V. FORMATO - 27/11/19

UNFOLDING

THE UNFOLDING PROBLEM

Consider a random variable y.

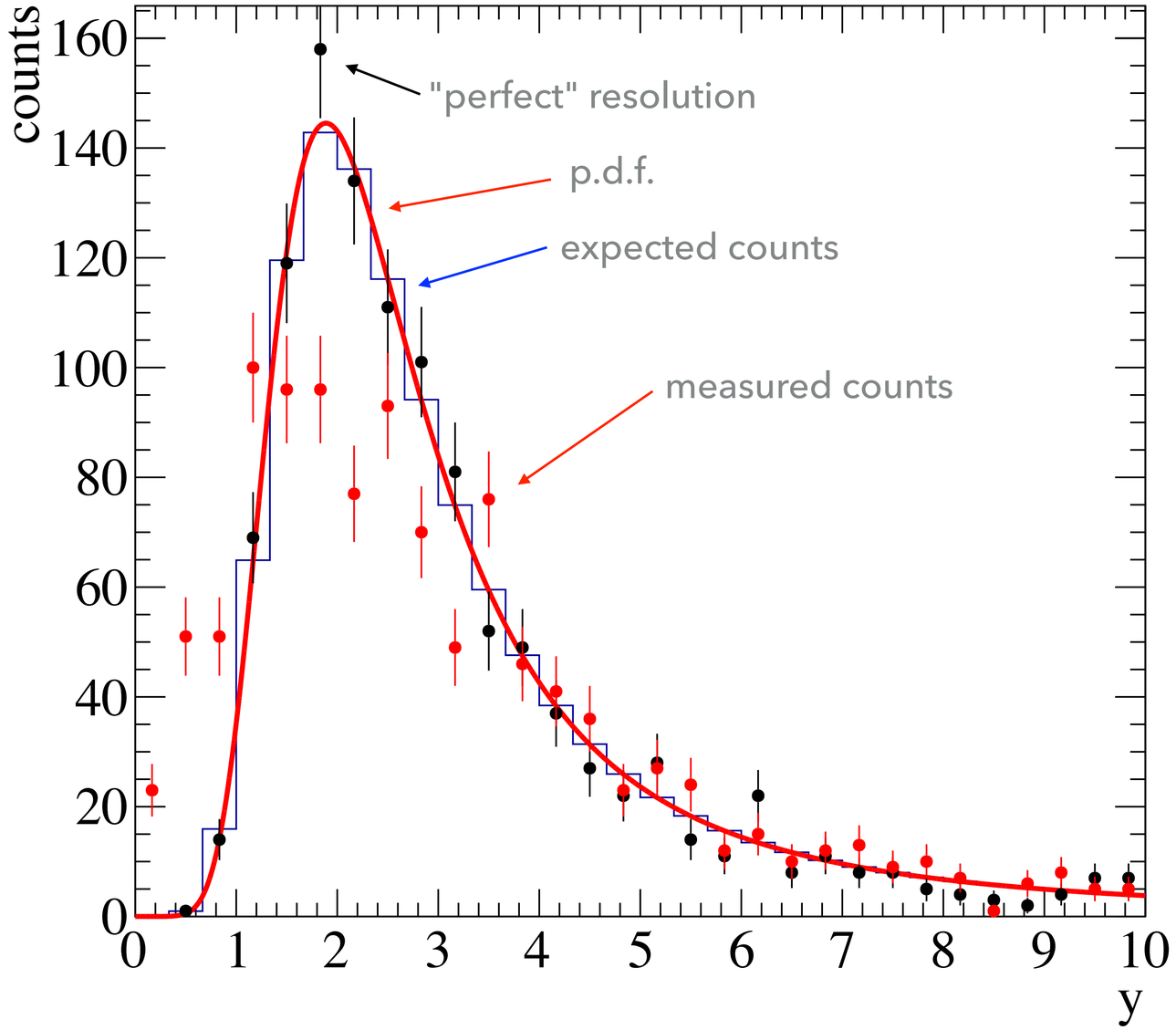
We might want to measure its p.d.f. $f(y)$, of, if the p.d.f. is known, we might want to measure some of its parameters...

Real detectors with their intrinsic resolution make things worse:

Generally $f(y)$ gets "smeared" by resolution effects: peaks broaden, tails get more populated, etc...

$$\tilde{f}(x) = \int K(x|y) f(y)\, dy$$

measured distribution — response function — true distribution



MIGRATION

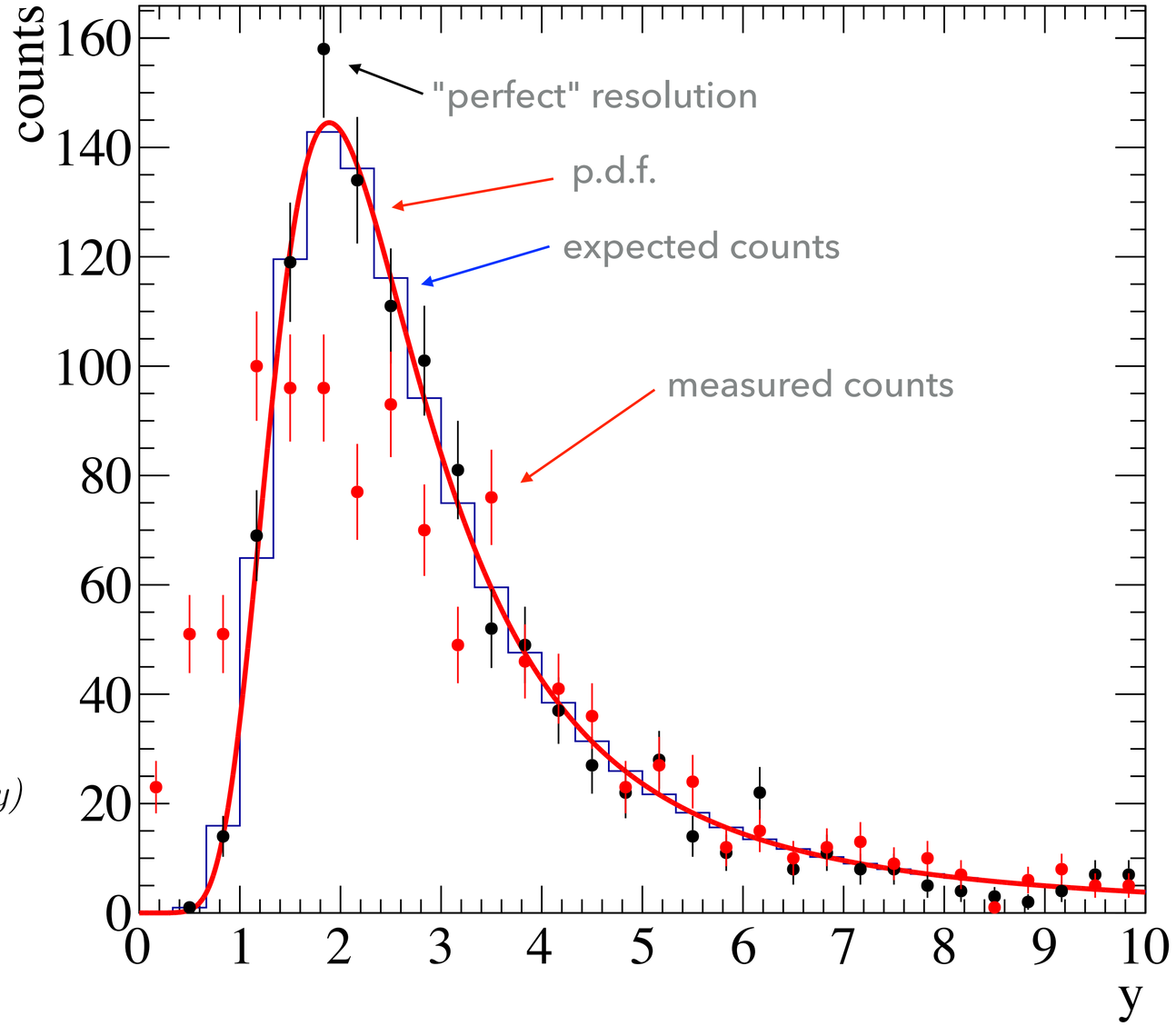
$$\tilde{f}(x) = \int K(x|y) f(y)\, dy$$

