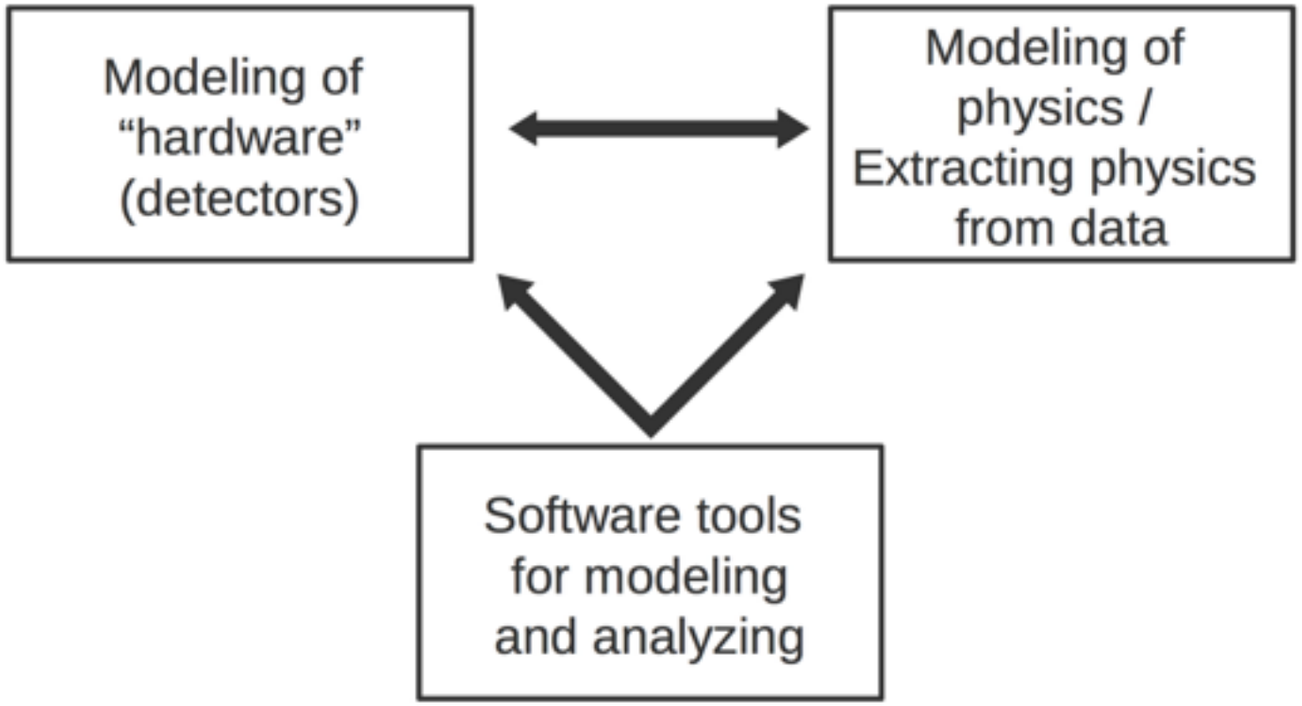
Summary of Track 2
Data Analysis, Algorithms and Tools

(L. Moneta, G. Watts)

Conveners: *L. Moneta, G. Watts, W. Brooks*
Advisors: *D. Damazio, A. Kryukov*

Outline

- Data Analysis - Algorithms and tools
  - bringing together different disciplines
    - Methods for data analysis, reconstruction and simulation
    - Software and computer tools



Some Statistics

- 28 abstracts submitted to the conference
  - 18 oral presentations
  - 10 posters

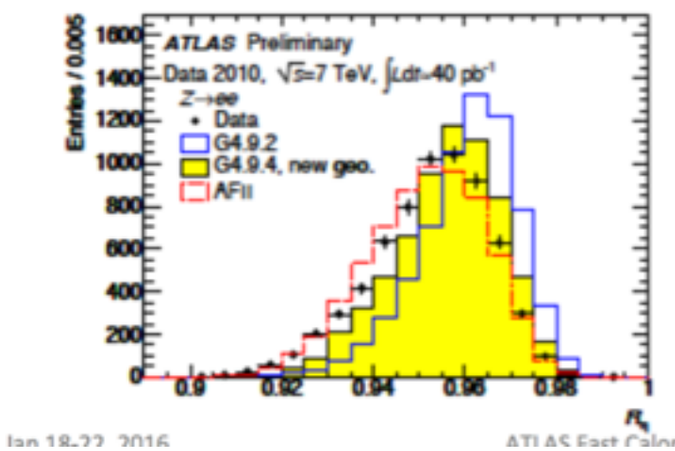
Track 2 Presentations

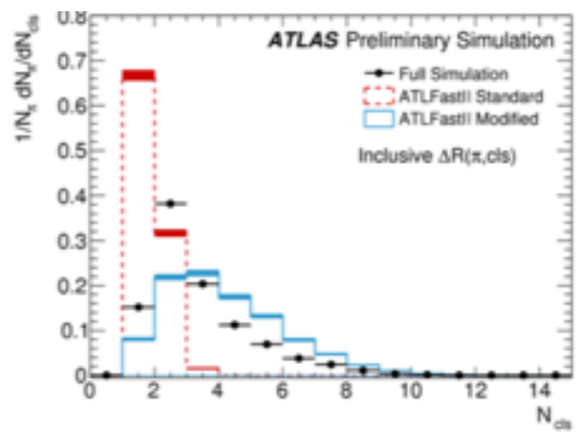
- **Methods for simulations:**
  - Fast simulation (calorimeter) in ATLAS
  - Delphi's 3 (Very fast simulation)
- **Reconstruction and trigger methods**
  - Neural e.m. ATLAS Calorimeter trigger
  - New method for event reconstruction in Liquid Argon TPC (Dune)
  - Novel real-time alignment and calibration for LHCb
  - Parallel 4D track finder
  - New Vertexing algorithm
- **Analysis methods** (many new and innovative methods, majority based on machine learning)
  - Approximate profile likelihood ratios using ML
  - Matrix element
  - Density Estimation trees
  - Re-weight distributions using Boosted Decision Trees
  - Deconvolution
  - New SVM and Generalisation
- **Software tools**
  - New ML tools in ROOT
  - Fitting and statistical studies on GPU
  - ROOT Notebooks and ROOT as a Service

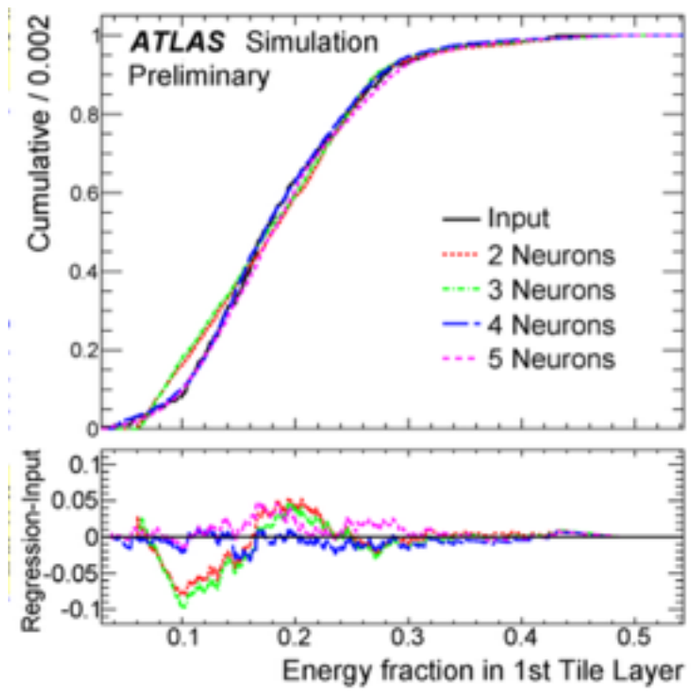
» **And also some very interesting posters**

Upgrading ATLAS Fast Calorimeter Simulation
(Z. Hubacek)

- New fast simulation for ATLAS calorimeter
  - current simulation not describing well hadronic objects
  - make a new parametrisation using NN



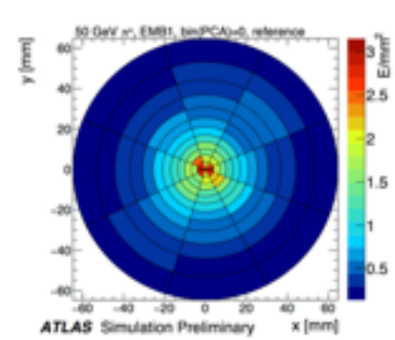


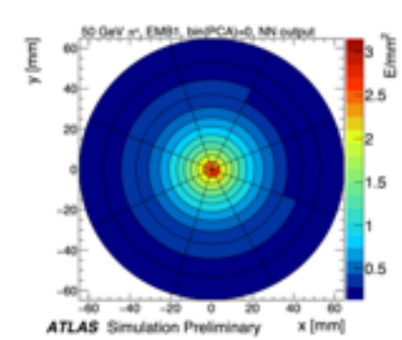


Shower Shape

- Re-simulate single particles (e, γ, π) on a fine energy and η grid (approx 15x100 points)
  - Using full Geant4 simulation, current ATLAS geometry
  - Save detailed spatial information (x,y,z,t,E) of the developed shower
- Re-derive the energy parameterization (longitudinal) and shower shape (lateral)
  - Reduce the amount of information to a compact form
  - Use TMVA Regression to approximate histograms

- The binned energy fractions are fit with a Neural Network
- Store TMVA weights instead of histograms

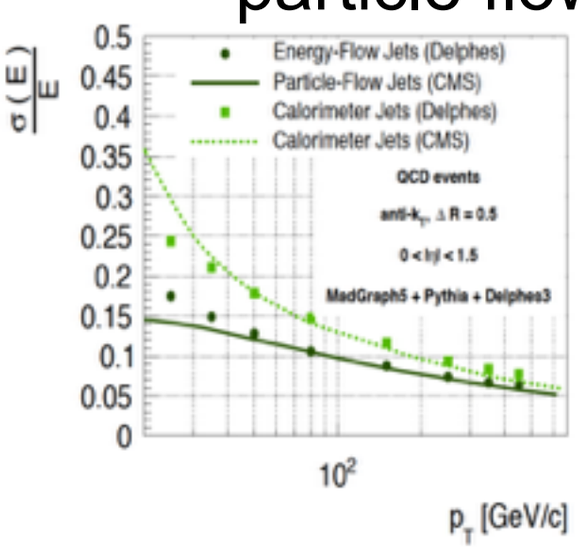


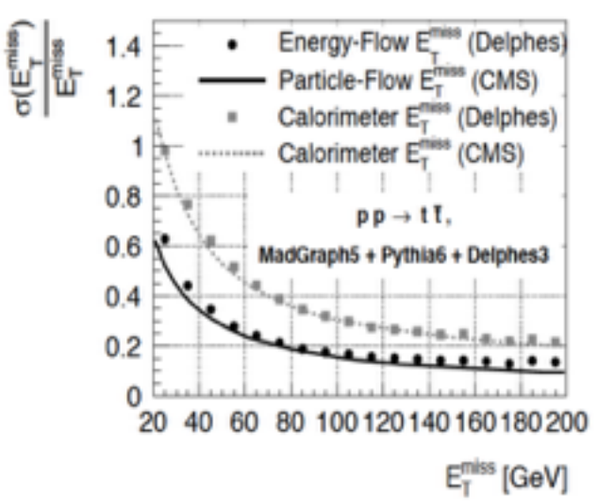


Delphes 3 (*M. Selvaggi*)

