Unfolding with ML using multiple reconstructed variables.

S. Glazov, IML Machine Learning Working Group meeting,
12/10/18.

Unfolding methods

The data unfolding problem:

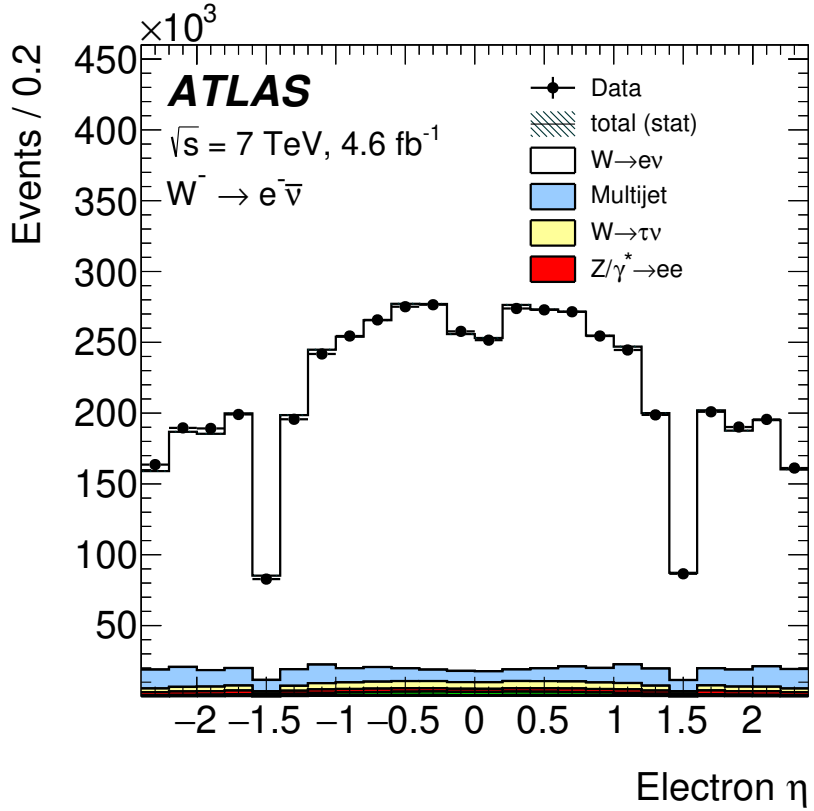
$$r = Mt$$

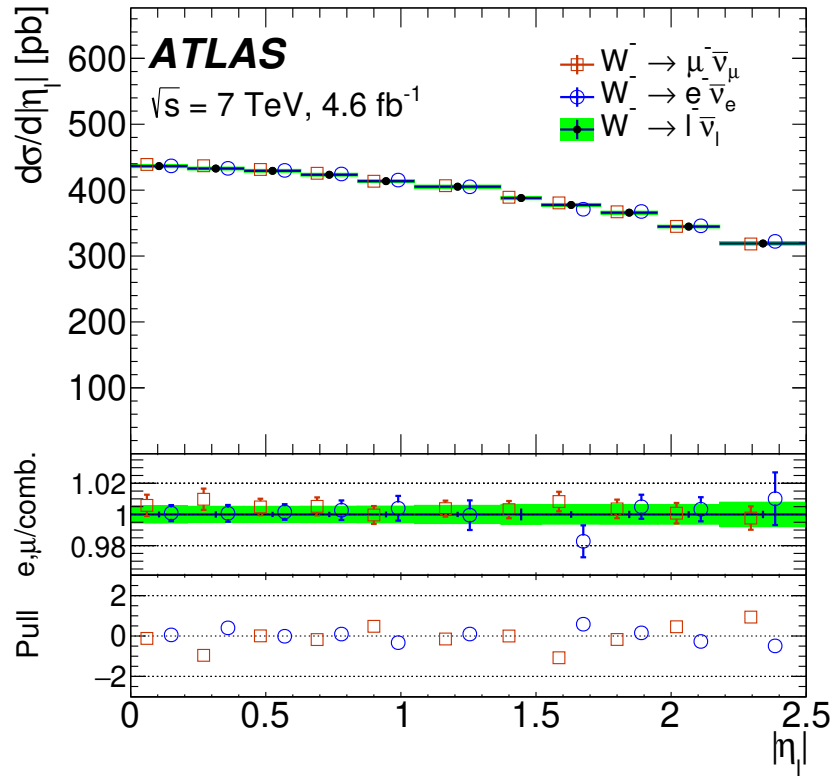
- Matrix inversion: $M^{-1}r = t$.
- Keeping large eigenvalues of M only: SVD, NIM A372, 496 (1996).
- Regularization using "smoothness" prior τ for t or its derivatives. E.g.: Sov. Math **5**, 1035 (1963), NIM 216, 183 (1983).
- Explicit "truth" prior using cross-entropy term: NIM A340, 400 (1994).
- Implicit truth prior in iterative unfolding procedures, e.g. Bayesian unfolding NIM 216, 177 (1983), NIM A362, 487 (1995). Iterative methods may be un-binned, using event-level weights NIM A355, 516 (1995).
- Folding methods: fold data multiple times and extrapolate back to zero foldings NIM A701, 17 (2013).