Type-1 Fredholm equation

The goal is to find $f(y)$: we need an estimator (possibly unbiased, efficient e consistent)

If we know a possible parametrization of $f(y;\, \theta)$ from theory, we can try to "fold" it with K and compare the result with data → parameter estimation.

If the shape of $f(y)$ is unknown, the unfolding problem reduces to "Find $f(y)$ given $f(x)$"



MIGRATION

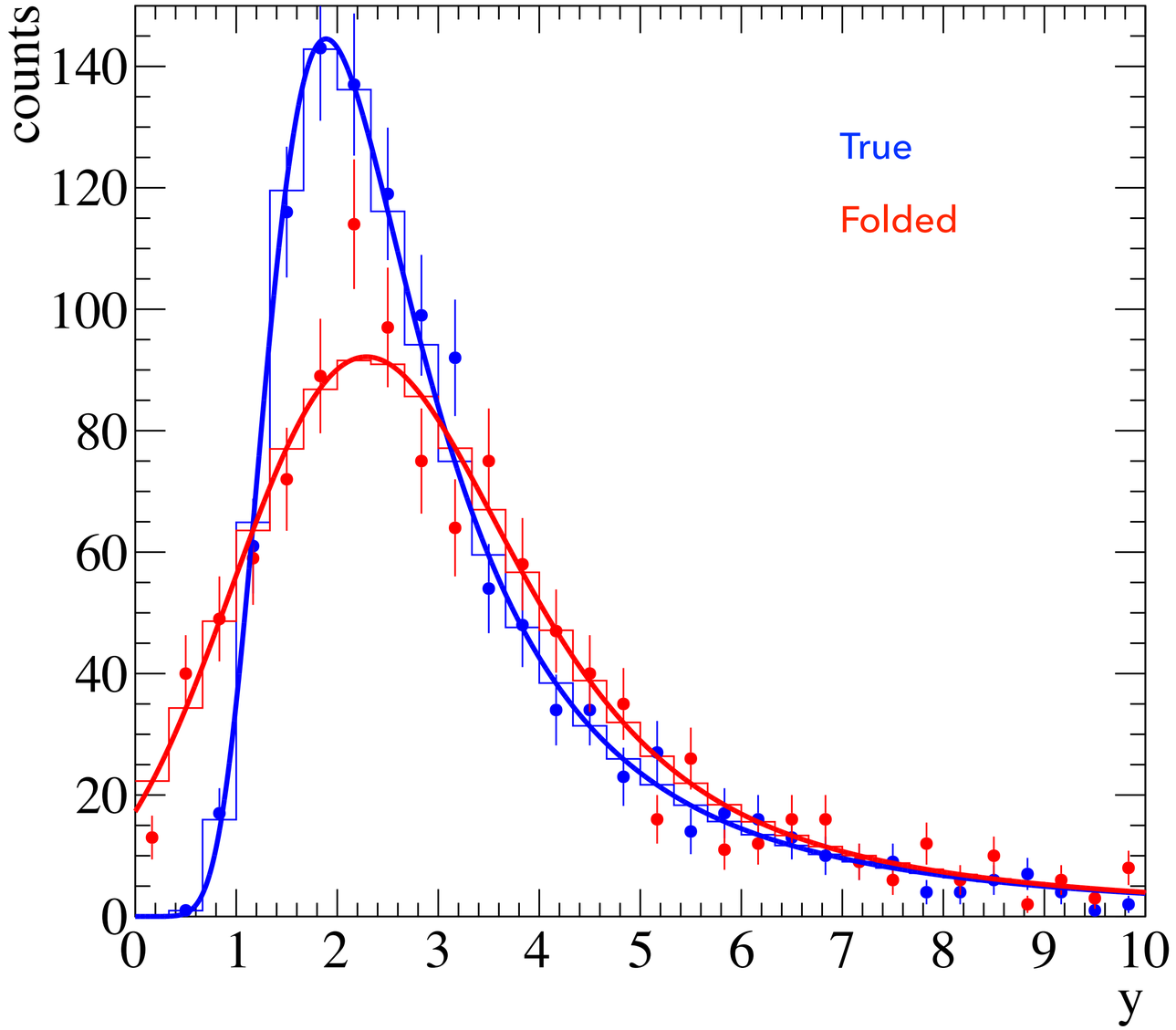
In the real world we are limited by resolution and statistics → work with histogram: discretization.