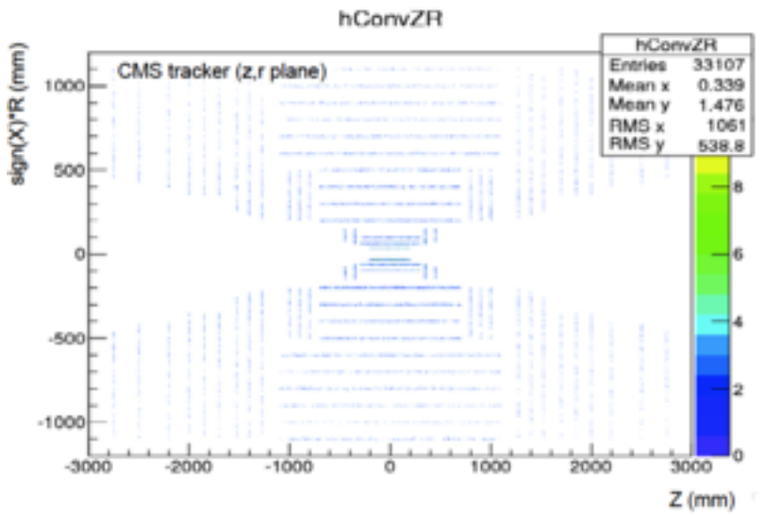
- Latest developments in Delphes 3
  - Generic Parametric simulation, parametric the detector response
    - Full simulation (GEANT) ~ 100 s/evt
    - Experiment fast simulation ~ 1s/evt
    - Delphes ~ 10 ms/evt

      → quick phenomenological studies
      → sensitive to acceptance and complex observable (Jets,MET)
      → scan big parameter space (SUSY-like)
      → preliminary tests of new geometries/resolutions (jet substructure)
      → educational purpose (bachelor/master thesis)
    - New developments:
      - particle flow





Photon conversions

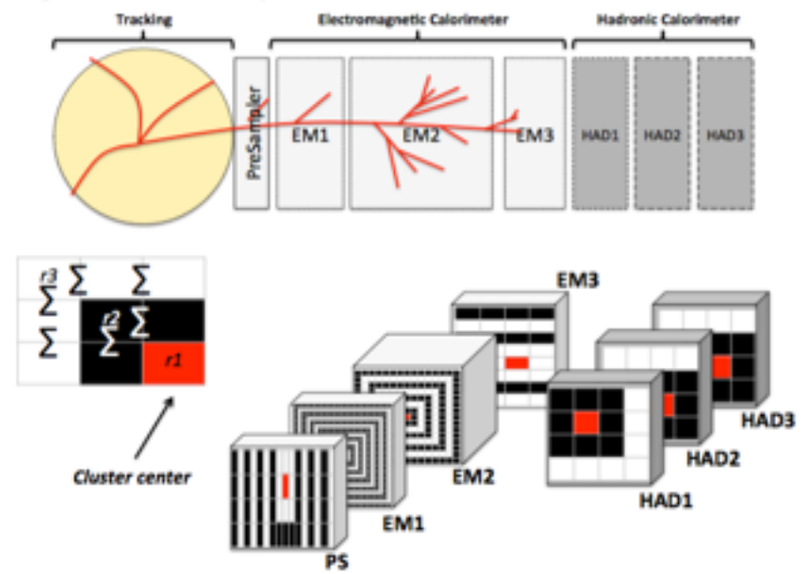


Neural e-Gamma trigger in ATLAS (*J.V. da Fonseca Pinto*)

- Calorimeter Ringer algorithm based on a Neural Network

Calorimeter Ring concept

Algorithm skept:



Total number of Rings per layer (covering 0.4 x 0.4 region in η x φ)						
PS	EM1	EM2	EM3	HAD1	HAD2	HAD3
8	64	8	8	4	4	4

- Build using all calorimeter layers, centered in a window (0.4 x 0.4 in η x ϕ) at the hottest cell of each layer;
- Hottest cell: first "ring" (on each layer);
- Next ring: collection of cells around the previous one;
- The ring "value" is the sum of the E_T of all cells composing the ring;
- Provides input data reduction for the neural processing (w.r.t using all cells);
- Keeps the physics interpretation (typical EM object shower shape).

Benchmark: L2Calo@e24_lhmedium_L1EM20V

Medium (P_D)

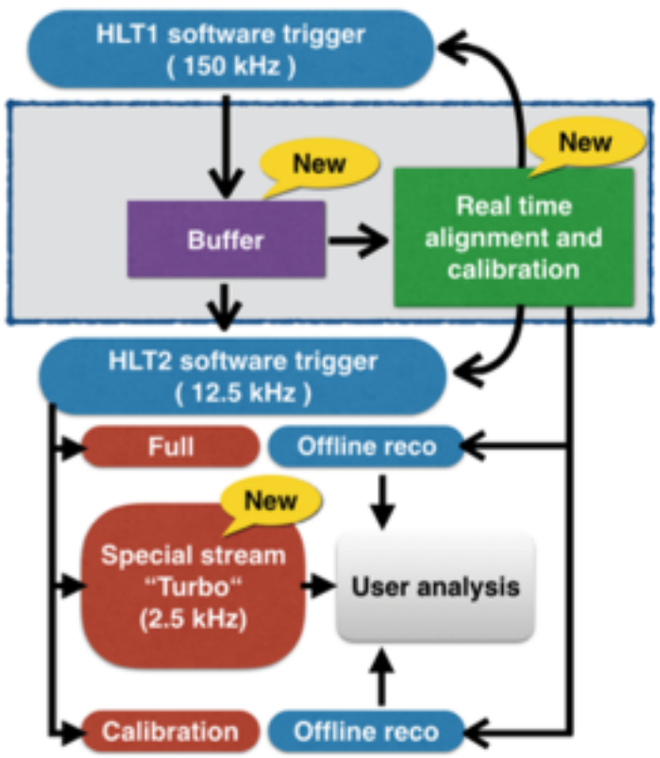
T&P efficiency

	T&P PD%	FR%
Ringer	97.78	5.37
L2Calo	97.66	12.73

- Superior performance as compared to cut-based algorithm:
  - Factor of ~2 reduction of the fake rate.

Novel Real-time alignment and calibration and track reconstruction for the upgrade at LHCb (*R. Quagliani*)

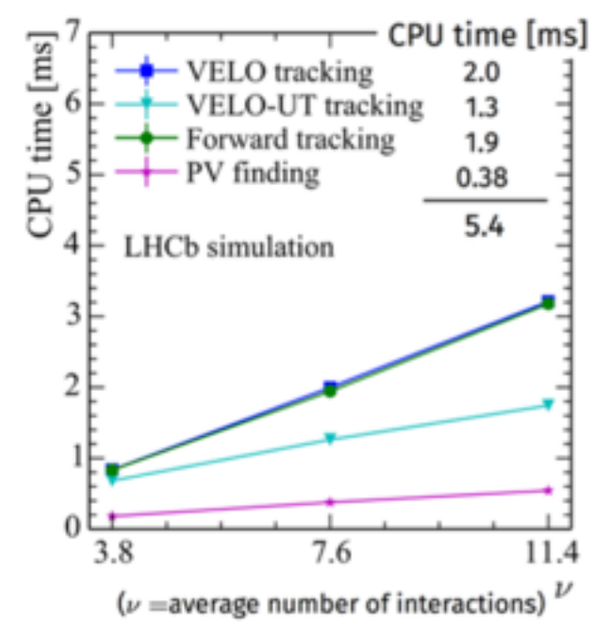
- Perform detector alignment and calibration in realtime to have best performance in High Level Trigger (HLT)
  - run automatically tracking alignment for each fill
    - vectorised and paralleled algorithms
  - new turbo stream for direct analysis



Need same reconstruction and performances achieved offline in HLT2

- Mandatory timing improvements to fit in timing budget.

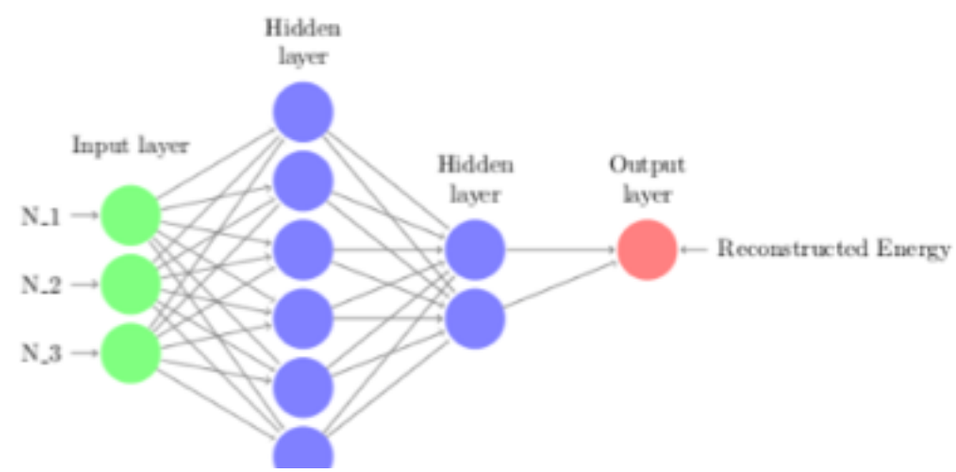
- LHCb Upgrade
  - fully software trigger
  - Tracking at 30 MHz
  - storing 2-5 Gb/s
  - 13 ms is CPU timing budget
    - used 5.4 ms for reco