Choice of reconstructed variables for unfolding

- The choice of truth-level definition is connected to the variable used at reconstructed level.
- Usually they are chosen as close as possible to each other, which typically leads to reduced model dependence.
- However in some cases improved resolution may be more important than increased model dependence. Moreover, known explicit model dependence may be preferred if its uncertainty can be estimated in a well-defined way.
- To improve resolution, one may consider using several reco-level variables for unfolding.
- The combination of the information can be non-trivial due to in general non-Gaussian shape of reco. variables: use ML methods for that.

Example 1: η_ℓ distribution

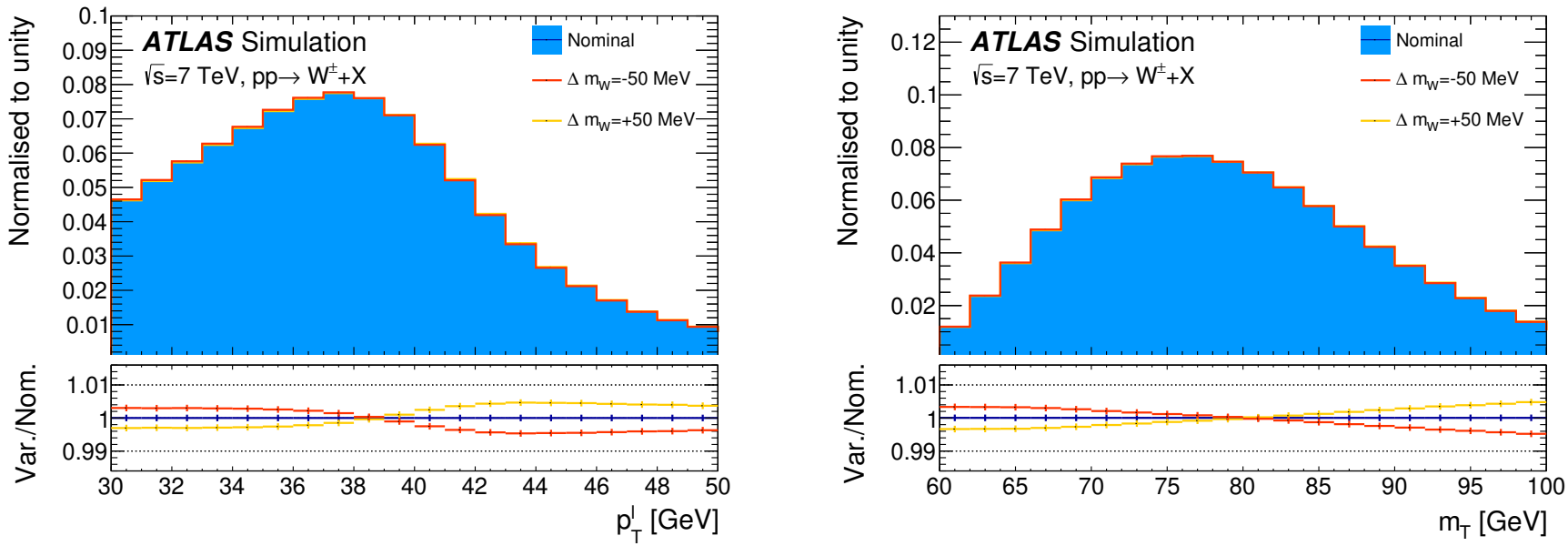




- Excellent resolution for η_ℓ with purity at 90% level for the bin sizes close to optimal in terms of underlying physics (e.g. variation of PDFs).
- $\rightarrow$ no need to think about extra variables.