$$f(y) \to \mu_i = \int_{y_i}^{y_{i+1}} f(y)\, dy$$

$$\tilde{f}(x) \to \nu_i = \int_{x_i}^{x_{i+1}} \tilde{f}(x)\, dx$$

$$K(x|y) \to R_{ij} = \frac{\mu_j}{\nu_i}$$

$$\nu_i = \mathbb{E}(n_i) = \sum_{j=1}^{M} R_{ij}\mu_j$$

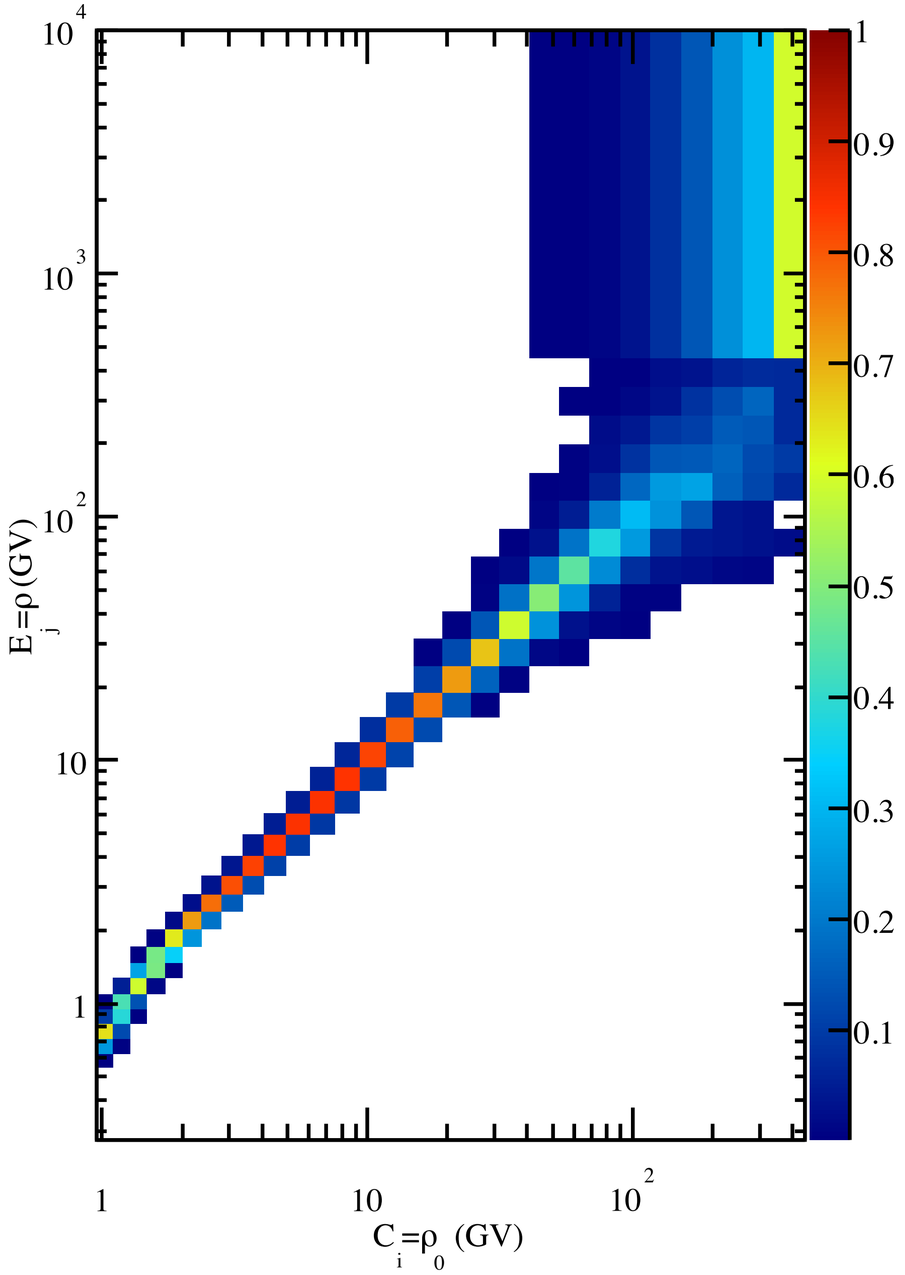


THE MATRIX

The general definition is:

$$R_{ij} = \frac{\int_{x_i}^{x_{i+1}} \int_{y_j}^{y_{j+1}} K(x|y)\, dx\, dy}{\int_{x_j}^{x_{j+1}} f(y)\, dy}$$

If the kernel K is known, the matrix can be derived analytically, otherwise it can be estimated from Montecarlo simulations.



EFFICIENCY

Some events may be lost and not measured due to detector inefficiency or dead areas in its acceptance. This effect can be included in the resolution matrix.

$$\sum_{j=1}^{M} R_{ij} = \mathbb{P}(\text{observed in any bin} \mid \text{true value in bin } i) = \varepsilon_i$$

Our sample might even contain background events, that may alter the distribution of x

$$\tilde{f}(x) = \int K(x|y) f(y) + b(y)\, dy$$

$$\beta_i = \int_{y_i}^{y_{i+1}} b(y)\, dy$$

$$\mathbb{E}(n_i) = \nu_i = \sum_{j=1}^{M} R_{ij}\mu_j + \beta_i$$

FIRST SOLUTION: MAXIMUM LIKELIHOOD

The problem we face has the form $\boldsymbol{\nu} = R\boldsymbol{\mu} + \boldsymbol{\beta}$. It might seem natural to assume the solution is

$$\boldsymbol{\mu} = R^{-1}(\boldsymbol{\nu} - \boldsymbol{\beta})$$

If our data follow a Poisson statistics

$$\mathbb{P}(n_i|\nu_i) = \nu_i^{n_i}\,\frac{e^{-\nu_i}}{n_i!}$$

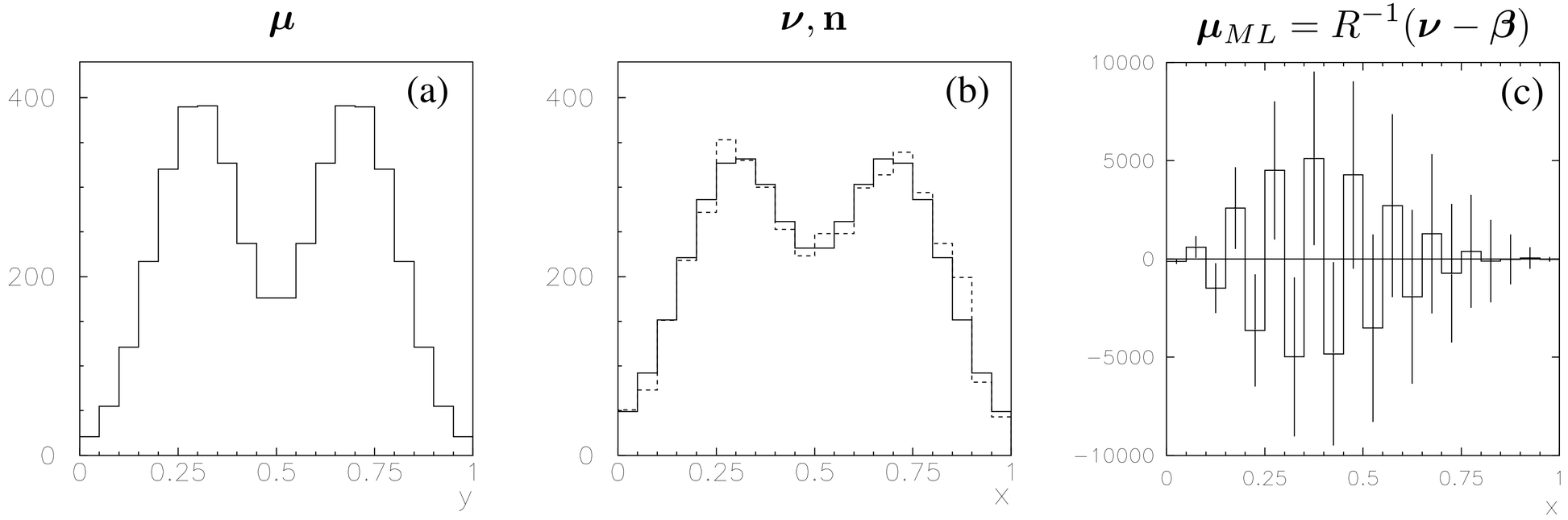
We can compute the log-likelihood: $\log \mathcal{L}(\boldsymbol{\mu}) = \sum_{i=1}^{N} n_i \log \nu_i - \nu_i \log n_i!$ and its maximum is (unsurprisingly)

$$\boldsymbol{\nu}_{ML} = \mathbf{n} \quad \Rightarrow \quad \boldsymbol{\mu}_{ML} = R^{-1}(\boldsymbol{\nu} - \boldsymbol{\beta})$$

...will it work?