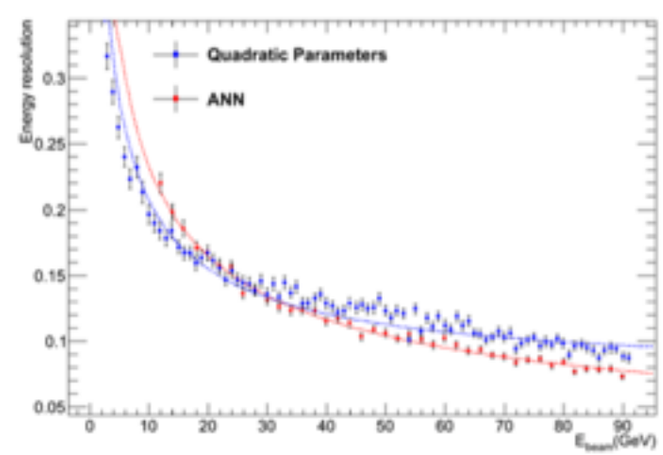


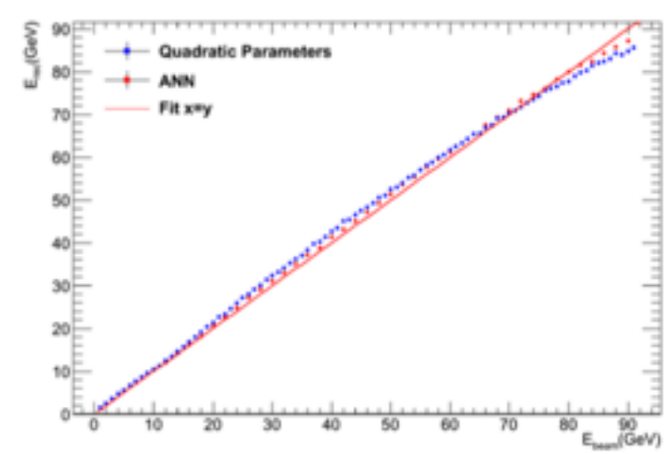
Energy reconstruction study in a High Granularity Hadronic Calorimeter for ILC (*S. Mannai*)

- Energy reconstruction using an Artificial Neural Network (Regression)
- Muon/electorn identification using ANN

  - TMultiLayerPerceptron of root package.
  - 2 hidden layers with 6 and 2 neurons.
  - The input variables: N_1,N_2,N_3 .
  - The output variable is the reconstructed energy: E_{rec} .
  - Monte Carlo Simulation
    - Training Samples: Odd energies, 1-99 GeV (50 training samples)
    - Test Samples: Even energies, 10-90 GeV (40 test samples)







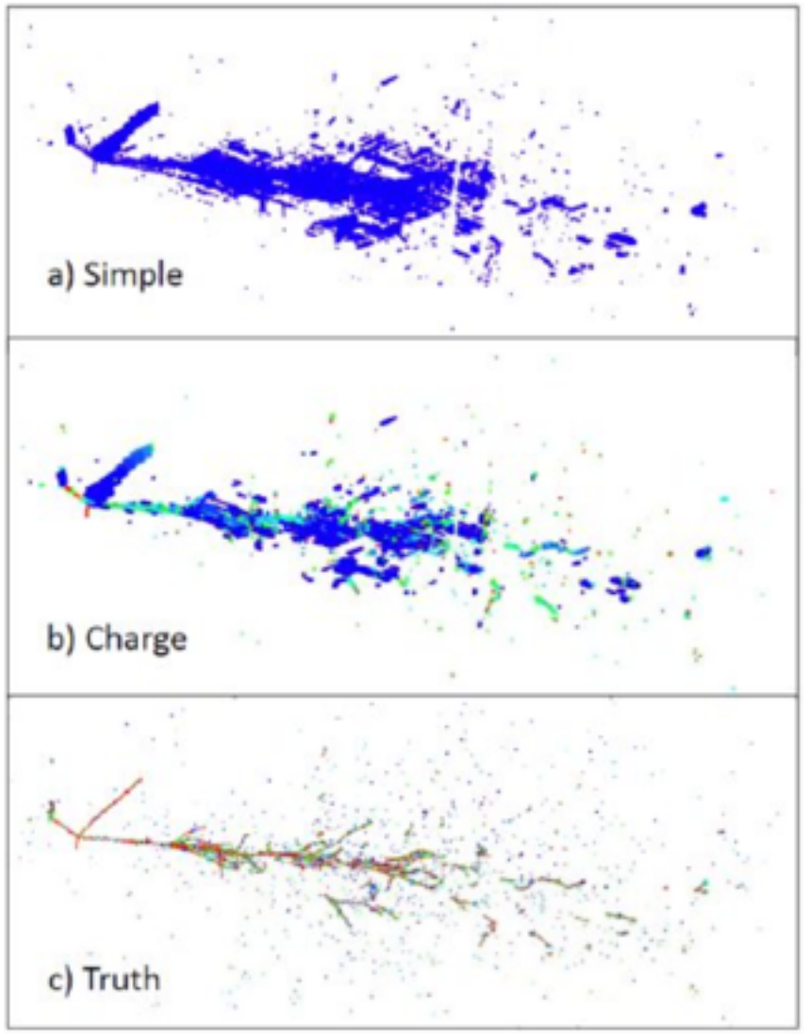
A novel method for event reconstruction in Liquid Argon Time Projection Chamber (*M. Potekhin*)

- Wire Cell reconstruction for neutrino experiments (DUNE)
  - New tomographic method
    - reconstruct a 3D model of the events

Wire Cell: the method

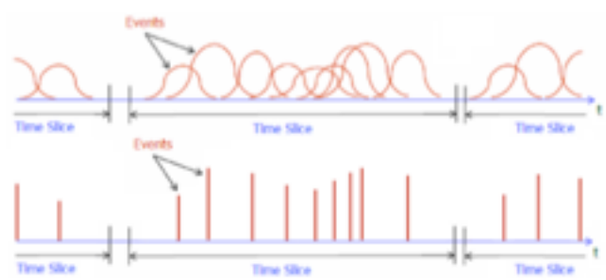
- The problem can then formulated as minimizing the residual $||w_{expected} - w_{observed}||$ over cells and wires.
- Assuming:
  - W_m is a vector merged wires (signals)
  - C_m is a vector of merged cells
  - G is the "geometry matrix" mapping merged cells to merged wires
  - V is covariance matrix (which can be approximated as a diagonal for now)

  the quantity to be minimized can be written as $x^2 = (W_m - G*C_m)^T V^{-1} (W_m - G*C_m)$
- The phase space being probed is the set of merged cells included in the computation.
- Secondary regularization is possible since one can introduce a penalty term in the x^2 by interrogating adjacent time bins (continuity hypothesis) - whether a merged cell is found in the vicinity of point in the next time bin or not. This approach must be tuned with heuristics since it works better for some hypotheses than another - for example, for particle tracks compared to EM showers.
- Markov Chain MC: eliminate cells, update matrices and recompute x^2. Compare to see if there is an improvement. Repeat. Solve for C.
- Combine the results of 2D imaging in each slice to obtain a 3D voxelized representation of the event.
- Do pattern recognition (e.g. track vs shower): more details below.

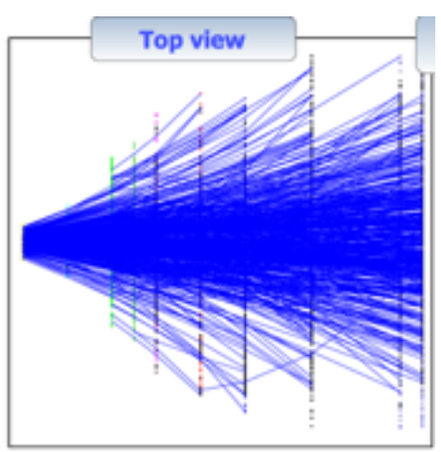


Parallel 4-Dimensional Cellular Track Finder for the CBM Experiment (*I. Kisel*)

- Reconstruction challenge in CBM at FAIR/GSI
- Cellular Automaton Tack Finder
  - Intrinsically parallel algorithm
  - extremely simple
  - very fast
  - 4D (time-based) Track reconstruction
    - continous beam

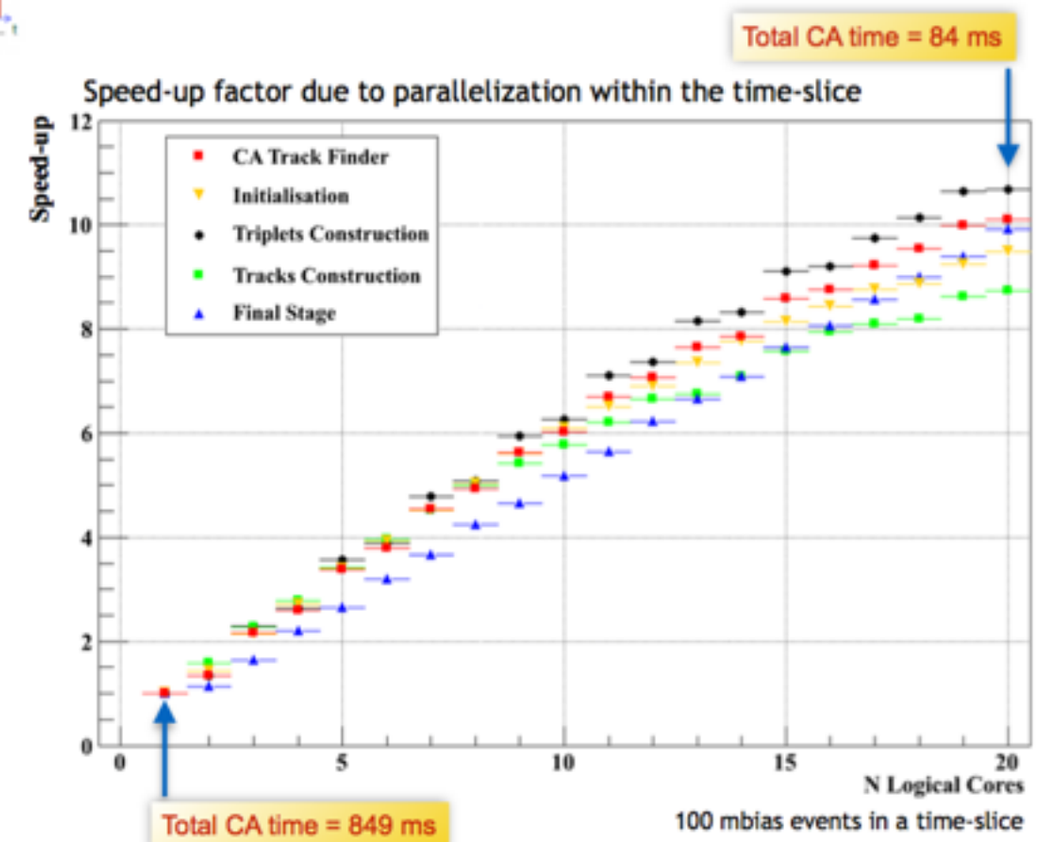


- The beam in the CBM will have no bunch structure, but continuous.
- Measurements in this case will be 4D (x, y, z, t).
- Significant overlapping of events in the detector system.
- Reconstruction of time slices rather than events is needed.



