, we build replicas

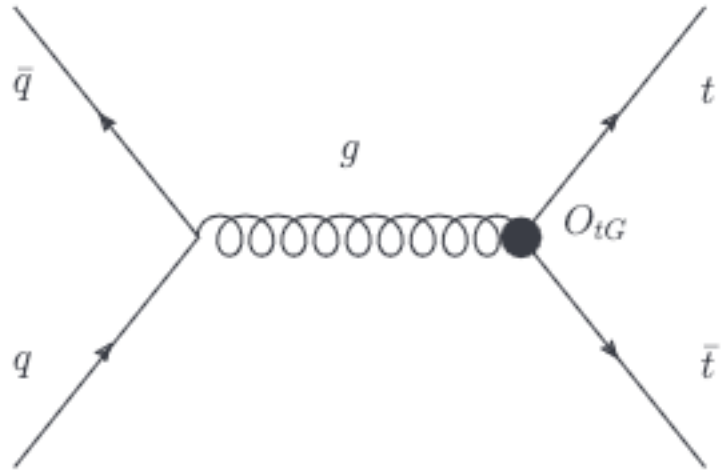
$$\hat{r}_\sigma^{(i)}(\boldsymbol{x},\boldsymbol{c}) \equiv 1 + \sum_{j=1}^{n_{\mathrm{eft}}} \mathrm{NN}_i^{(j)}(\boldsymbol{x})c_j + \sum_{j=1}^{n_{\mathrm{eft}}}\sum_{k\geq j}^{n_{\mathrm{eft}}} \mathrm{NN}_i^{(j,k)}(\boldsymbol{x})c_jc_k\,, \qquad i = 1,\ldots,N_{\mathrm{rep}}$$

Example ttbar

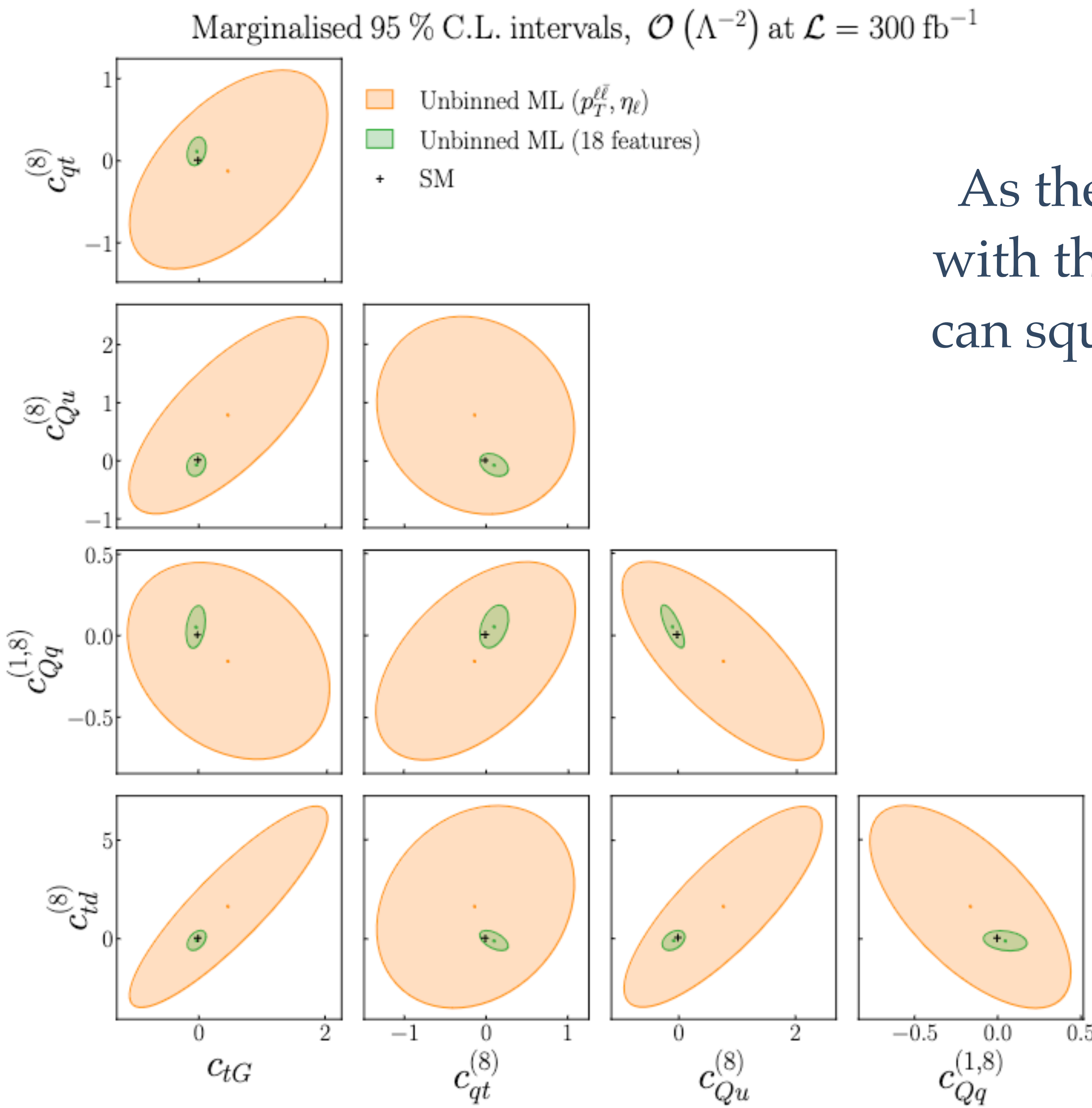
Comparison binned / unbinned analysis







Example ttbar



As the NN training scales well with the number of features, one can squeeze in more information

here example of fully leptonic ttbar

Conclusions

The state-of-the-art in our understanding of SMEFT:
global analyses including hundreds of observations
and dozens of possible EFT deviations

So far, global fits are done using *traditional* methods
but with more data and higher expectations
we have to step up our game

Here, PoC using ML to optimise EFT fits, parallelisation:
scaling goes as nEFT^2/nProcessors
so can handle increasing EFT parameters

We train NNs to approximate binned and unbinned
likelihood ratios, unbinned keeps more information

NNs are good at handling large feature space,
can squeeze more information with little increase in computational cost

To the skeptics: ML is not a hammer searching for a nail,
it is yet another tool we can & should learn to use