arXiv:1612.03016

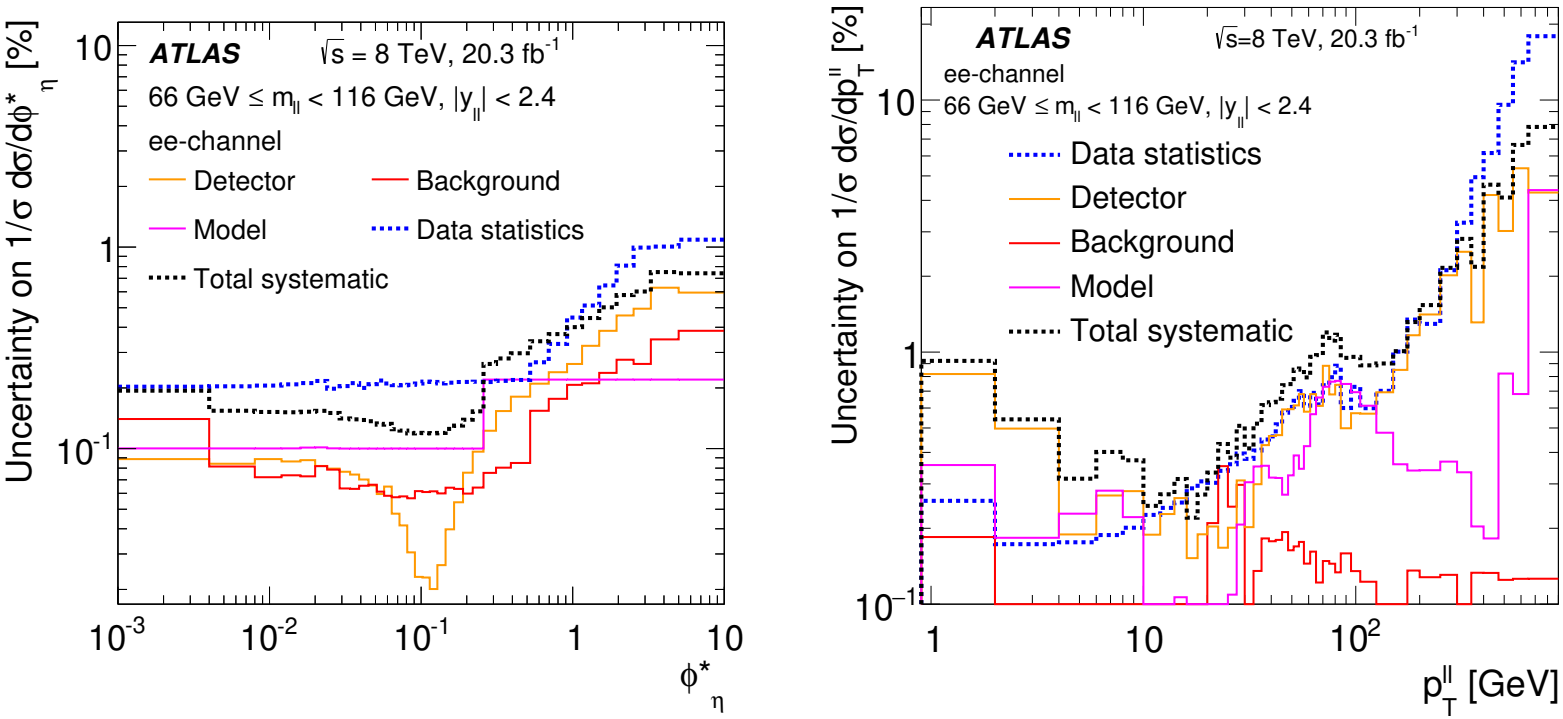
Example 2: m_W unfolding



- m_W "unfolded" using reco level fits to p_T^ℓ and M_T distributions.
- Reco and unfolded variables are pretty far from each other, this can not be avoided.
- The combination is performed after unfolding, providing extra consistency check.
- Modeling uncertainties have the largest contribution for the result, however they are studied in a systematic way.

arXiv:1701.07240

Example 3: Z_{p_T} and Z_{ϕ^*}



- ϕ^*_η and $p_{\mathrm{T}}^{\ell\ell}$ variables probe very similar physics, with ϕ^*_η having much better detector resolution and $p_{\mathrm{T}}^{\ell\ell}$ closer to the underlying physics.