Stage of the algorithm	% of total execution time
Initialisation	8
Triplets construction	64
Tracks construction	15
Final cleaning	13

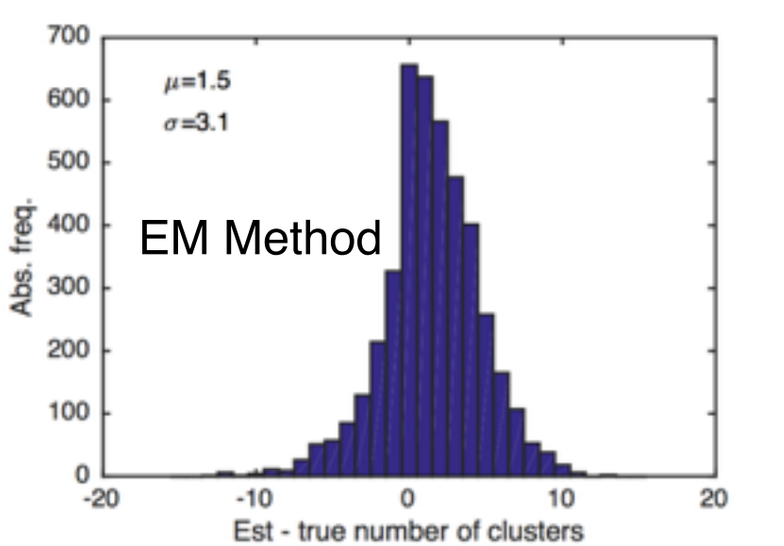
Efficiency, %	3D	3+1 D	4D
All tracks	83.8	80.4	83.0
Primary high-p	96.1	94.3	92.8
Primary low-p	79.8	76.2	83.1
Secondary high-p	76.6	65.1	73.2
Secondary low-p	40.9	34.9	36.8
Clone level	0.4	2.5	1.7
Ghost level	0.1	8.2	0.3
Time/event/core, ms	8.2	31.5	8.5

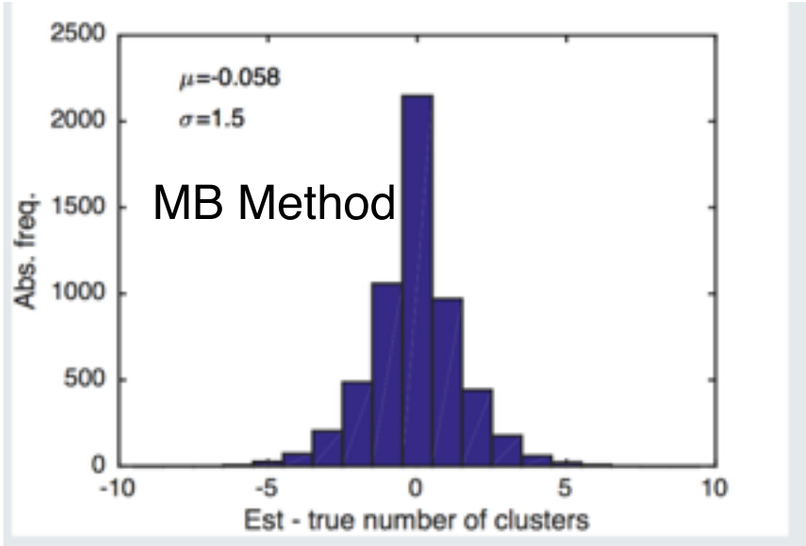


4D event building is scalable with the speed-up factor of 10.1; 3D reconstruction time 8.2 ms/event is recovered in 4D case

Vertex finding by sparse model-based clustering (*R. Fruhwirth*)

- Comparison of Sparse model-based clustering vs EM algorithm (iterative Max. Likelihood) for primary vertex finding
- Model-based clustering
  - Bayesian method including prior densities for number of clusters, size and cluster spread.
  - Based on a normal model
  - Using Markov-Chain MC for sampling posterior





Good Results
but clustering is slow
due to MCMC sampling

The matrix element method at LHC (*S. Wertz*)

- Reconstruct event probability density

Sample likelihood
→ M.L. parameter fit
$\prod_{i \in events} P(\mathbf{x}_i|\alpha)$

Neyman-Pearson discriminant [4]
→ Hypothesis testing/search for rare process
$P(\mathbf{x}|S)/\sum_i r_i P(\mathbf{x}|B_i)$

<u>… Can be computed!</u>

$$P(\mathbf{x}|\alpha) = \frac{1}{A_\alpha \sigma_\alpha} \int \mathrm{d}\Phi(y) \frac{\mathrm{d}x_1 \mathrm{d}x_2}{x_1 x_2 s} f(x_1) f(x_2) |\mathcal{M}_\alpha(y, x_1, x_2)|^2 W(\mathbf{x}|y) \epsilon_\alpha(y)$$

Theoretical hypothesis (Matrix Element) + Parton shower + Detector (transfer functions, efficiencies) + Experimental information (whole event **x**)

→ New project: **Mo**dular **M**atrix **E**lement **M**ethod implement**ta**tion

<u>Goal</u>:

- **Modular** framework written in **C++**
- Changes of variables, transfer functions, interface to matrix elements, I/O, …: modules
- Provide default modules + easily add user-tailored modules

<u>Status</u>:

- Non-modular prototype validated ↔ MadWeight (next slides)
- Based on: **Cuba** (integration) [20], ROOT, **LHAPDF** [21]
- Modular skeleton in place & in active development – available on github
- New MadGraph C++ matrix element exporter → faster, easier use: under review

Approximating Likelihood Ratios using Machine Learning *(J. Pavez)*

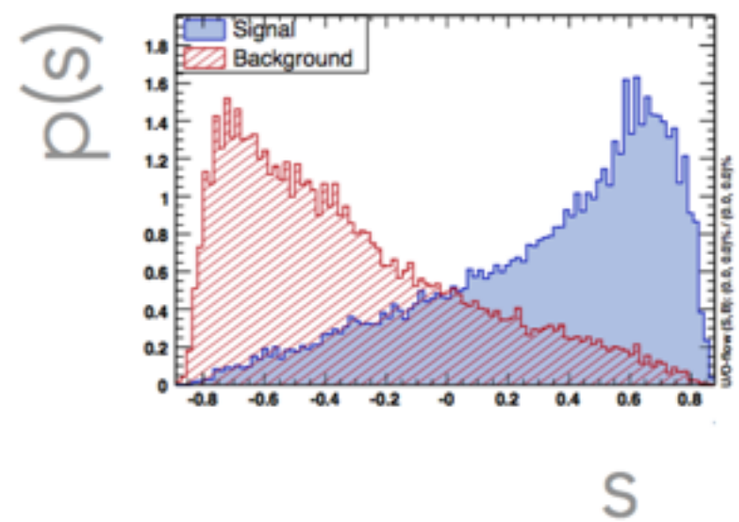
- New method to estimate likelihood ratio using ML

  - Then it can be proved that the **likelihood ratio**:

$$T(D) = \prod_{i=1}^{N} \frac{p(x_i \mid \theta_0)}{p(x_i \mid \theta_1)}$$

  - Is **equivalent** to the ratio:

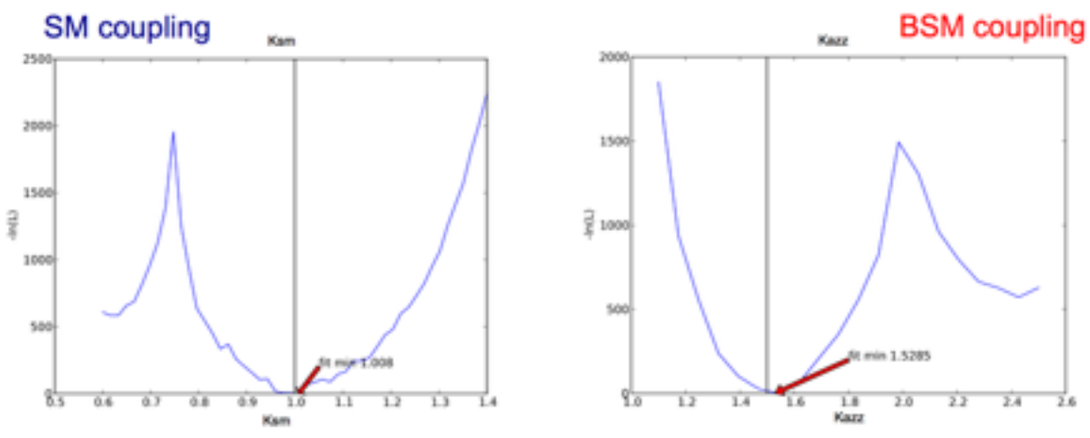
$$T(D) = \prod_{i=1}^{N} \frac{p(s(x;\theta_0,\theta_1) \mid \theta_0)}{p(s(x;\theta_0,\theta_1) \mid \theta_1)}$$



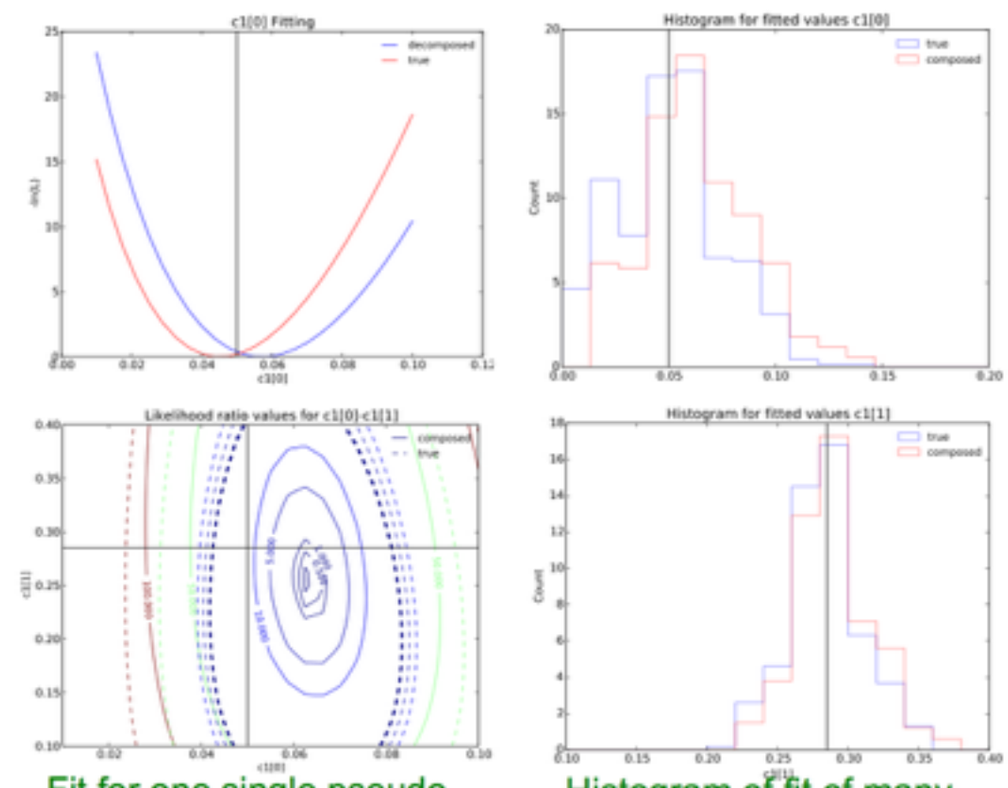
- Decomposition of likelihood ratio in case of several components (e.g. signal and background components)
  - decompose LR into pairwise comparisons
  - training with any signal and backgrounds
    - e.g. SM signal + backgrounds vs BSM + backgrounds
- Applications: Maximum Likelihood fit to signal + background contributions using the approximated likelihood