FIRST SOLUTION: MAXIMUM LIKELIHOOD

(Example from http://www.ippp.dur.ac.uk/Workshops/02/statistics/proceedings/cowan.ps)



1) Statistical fluctuations are not foreseen in R, applying R^{-1} amplifies them, as if they were "physical"

2) From a mathematical perspective the problem is not well formulated. R might even be not invertible...

INTERMISSION: SINGULAR VALUE DECOMPOSITION

Given a m x n A.

it can be written as $A = USV^T$

where U is a m x m matrix, V a n x n matrix, and S a m x n matrix with positive non-null values on its diagonal (s_i).

The various s_i are called "singular values" ant the columns of U and V are called left and right "singular vectors".

How can this help? 🤔

It is a generalization of "diagonalizing" a matrix. (http://people.csail.mit.edu/hasinoff/320/SingularValueDecomposition.pdf)

Example: $A = \frac{1}{2}\begin{pmatrix} 1+\varepsilon & 1-\varepsilon \\ 1-\varepsilon & 1+\varepsilon \end{pmatrix} \longrightarrow U = V = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$, $S = \begin{pmatrix} 1 & 0 \\ 0 & \varepsilon \end{pmatrix}$

$$A^{-1} = VS^{-1}U^T = \frac{1}{2}\begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix} + \frac{1}{2\varepsilon}\begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix}$$

INTERMISSION: SINGULAR VALUE DECOMPOSITION

Example:

$$\mathbf{n} = \begin{pmatrix} n_1 \\ n_2 \end{pmatrix} \quad , \qquad \boldsymbol{\beta} = \mathbf{0} \quad , \qquad \hat{V} = \begin{pmatrix} n_1 & 0 \\ 0 & n_2 \end{pmatrix}$$

$$R\boldsymbol{\mu} = \mathbf{n} \quad \Rightarrow \quad S(V^T\boldsymbol{\mu}) = U^T \quad \Rightarrow \quad S\mathbf{z} = \mathbf{d}$$

$$\mathbf{z} = \frac{1}{\sqrt{2}} \begin{pmatrix} \mu_1 + \mu_2 \\ \mu_1 - \mu_2 \end{pmatrix} \quad , \qquad \mathbf{d} = \frac{1}{\sqrt{2}} \begin{pmatrix} n_1 + n_2 \\ n_1 - n_2 \end{pmatrix}$$

but this is now a diagonal system and we can solve it easily, rotating back:

$$\boldsymbol{\mu} = V\mathbf{z} = \frac{n_1 - n_2}{2\varepsilon} \begin{pmatrix} 1 \\ -1 \end{pmatrix} + \frac{n_1 + n_2}{2} \begin{pmatrix} 1 \\ 1 \end{pmatrix}$$

The solution is a weighted sum of two eigenvectors of the resolution matrix. Suppose that $(n_1 - n_2)^2 \leq n_1 + n_2$ meaning that counts in the two bins are "statistically similar". The first part of the solution is actually a random number with huge fluctuations. But if ε is small enough, this term can dominate over the more stable one!

(Understandable: for $\varepsilon \to 0$ the detector cannot really distinguish between the two bins...)

BEFORE WE THROW AWAY THIS METHOD...

$$\boldsymbol{\mu}_{ML} = R^{-1}(\boldsymbol{\nu} - \boldsymbol{\beta})$$ is unbiased

and its covariance is

$$V_{ij} = \mathrm{cov}[\mu_{ML,i},\, \mu_{ML,i}] = \sum_{k,l=1}^{N} (R^{-1})_{ik}(R^{-1})_{jl}\, \mathrm{cov}[n_k,\, n_l]$$

$$= \sum_{k,l=1}^{N} (R^{-1})_{ik}(R^{-1})_{jl}\, \delta_{kl}\nu_k = \sum_{k=1}^{N}(R^{-1})_{ik}(R^{-1})_{jk}\, \nu_k$$

(remember the plot: the errors were quite big)

BEFORE WE THROW AWAY THIS METHOD...

Cramér-Rao: unbiased estimator → variance has lower limit. This is our case, what's its Cramér-Rao limit?

$$-\mathbb{E}\left[\frac{\partial^2 \log \mathcal{L}}{\partial \mu_k \partial \mu_l}\right] = \sum_{i=1}^{N} \frac{R_{ik} R_{il}}{\nu_k}$$

remember the variance of the estimator...

$$V_{ij} = \sum_{k=1}^{N} (R^{-1})_{ik} (R^{-1})_{jk}\, \nu_k$$

This is actually the lowest variance estimator (among all unbiased estimators).

To have "reasonable" errors we might have to accept a small bias...

...compromising between bias and variance is the most critical part of any unfolding procedure.

CORRECTION FACTOR

Let's start from the Maximum Likelihood example: define $\mu_i = C_i(n_i - \beta_i)$ where $C_i = \mu_i^{\mathrm{MC}}/\nu_i^{\mathrm{MC}}$

this correction factor is estimated from the Montecarlo simulation and its variance is:

$$V_{ij} = \mathrm{cov}[\mu_i,\, \mu_j] = C_i C_j \mathrm{cov}[n_i,\, n_j]$$

often C_i is ~O(1), and the variance is considerably smaller w.r.t. the Maximum Likelihood case.

However its bias is $b = \left(\frac{\mu_i^{\mathrm{MC}}}{\nu_i^{\mathrm{MC}}} - \frac{\mu_i}{\nu_i - \beta_i}\right)(\nu_i - \beta_i)$ and it strongly depends on how well the MC can describe reality...

Considering $C_i = 0.1;\ \beta_i = 0;\ n_i = 100;\ \rightarrow\ \mu_i = C_i n_i = 10;\quad \sigma_{\mu_i} = C_i\sqrt{n_i} = 1$

and we get a 10% uncertainty. How is this possible if only 10 out of 100 events actually carry the information?

CORRECTION FACTOR

Features:

- C implicitly depends from the distribution we are trying to estimate
- Bin-to-bin correlations are completely ignored
- The total number of events before/after the correction is usually not preserved!

The reduction of variance is tied to a bias which is difficult to quantify.

A common trick to reduce the bias is choosing the bin width to be 2-3 times larger that the expected experimental resolution. (this is a good practice in general)

Used (the iterative version) in the AMS experiment

REGULARISED UNFOLDING

Let's take -2log$L$ as distance measure between n and the expected value ν
Consider the ML solution, and let's stay in a Neighbourhood, defined by some $\Delta \log L$. Between all the estimator we choose the "smoother" one, as long as it satisfies $\log \mathcal{L}(\hat{\mu}) \geq \log \mathcal{L}_{\max} - \Delta \log \mathcal{L}$

We are looking for an estimator maximising $\phi(\hat{\boldsymbol{\mu}}) = \log \mathcal{L}(\hat{\boldsymbol{\mu}}) + \tau S(\hat{\boldsymbol{\mu}})$

Regularization parameter

Regularization function

If we want to keep constant the total number of events we also have to add another constraint

$$\phi(\hat{\boldsymbol{\mu}}) = \log \mathcal{L}(\hat{\boldsymbol{\mu}}) + \tau S(\hat{\boldsymbol{\mu}}) + \lambda \left(\sum_{i=1}^{N} n_i - \nu_i \right)$$

What choices for S? What values for τ?