- Additional interest is to probe low p_{T} region to study location of the peak: resolution is essential.
- $\rightarrow$ use both variables at reco level to unfold to single p_{T}. Maybe add $M_{\ell\ell}$ information to the mix, to improve the resolution.

arXiv:1512.02192

A method for ML unfolding

- Determination of the truth-bin using reco-level variables is a classification problem, well suited for ML methods.
- Use 3-layer NN: input later, fully connected layer, and final "categorical" layer with softmax ($f_i(x) = \exp x_i / \sum_{k=1}^{N_{\rm bin}} \exp x_k$) activation. Use categorical cross entropy as the loss function.

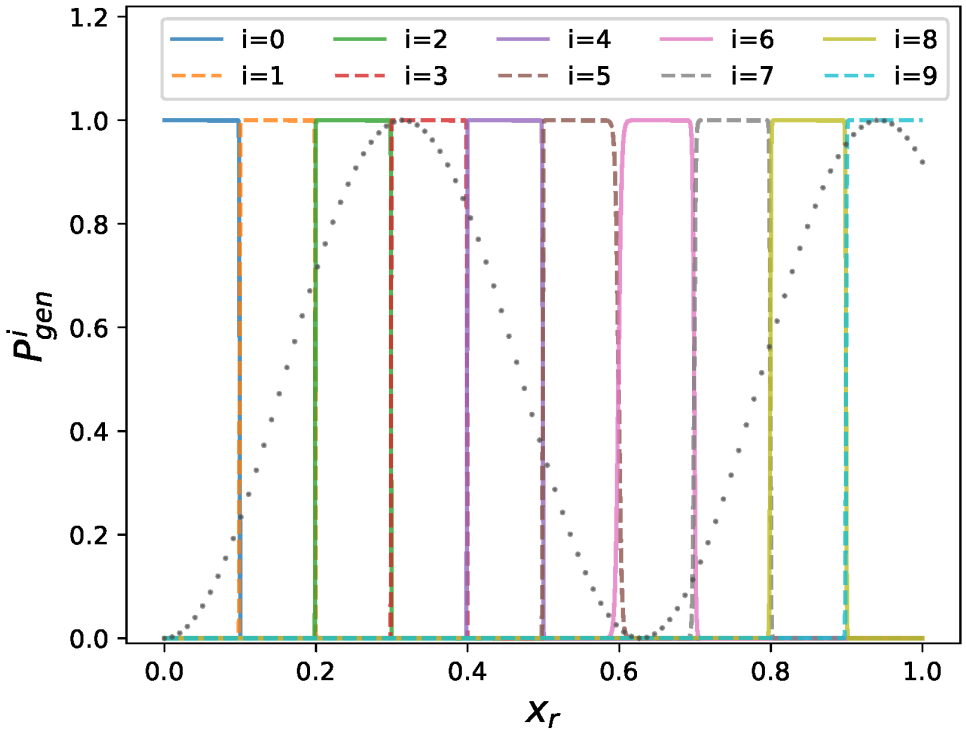
$$H(p,q) = -\sum_{e=1}^{N_{\rm evt}} \sum_{i}^{N_{\rm bin}} p_i^e \log q_i^e$$