Approximating Likelihood Ratios using Machine Learning *(J. Pavez)*

- Python package available in github
  - e.g. like wrapping TMVA (and other ML Python packages) in RooFit
- Planning Integration with common tools (e.g. RooFit/RooStats)
- Need to understand better how errors affect the approximation
- Example applications:
  - Higgs coupling parameters



- Signal plus background fits

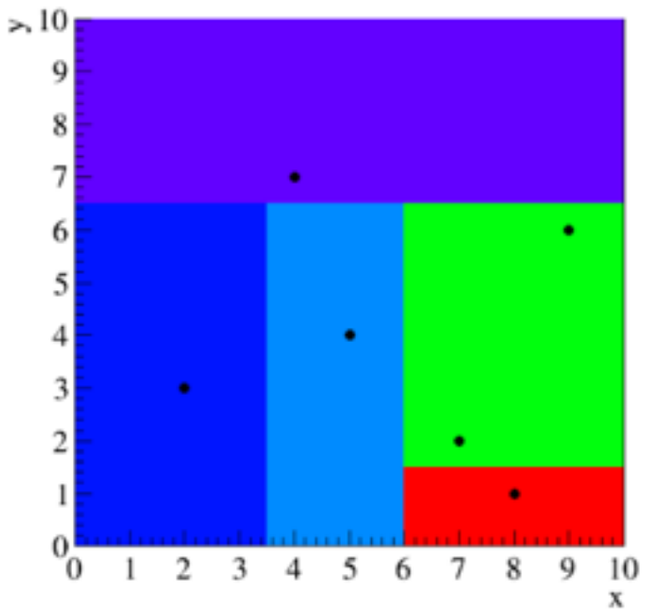


Fit for one single pseudo-experiment for:
A) Signal contribution.
B) Signal and Bkg.

Histogram of fit of many pseudo-experiments for:
A) Signal contribution.
B) Bkg contribution.

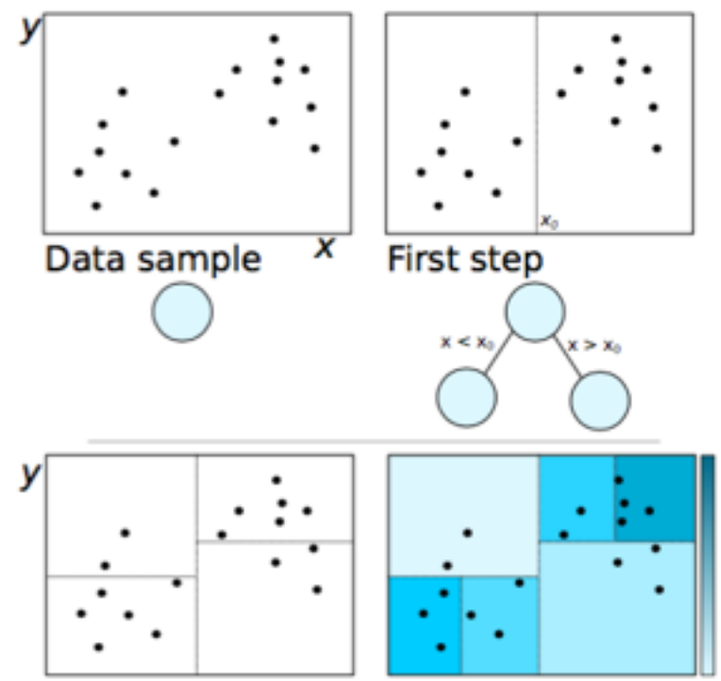
Density Estimation Trees (*L. Anderlini*)

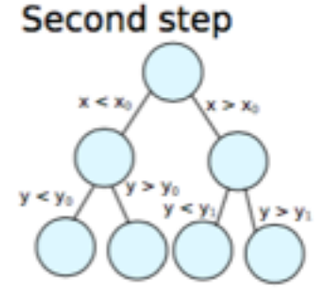
- A fast non-parametric method for density estimation



Density Estimation Tree

Density Estimator;
Focused on volumes, not data points;
Useful for fast modelling;
Low-density branches grow less;
Not useful for searches.





$$\hat{f}(x,y) = \frac{N_{\text{bin}}(x,y)}{V(x,y)\ N_{\text{tot}}}$$

- Applications:
  - sampling (fast MC simulation)
  - Efficiency table

Using Density Estimation Trees

Train DET T_{ALL} with uncut calib. sample;
Train DET T_{CUT} with cut calib. sample;
Create efficiency table as

$$T_\varepsilon = T_{\text{CUT}}/T_{\text{ALL}}$$

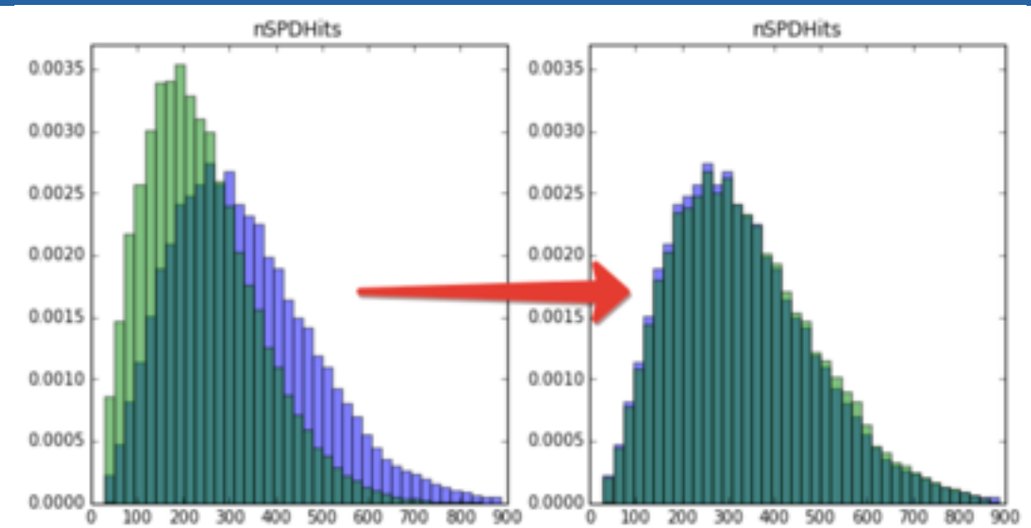
Eval the tree T_ε in $\vec{x}$ and read $\varepsilon(\vec{x})$.

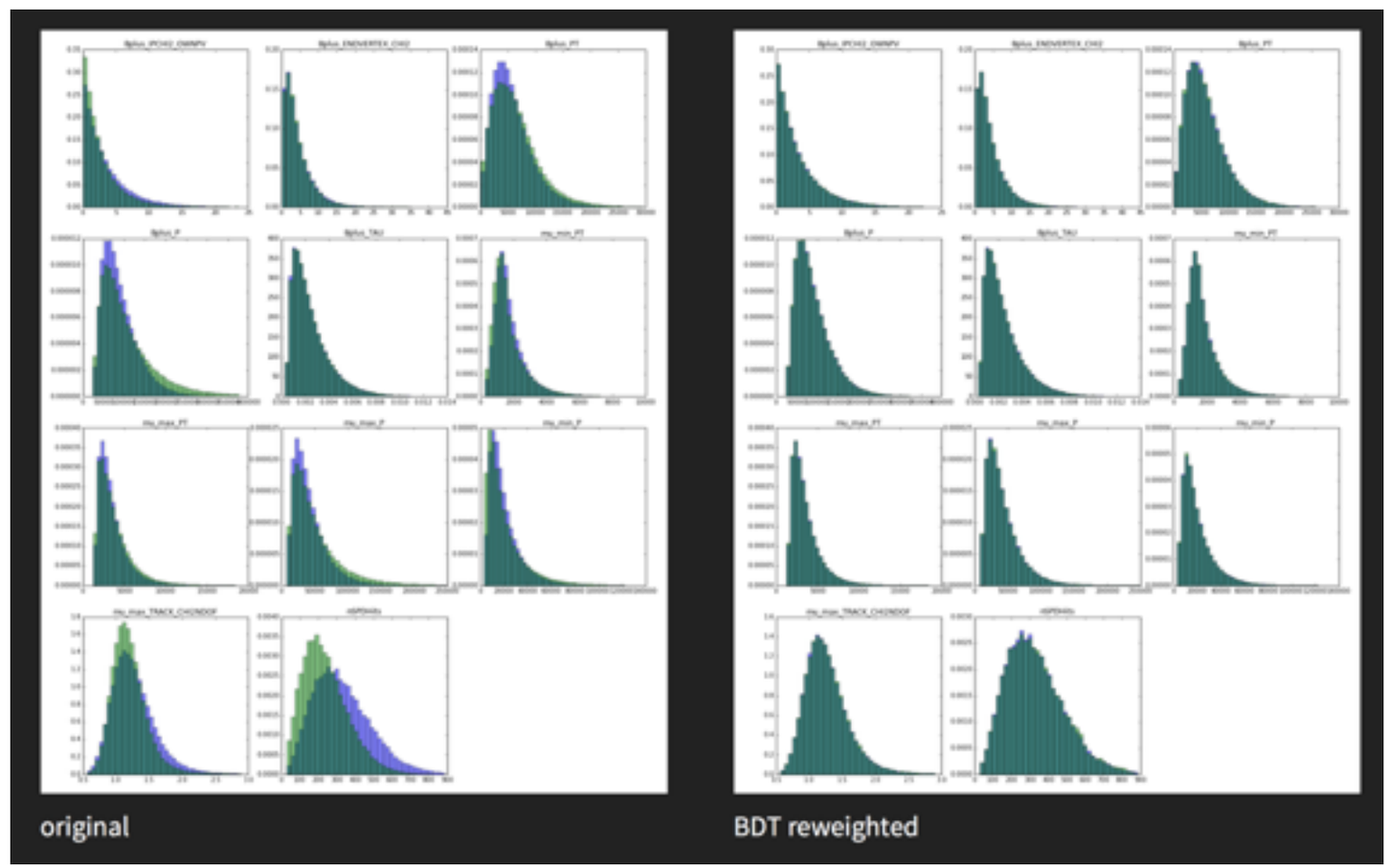
Reweighing Distributions with boosted decision trees (*A. Rogozhnikov*)

- Use ML algorithm to solve re-weighting problem
- Split by maximizing symmetrical chi-square

$$\chi^2 = \sum_{\text{bin}} \frac{(w_{\text{bin, original}} - w_{\text{bin, target}})^2}{w_{\text{bin, original}} + w_{\text{bin, target}}}$$