REGULARISED UNFOLDING

Consider

$$S[f(y)] = \int \left(\frac{d^k f}{dy^k} \right)^2 dy$$

which, for k=2, gives

$$S[\boldsymbol{\mu}] = \sum_{i=2}^{M-1} \left[(\mu_{i+1} - \mu_i) - (\mu_i - \mu_{i-1}) \right]^2$$

(we write our vectors in the basis defined by the covariance matrix)

$$\log \mathcal{L} = -\frac{1}{2}\chi^2 = -\frac{1}{2}(R\boldsymbol{\mu} - \mathbf{y})^T V_y^{-1} (R\boldsymbol{\mu} - \mathbf{y}) \quad = \quad -\frac{1}{2}(R'\boldsymbol{\mu}' - \mathbf{y}')^T (R'\boldsymbol{\mu}' - \mathbf{y}')$$

In the end we try to minimze

$$\phi(\hat{\boldsymbol{\mu}}) = (R'\hat{\boldsymbol{\mu}} - \mathbf{y}')^T (R'\hat{\boldsymbol{\mu}} - \mathbf{y}') + \tau (C\hat{\boldsymbol{\mu}})^T (C\hat{\boldsymbol{\mu}})$$

Our problem becomes

$$\begin{bmatrix} R' \\ \sqrt{\tau} \cdot C \end{bmatrix} \hat{\boldsymbol{\mu}} = \begin{bmatrix} \mathbf{y}' \\ \mathbf{0} \end{bmatrix}$$

$$C = \begin{pmatrix} -1 & 1 & 0 & 0 & \dots & \\ 1 & -2 & 1 & 0 & \dots & \\ 0 & 1 & -2 & 1 & \dots & \\ & \dots & & & \dots & \\ & \dots & & 1 & -2 & 1 \\ & \dots & & & 1 & -1 \end{pmatrix}$$

REGULARISED UNFOLDING

$$\begin{bmatrix} R' \\ \sqrt{\tau}\cdot C \end{bmatrix} \hat{\boldsymbol{\mu}} = \begin{bmatrix} \mathbf{y}' \\ \mathbf{0} \end{bmatrix} \quad \Rightarrow \quad \begin{bmatrix} R'C^{-1} \\ \sqrt{\tau}\cdot \mathbb{1} \end{bmatrix} \hat{\boldsymbol{\mu}} = \begin{bmatrix} \mathbf{y}' \\ \mathbf{0} \end{bmatrix}$$

(improves stability)

We start from τ=0, apply the SVD on the "main" matrix and solve the problem just as we did before: $R'C^{-1} = USV^T$

$$s_i z_i = d_i \quad \Rightarrow \quad \hat{\boldsymbol{\mu}} = C^{-1} V \mathbf{z}$$

and we get the same "standard" solution, with its own problems. "Rimettiamoci l'accendino"...

$$d_i \to d_i \frac{s_i^2}{s_i^2+\tau} \quad , \qquad z_i = \frac{d_i s_i}{s_i^2+\tau}$$

The covariance matrix for z_i is easy to compute

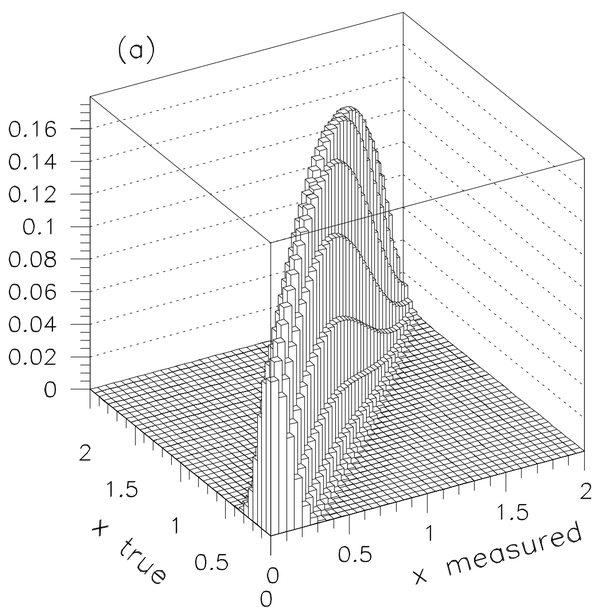
$$Z_{ik} = \frac{s_i^2}{(s_i^2+\tau)}\delta_{ik} \quad \Rightarrow \quad V_\mu = C^{-1} V Z V^T (C^{-1})^T$$

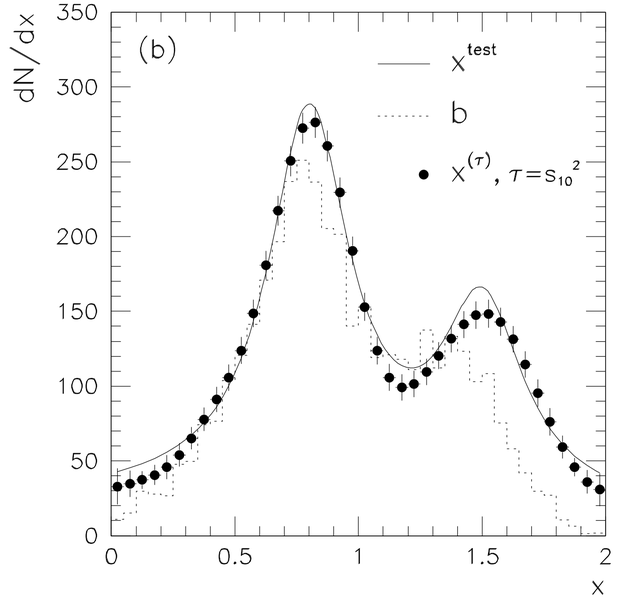
what we obtained is that now contribution to the solution from the eigenvectors of R are weighted by a term that prefers low curvatures

SVD REGULARISED

(from http://people.csail.mit.edu/hasinoff/320/SingularValueDecomposition.pdf)