- $\rightarrow$ a version of event-level iterative unfolding.
- Implemented in Keras:

```
model = Sequential()
model.add(Dense(nvar,activation='linear',input_shape=(nvar,1)))
model.add(Dense(kappa*NBins**2,activation='relu'))
model.add(Dense(NBins,activation='softmax'))
model.compile(loss=keras.losses.categorical_crossentropy,
          optimizer=keras.optimizers.Adadelta(),
          metrics=['accuracy'])
```

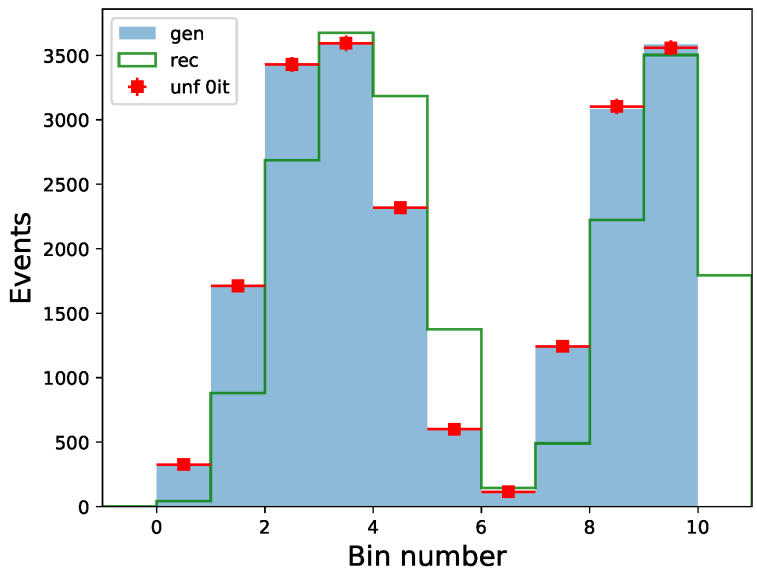
arXiv:1712.01814

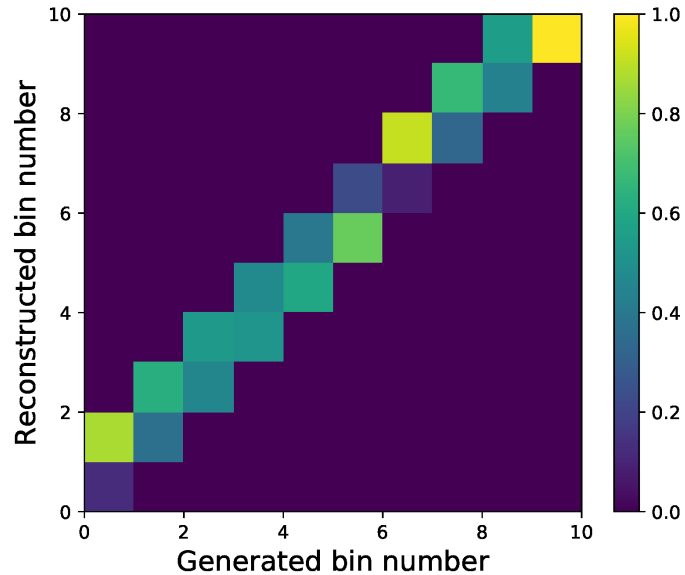
Test 0: determine the bin boundaries



- Proof of principle test: use single reco-level variable, unfold $f(x) = \sin^2(5x)$ distribution for $0 < x < 1$ using 10 bins.
- Start with no-smearing: the classifier needs to find truth-bin boundaries.
- The test is useful to determine the model topology and tune minimization algorithm.