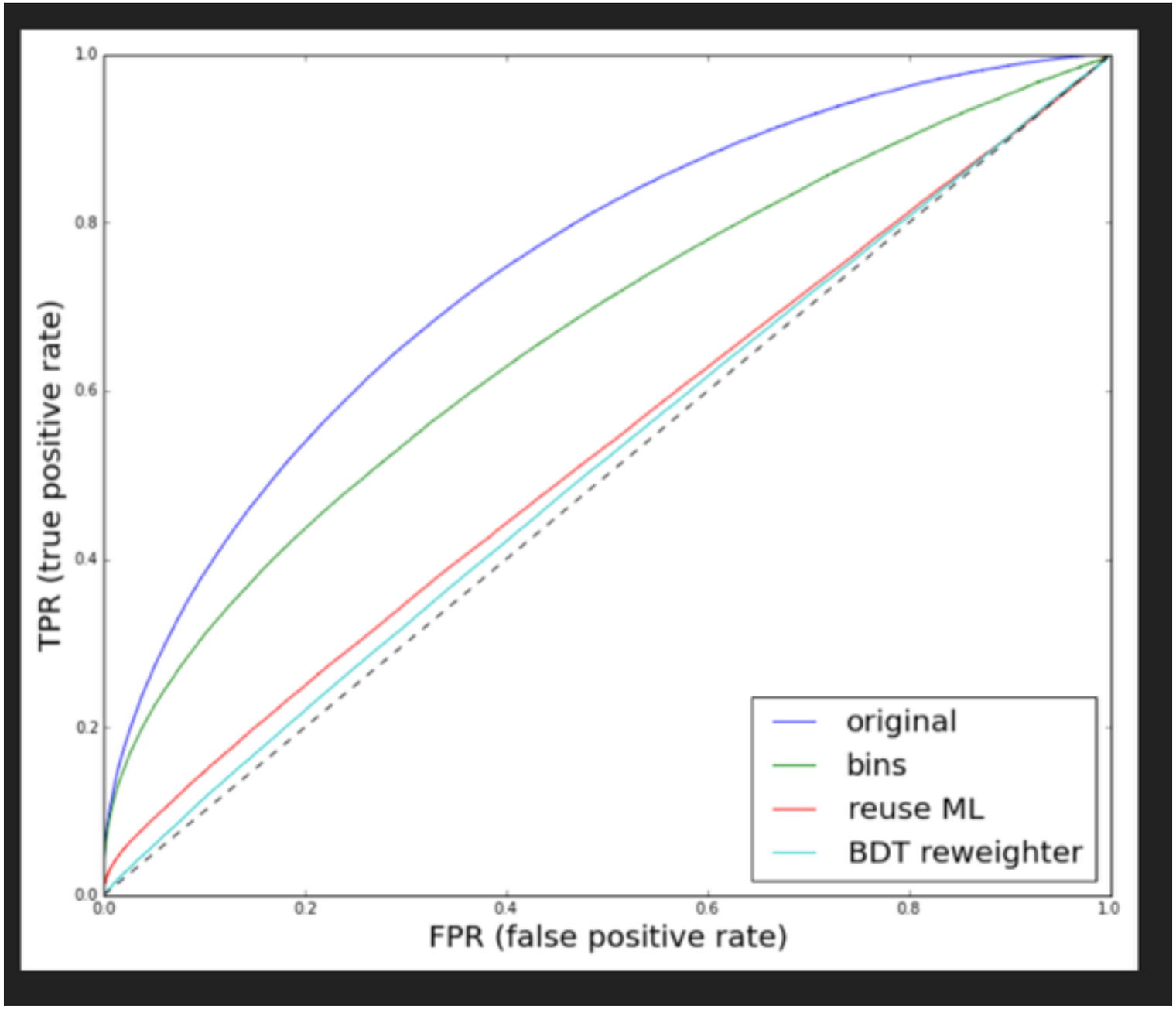
- Can work for the multi-dimensional case





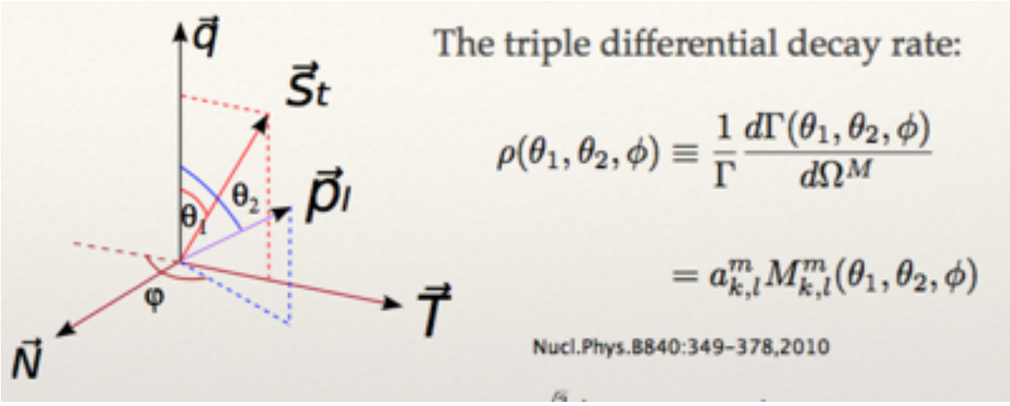
Reweighing Distributions with boosted decision trees (*A. Rogozhnikov*)

- Use also a ML method for multi-dimensional comparison of distributions (goodness of fit tests)
  - ROC curve for judging quality



Deconvolving the detector from an observed signal in Fourier space (*J. Boudreau*)

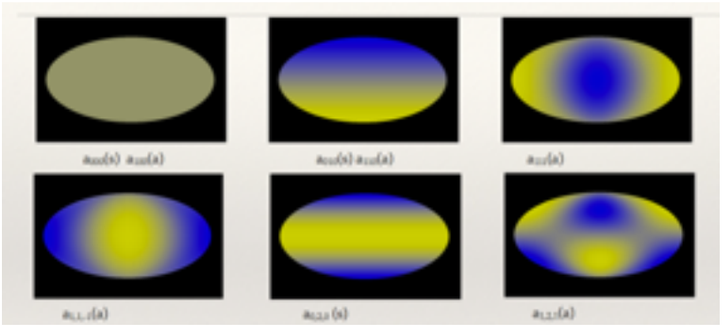
- New and innovative method for measuring angular distributions
  - can be written with orthogonal functions



$$a^m_{k,l} = \int \rho(\theta_1,\theta_2,\phi) M^{m*}_{kl}(\theta_1,\theta_2,\phi) d\Omega^M$$

$$M^m_{k,l}(\theta_1,\theta_2,\phi) = \sqrt{2\pi} Y^m_k(\theta_1,0) Y^m_l(\theta_2,\phi)$$

- Examples:
  - Single-top production & decay
  - Heavy Higgs decay to two vector bosons
  - Obtaining coefficients ($a^m_{k,l}$) using Monte Carlo integration
  - Deconvolution for Detector effects



Deconvolving the detector from an observed signal in Fourier space (*J. Boudreau*)

Deconvolve. Conceptually:

$$\mathbf{G} \cdot \vec{a} = \vec{\mathcal{A}}$$

From MC — Reco coefficients measured in data

Physics coefficients. You want. $\vec{a} = \mathbf{G}^{-1} \cdot \vec{\mathcal{A}}$

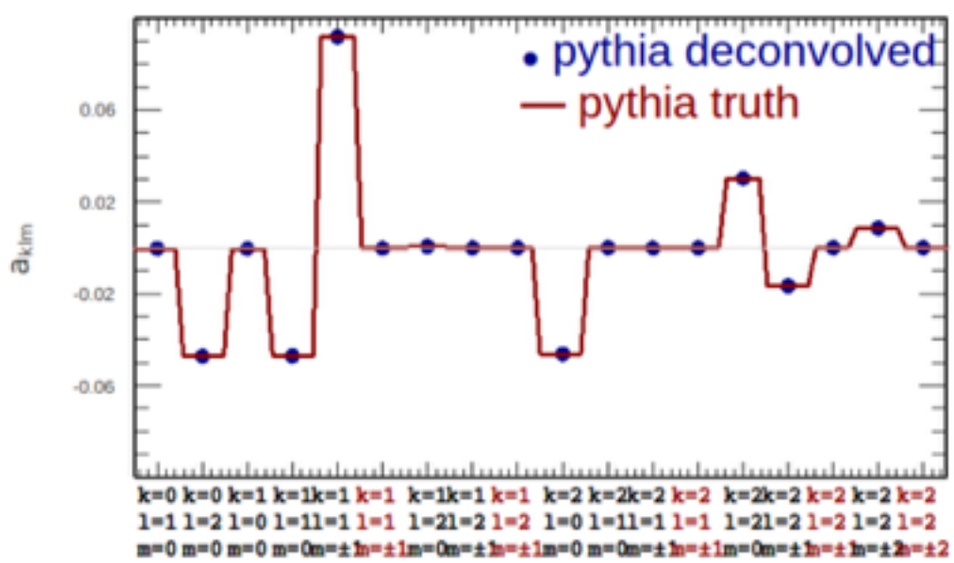
But given that you G is a rectangular matrix in general, you will have to minimize a χ^2:

$$\chi^2(\vec{a}) = (\vec{\mathcal{A}} - \mathbf{G} \cdot \vec{a})^T \cdot \mathbf{C}^{-1} \cdot (\vec{\mathcal{A}} - \mathbf{G} \cdot \vec{a})$$

With analytic solution and error matrix:

$$\vec{a} = (\mathbf{G}^T\mathbf{C}^{-1}\mathbf{G})^{-1}\mathbf{G}^T\mathbf{C}^{-1}\vec{\mathcal{A}} \qquad \mathbf{V} \equiv \mathrm{Cov}(\vec{a}) = (\mathbf{G}^T\mathbf{C}^{-1}\mathbf{G})^{-1}$$

32



Deconvolving the detector from an observed signal in Fourier space: the recipe

1. Do an orthogonal series analysis of the Monte Carlo in the space of true and reconstructed angles.
2. Coefficient Conversion: Joint PDF-> Conditional PDF & determination of **G**. Procedure is sketched in the previous slides, will be fully described in proceedings.
3. Do an orthogonal series analysis of data sample to determine coefficients $\mathcal{A}$ of reconstructed angular distributions and their covariance matrix **C**.
4. Apply this equation:

$$\vec{a} = (\mathbf{G}^T\mathbf{C}^{-1}\mathbf{G})^{-1}\mathbf{G}^T\mathbf{C}^{-1}\vec{\mathcal{A}} \qquad \mathbf{V} \equiv \mathrm{Cov}(\vec{a}) = (\mathbf{G}^T\mathbf{C}^{-1}\mathbf{G})^{-1}$$

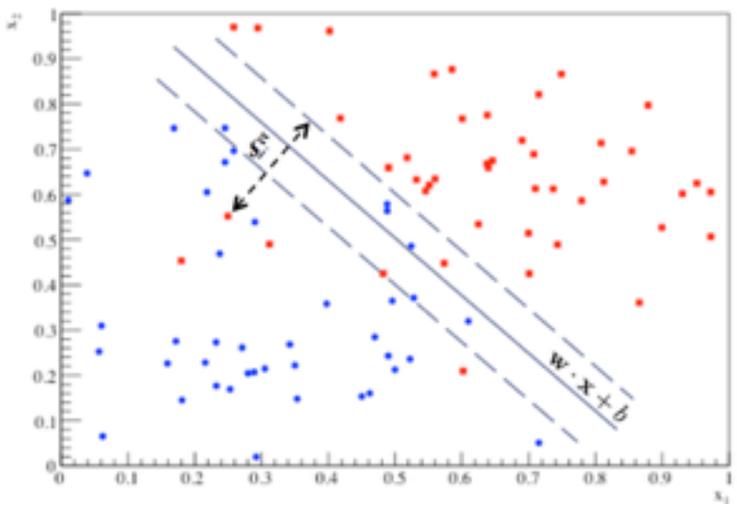
to obtain the physics coefficients & the full covariance thereof.