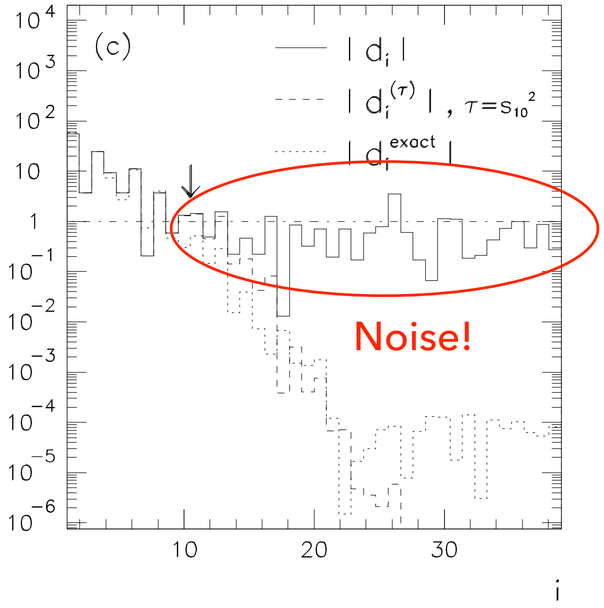
Figure 1: **a).** The probability matrix $\hat{A}$ corresponding to the response function (58). **b).** The true distribution (59) (solid curve) compared to the measured histogram b and the unfolded distribution $x^{(\tau)}$ for $\tau = s_{10}^2$. **c).** The absolute values of d_i (solid line) compared to the regularized r.h.s. (dashed line) and the one unaffected by the statistical fluctuations (dotted line). The horizontal line shows statistical errors in d_i, while the arrow indicates the boundary between the significant and non-significant equations. **d).** The deviation of the unfolded distribution from the true exact one (see text for details).

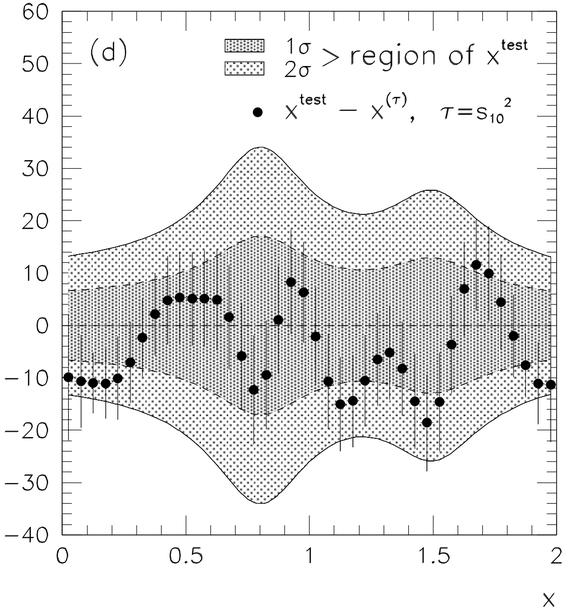




From the d_i plot it's easy to see which eigenvector of R contributes to fluctuations and adding noise to the standard solution!

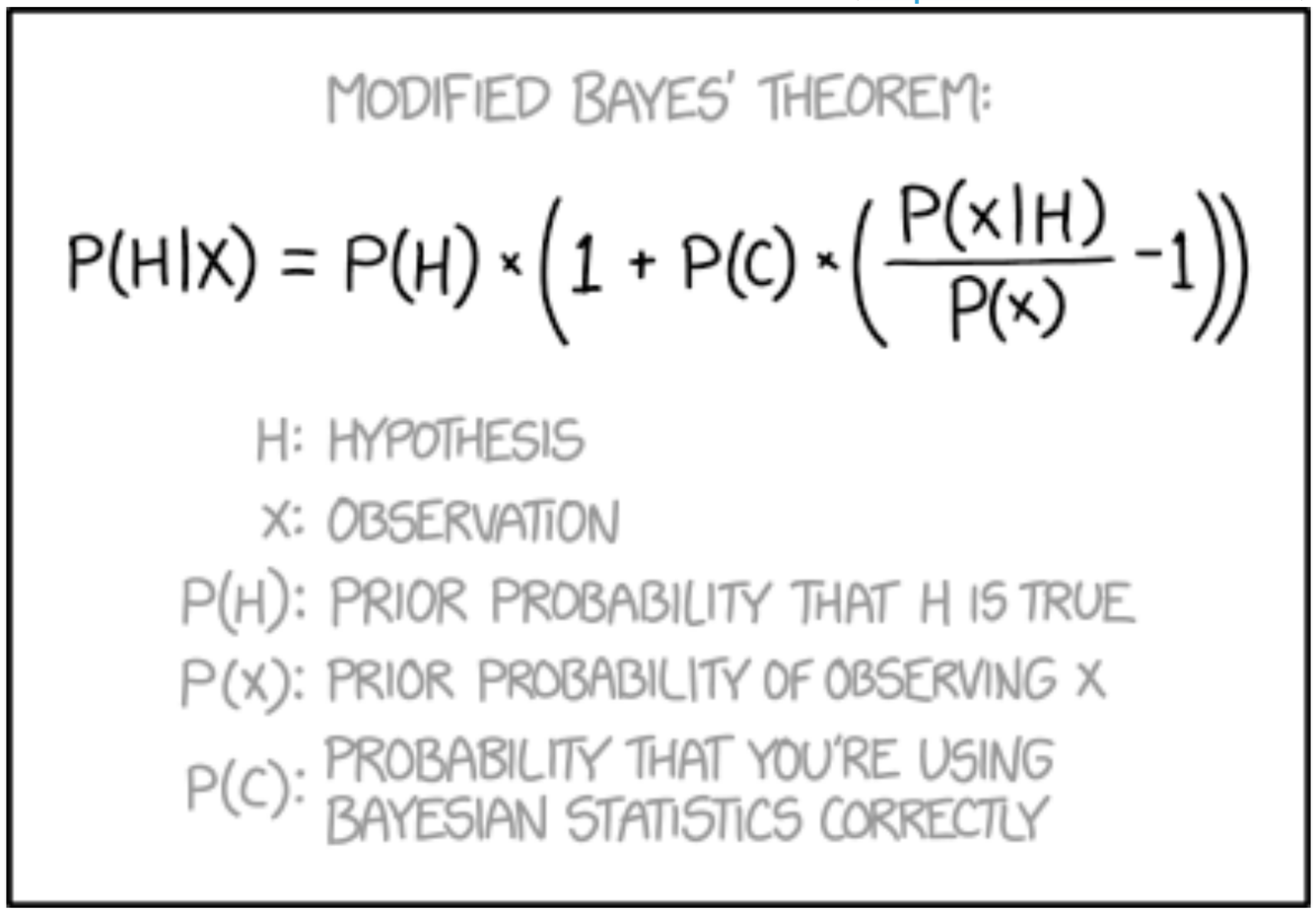
Regularization aims to suppress the contribution from these noisy eigenvectors





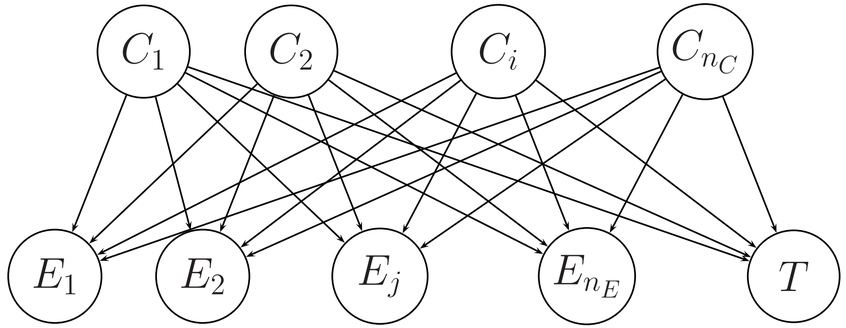
BAYESIAN UNFOLDING

(https://xkcd.com/2059/)



BAYESIAN UNFOLDING

Another way of visualizing the resolution matrix is as a link between all "causes" and possible "effects".