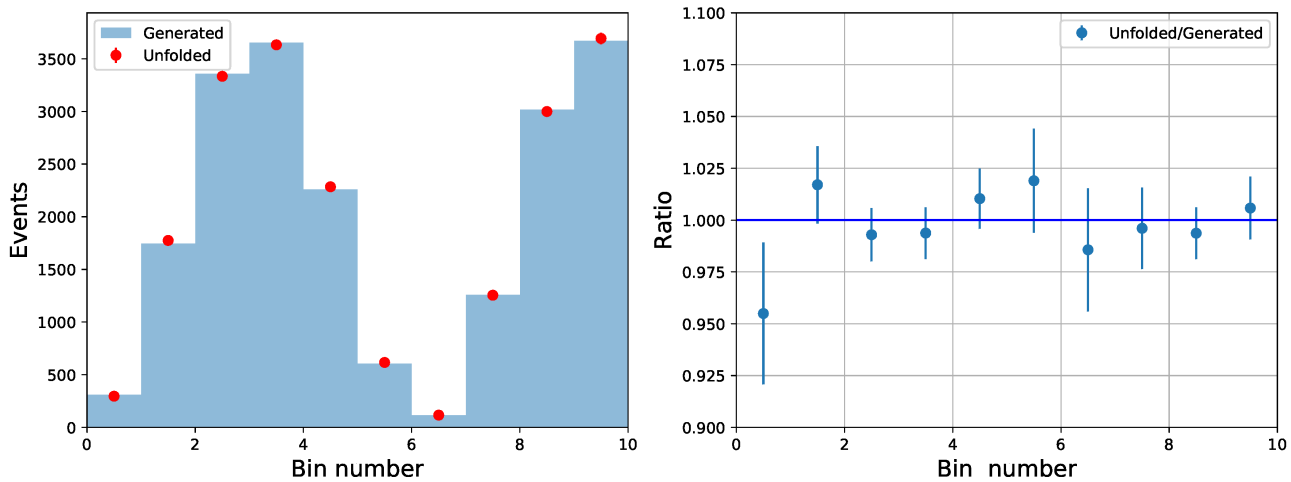
Automatic calibration





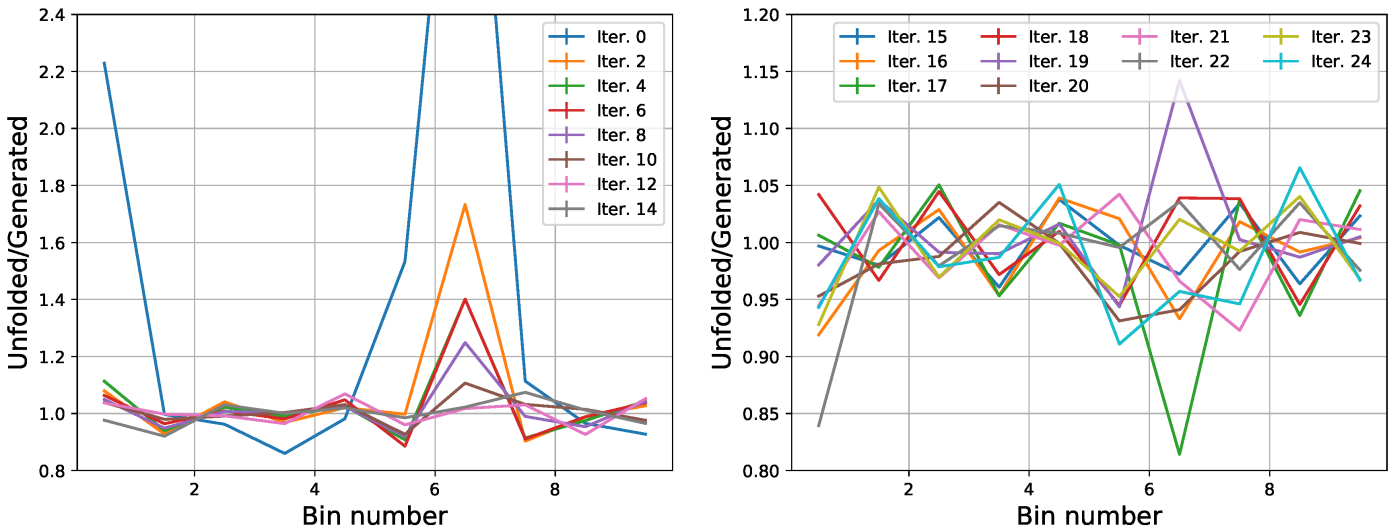
- Since the method determines effective bin boundaries, it automatically calibrates for reco vs truth mis-calibrations.
- Example: shift reconstructed variable by 0.5 bin width.

Proof of principle example



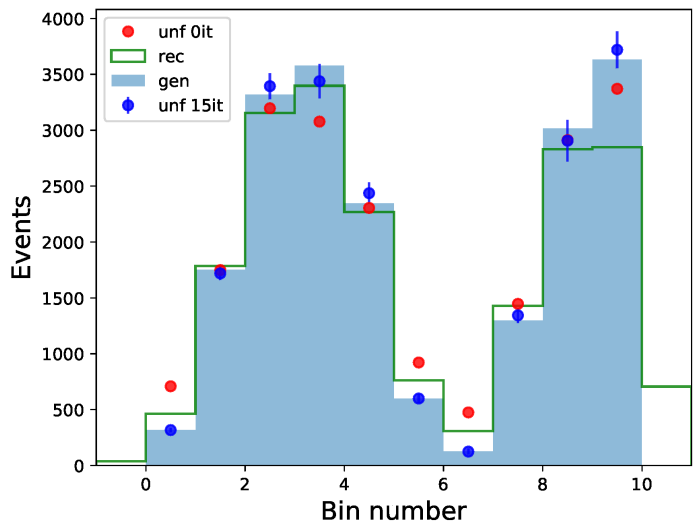
- Unfolding is done in an iterative procedure: updating the distribution of the truth variable based on previous iteration and re-training.