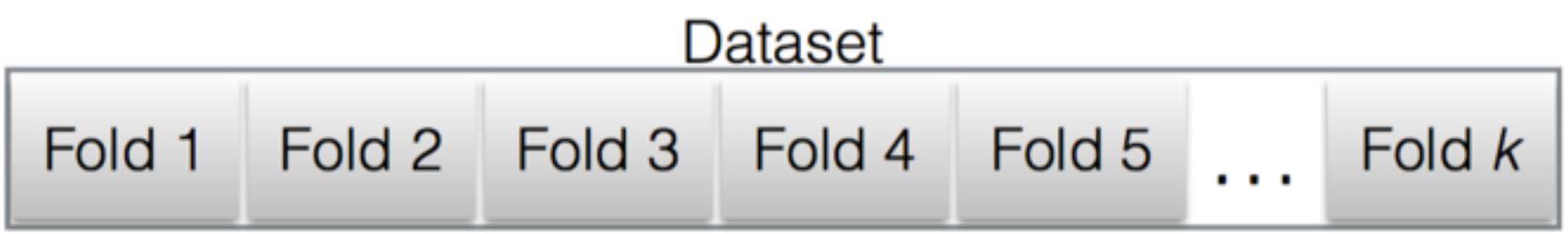
Then propagate the measurement to fundamental parameters: coupling constants in th interaction Lagrangian

36

Support Vector Machines and Generalisation for HEP (*T. Stevenson*)

- New Support Vector Machine method for HEP
  - include a larger set of Kernel functions
  - will be soon included in new TMVA release
- k-fold Cross Validation Method
  - Also soon available for TMVA



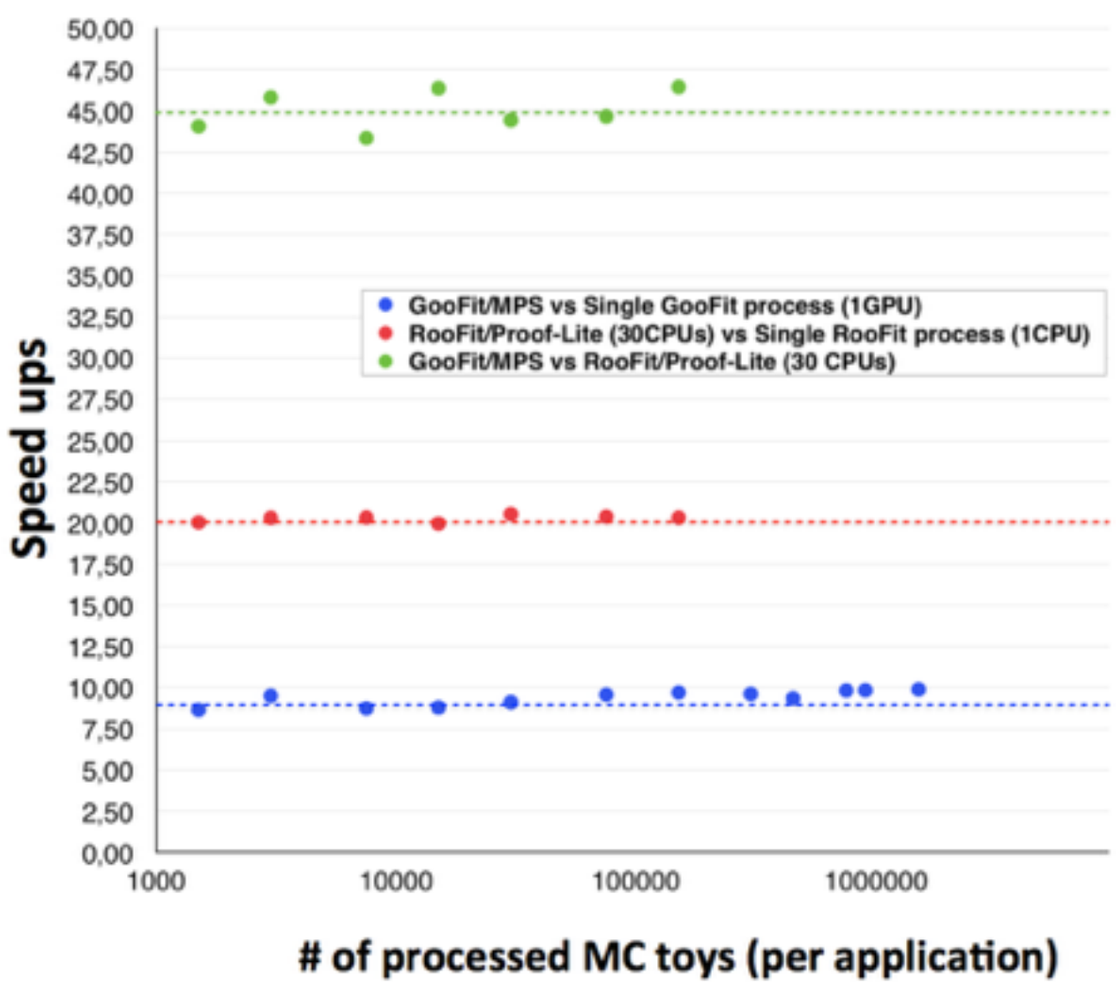


- Split the dataset into *k* randomly sampled independent subsets (folds).
- Train classifier with *k*-1 folds and test with remaining fold.
- Repeat k times.

GPU for Statistical Data Analysis in HEP (*A. Pom*

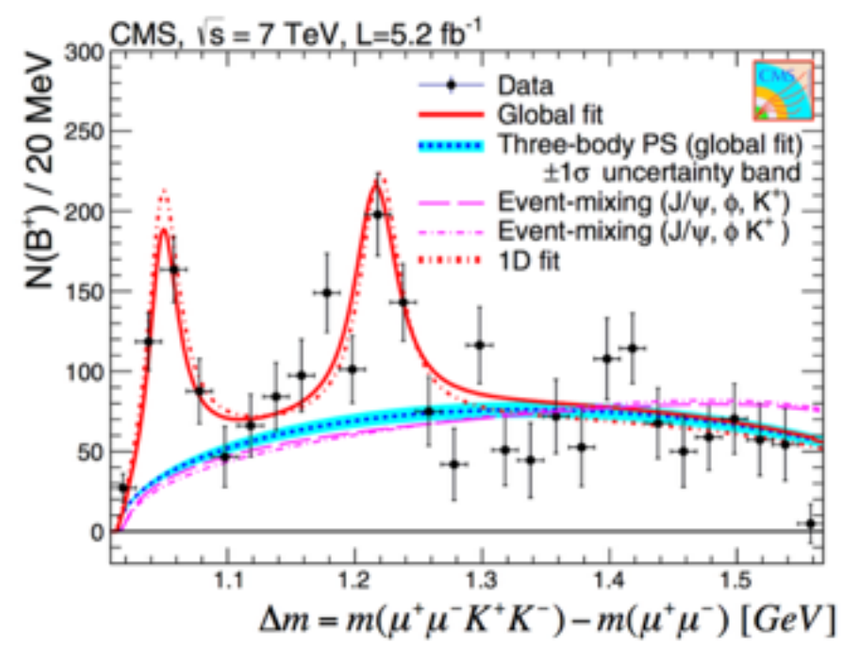
- Performance study of GooFit (on GPU) vs RooFit (on CPU)
  - parallel GooFit (CPU & GPU) vs parallel RooFit (with ProofLite) on CPU

We can compare: - 1 PROOF-Lite job using 30 workers (on 30 CPU cores)
with : - 2 GooFit/MPS jobs (each one running 15 simultaneous processes)