If we do so, we can apply Bayes' theorem

resolution matrix

prior

posterior

$$\mathbb{P}(C_i|E_j) = \frac{\mathbb{P}(E_j|C_i) \cdot \mathbb{P}(C_i)}{\sum_k \mathbb{P}(E_j|C_k) \cdot \mathbb{P}(C_k)}$$

BAYESIAN UNFOLDING

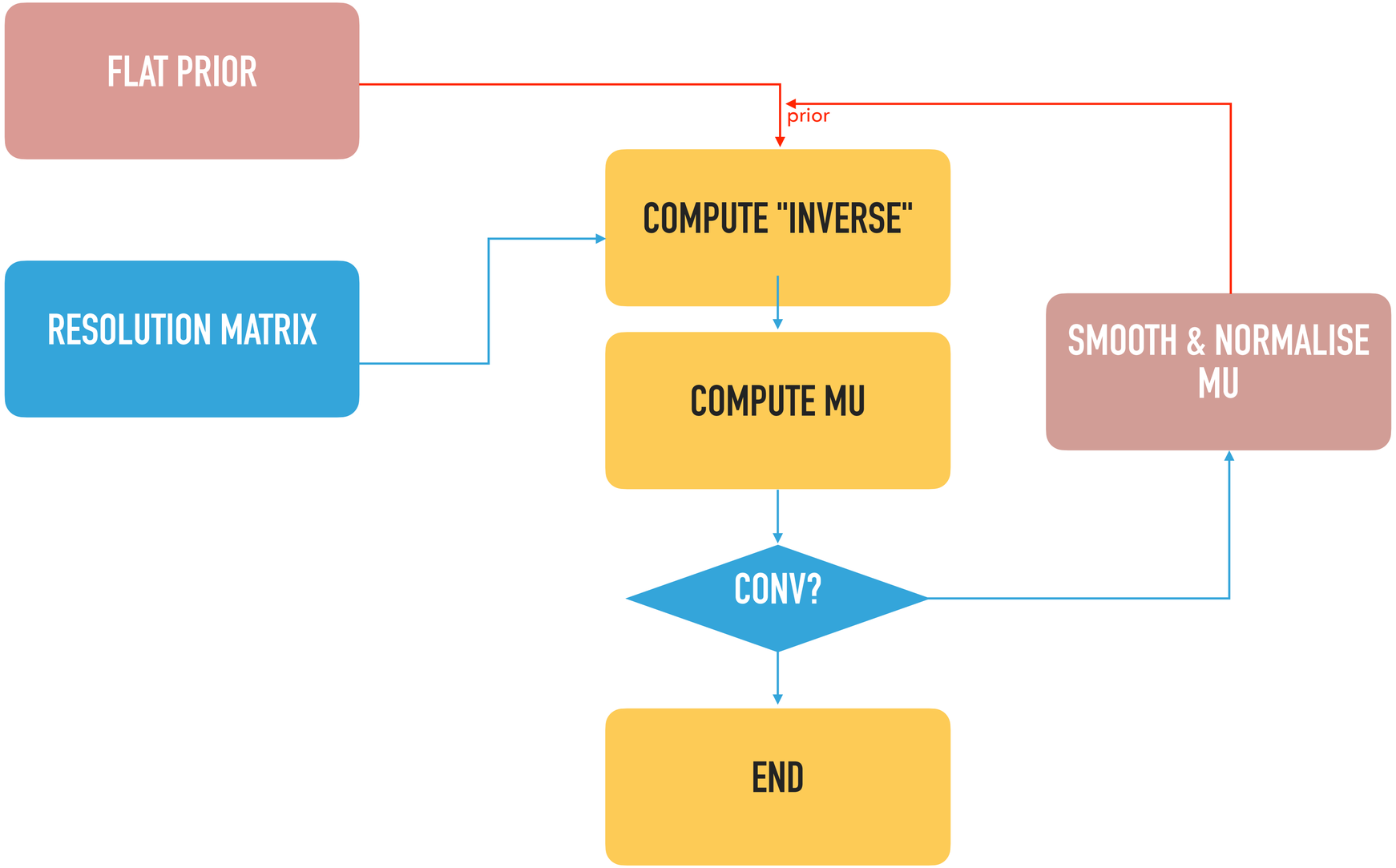
Or, in our own notation...

$$\mu_i = \frac{1}{\varepsilon_i} \sum_{j=1}^{N} \mathbb{P}(C_i|E_j)(n_j - \beta_j) = \frac{1}{\varepsilon_i} \sum_{j=1}^{N} \theta_{ij}(n_j - \beta_j)$$

$$\varepsilon_i = \sum_{j=1}^{N} \mathbb{P}(E_j|C_i) = \sum_{j=1}^{N} \lambda_{ij}$$

$$\lambda_{ij} = \left.\frac{\nu_j}{\mu_i}\right|_{\mathrm{MC}} \longrightarrow \theta_{ij} = \frac{\lambda_{ji}\mathbb{P}(C_i)}{\sum_k \lambda_{jk}\mathbb{P}(C_k)}$$

BAYESIAN UNFOLDING



BAYESIAN UNFOLDING: ERROR ESTIMATION

The original proposal estimated errors using several approximations and propagating the variance.

An improved version followed a few years later. Let's start from the resolution matrix

$$\boldsymbol{\lambda}_i = (\lambda_{i1}, \ldots, \lambda_{iN}) \sim \mathrm{Dir}\left[\boldsymbol{\alpha}_{\mathrm{prior}} + \boldsymbol{\nu}|_{\mu_i}^{\mathrm{MC}}\right]$$

since we obtained it from a MC simulation, it is affected by some statistical fluctuations. We can extract several realizations of the sam
matrix by using its dual distribution (Dirichlet).