- Convergence is hard to prove mathematically: use numerical closure tests.
- Add smearing (0.5 bin Gaussian), first test uses truth-distribution as the prior. Use bootstrap method to estimate uncertainties.

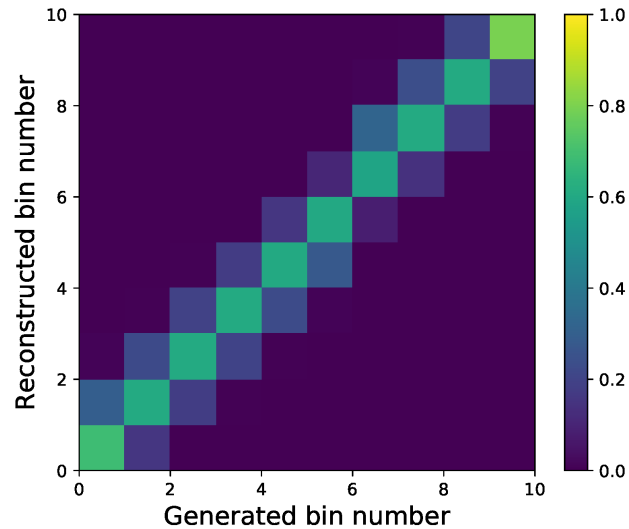
Convergence of the method



- Start with the flat prior, observe convergence to the truth distribution after about ten iterations. After that the unfolding starts to oscillate around the minimum.
- More details can be found in `https://github.com/aglazov/MLUnfold.git` which contains a jupyter notebook (Keras).