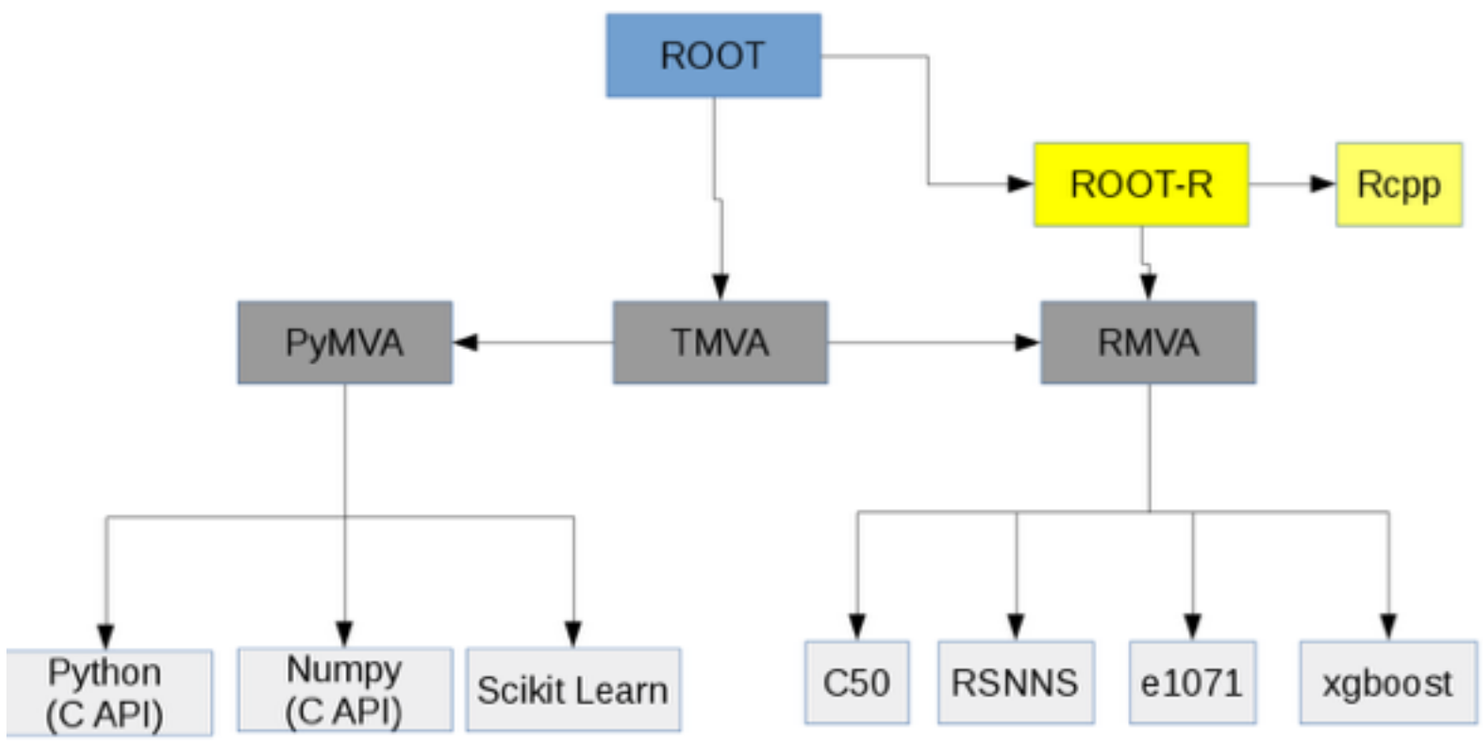
~45

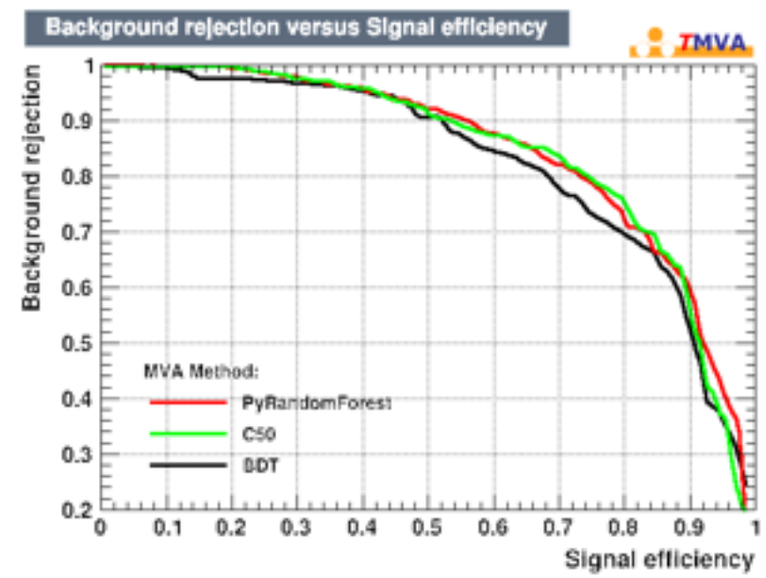
$$S_{n=30=N}\Big|_{2-TK20} = \frac{T^{RooFit}_{n=30}}{T^{GooFit}_{N=30}\Big|_{2-TK20}}$$

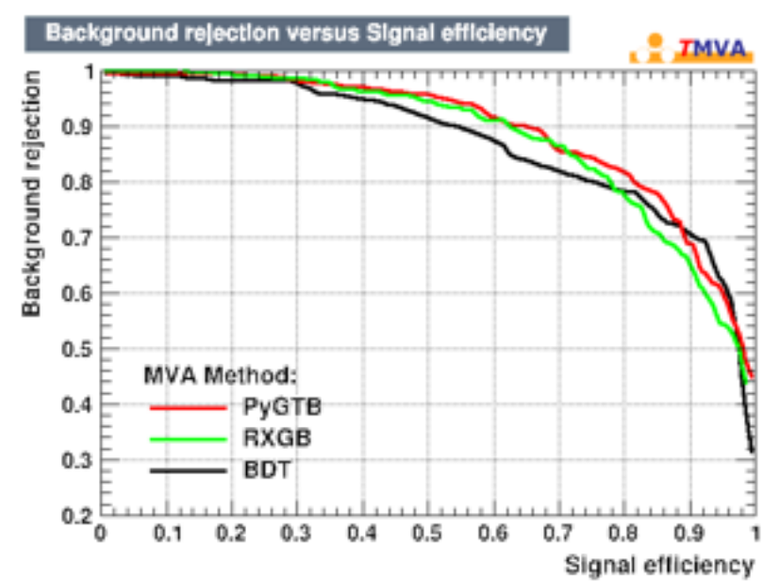


New Developments of Machine Learning Tools in ROOT (*O. Zapata, L.M.*)

- Integration in ROOT of algorithms based on R (using ROOT-R interface) and Python
- Modernisation of TMVA (DataLoader, Variable Importance)







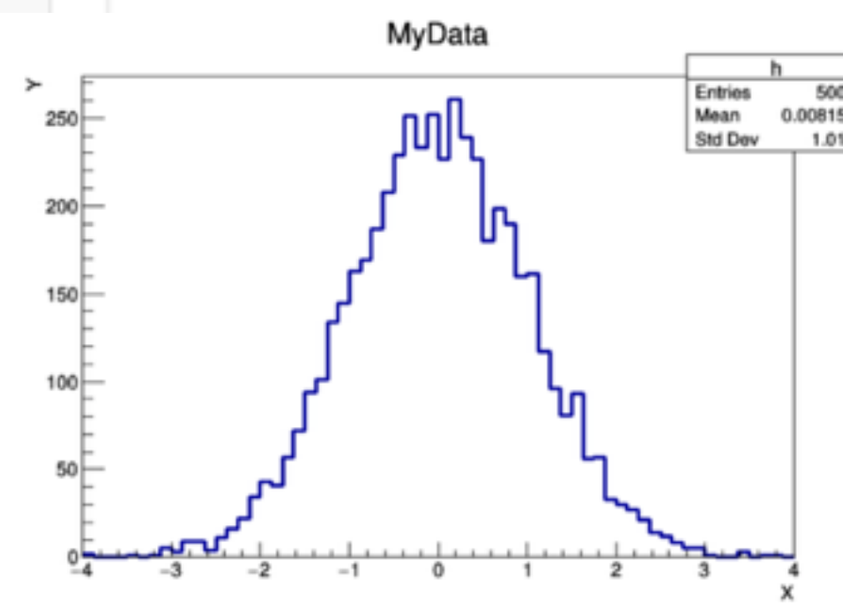
Data Mining as a Service with ROOT (*E. Tejedor*)

- ROOT Jupiter Notebooks
  - Python and C++ kernels
  - Very useful tool for prototyping, tutorials and training

File Edit View Insert Cell Kernel Help Python 2

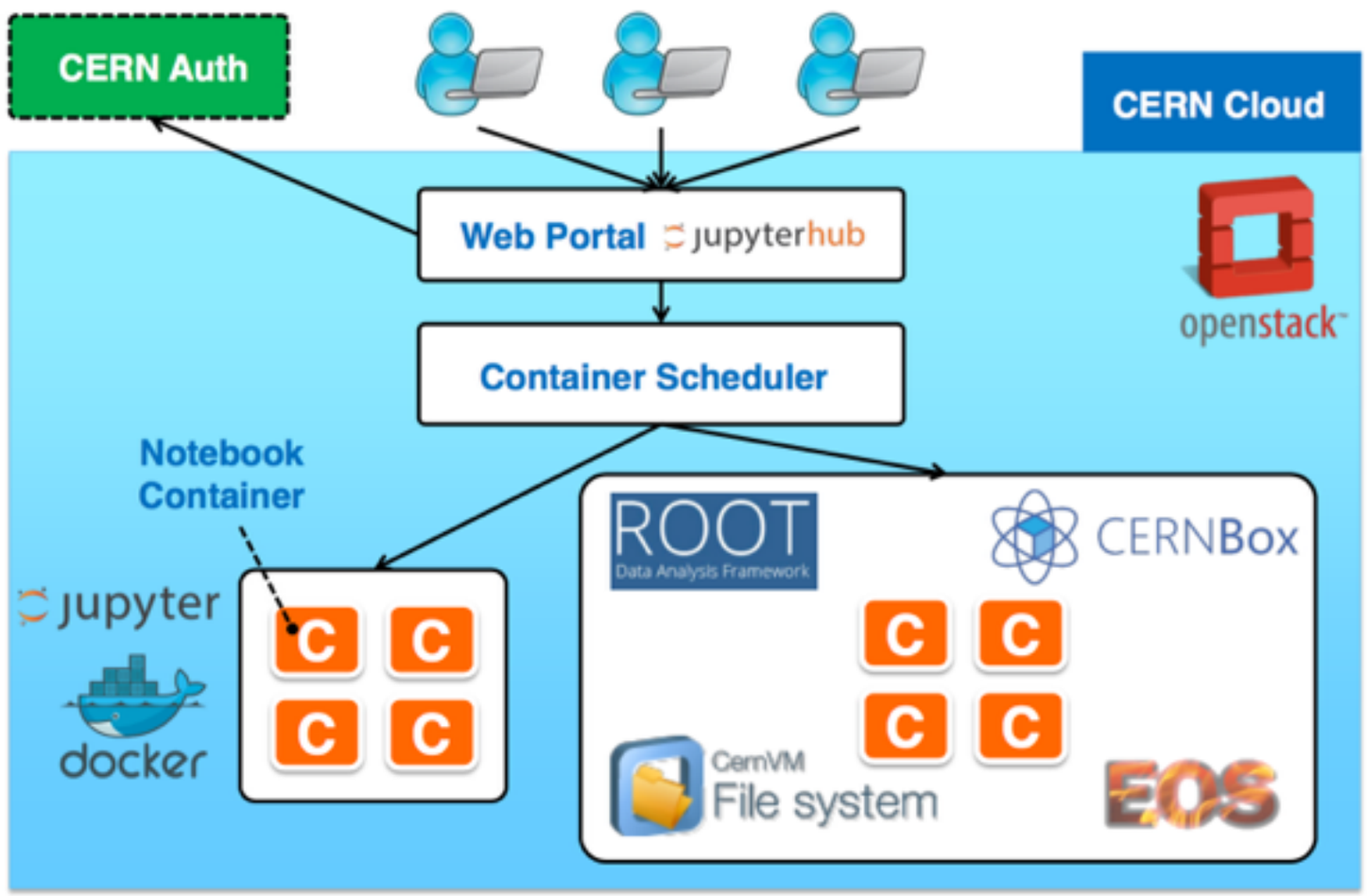
Code Cell Toolbar: None

```
In [1]: import ROOT # This triggers the integration layer

        Welcome to JupyROOT 6.07/03

In [2]: %%cpp
        auto myHisto = TH1F("h","MyData;X;Y",64,-4,4); // C++11

In [3]: h = ROOT.myHisto # Find the variable back in Python!
        h.FillRandom("gaus")
        c = ROOT.TCanvas()
        h.Draw()
        c.Draw()
```



Data Mining as a Service with ROOT (*E. Tejedor*)

- Provide centrally ROOT as a service
  - integrated in CERN computing infrastructure
    - Cernbox (EOS) to store the data and notebooks



Conclusions

- Very interesting presentations
  - new methods and ideas presented
    - lots of them based on Machine Learning
  - very active and lively discussions
- Really following ACAT spirits
    - more workshop than a conference
    - bridging knowledge between physics (analysis methods) and computer science
- Big thanks to the organisers for hosting us in this beautiful place
- Looking forward for next ACAT