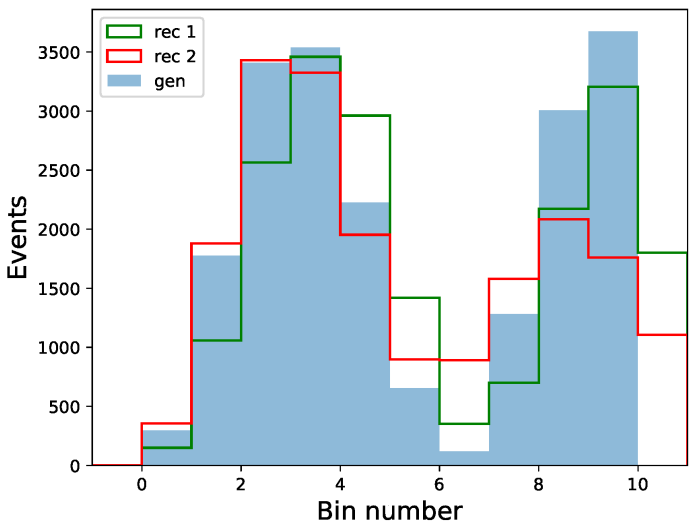
Unfolding results for one variable

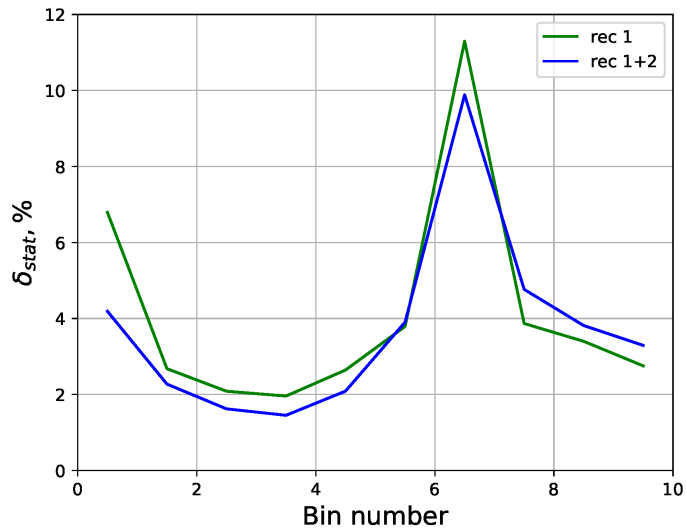




- The initial iteration of ML unfolding with flat prior acts as "extra smearing" vs reconstructed distribution (similar to iterative Bayesian unfolding).
- No bias within uncertainties after 15th iteration.
- Purities are at 60% level with exception of the bin at the minimum where the purity is 20% only; the accuracy of the ML-unfolding predictions is about 65%.

Advantage of second variable





- Add second variable:
  - 1st with 0.5-bin Gaussian smearing and +0.5 bin shift;
  - 2nd: with 1.5 bin log-normal smearing and -0.5 bin shift.
- The second variable improves convergence (5 iteration instead of 10), leads to reduction of uncertainty for the first bins.

Discussion

- Selection of the definition at the truth level can be complemented by selection of an optimal reco-level variable.
- The selection of the optimal reco-variable can be performed using ML methods, based on multiple input variables.
- This can be useful in various situations: when the connection with truth is not straightforward due to missing information (e.g. m_W, p_{T_W}, y_W); when the resolution is of critical importance (e.g. p_{T_Z} at low p_T); when the full event information may bring additional calibration info (e.g. balance in Z+jet events for jet kinematics).