For each extraction of the resolution matrix, we compute the "bayesan inverse" $\boldsymbol{\theta}_i = (\theta_{i1}, \ldots, \theta_{iN})$

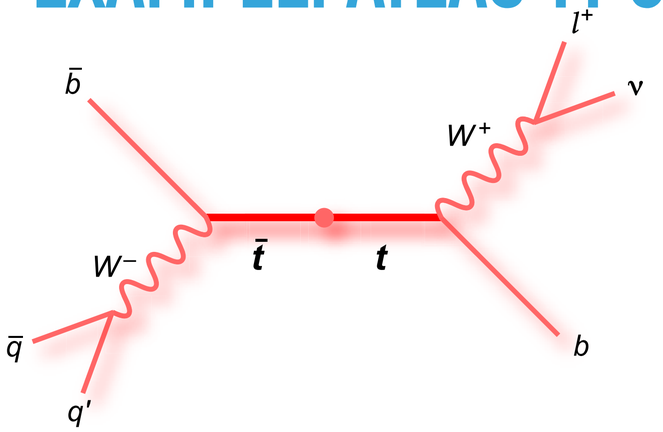
Events will then be assigned into the "cause" bins using a multinomial distribution which depends on the "inverse" matrix

$$\boldsymbol{\mu}|_{\nu_j} \sim \mathrm{Mult}\left[\nu_j, \boldsymbol{\theta}_j\right] \quad \Rightarrow \quad \boldsymbol{\mu} = \sum_{j=1}^{N} \boldsymbol{\mu}|_{\nu_j}$$

Rinse and repeat

EXAMPLE: ATLAS TT CHARGE ASIMMETRY

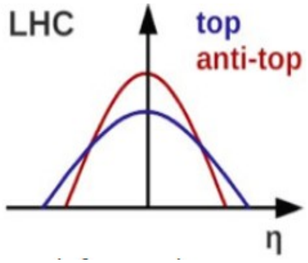
with ATLAS @ LHC



$\int L dt = \mathbf{1\ fb^{-1}}$ *(2011)*

- Stop when A_C changes by less than 0.1% on MC
- Stat uncertainty checked with pseudoexpriments
- Syst uncertainty propagated to response matrix and bkg
- Re-weight $t\bar{t}$ events to vary A_C and check unfolding linearity.

- Expect

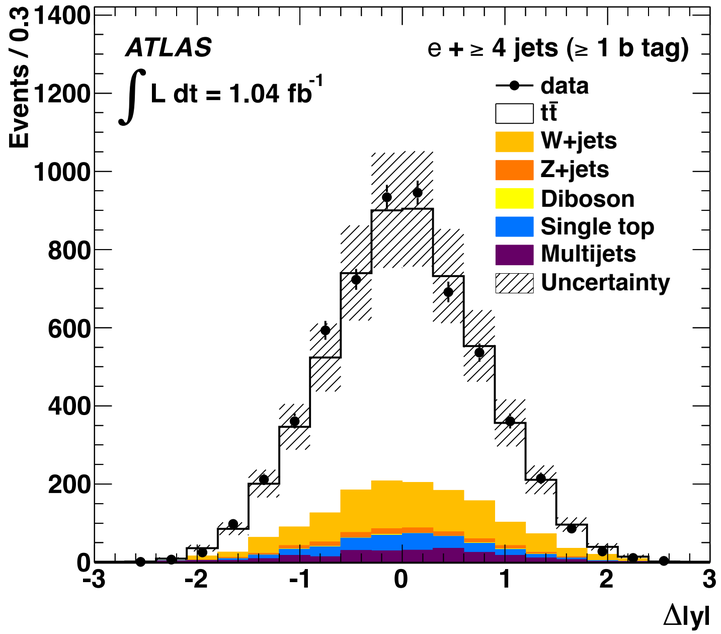


MC@NLO@ 7TeV LHC predicts A_C = 0.006 +/- 0.002

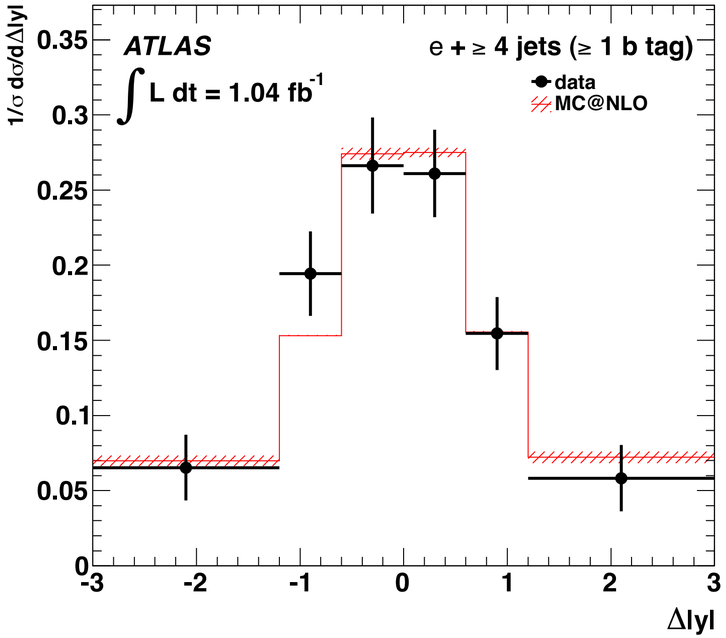
- Reconstruct $t\bar{t}$ and study

$$A_C = \frac{N(\Delta|Y| > 0) - N(\Delta|Y| < 0)}{N(\Delta|Y| > 0) + N(\Delta|Y| < 0)}$$

arxiv:1203.4211[hep-ex]



unfolded



ITERATIVE CORRECTION FACTOR

Used by AMS-02 in measuring CR nuclei fluxes

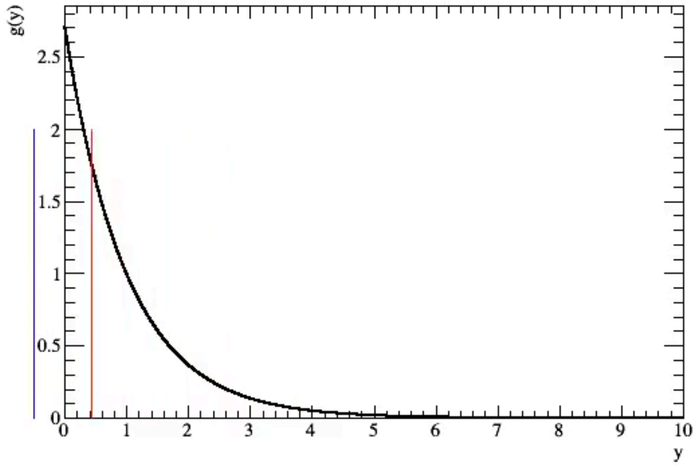
$$\mu_i = C_i(n_i - \beta_i)$$

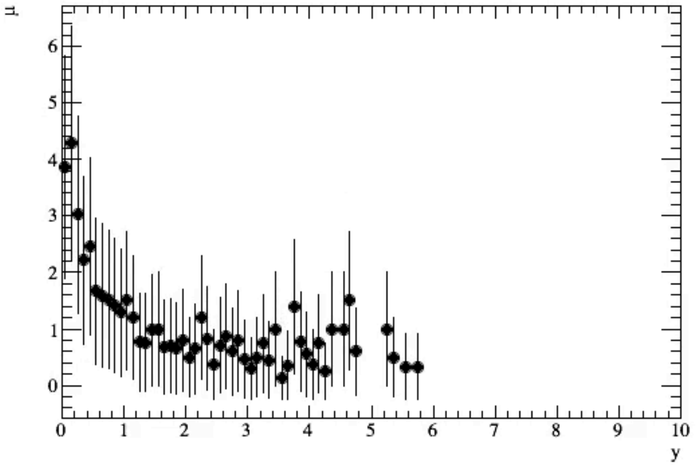
Start from a given initial distribution $g_0(y)$. Simulate events, and fill histogram both with the "true" variable, and the "measured" one.

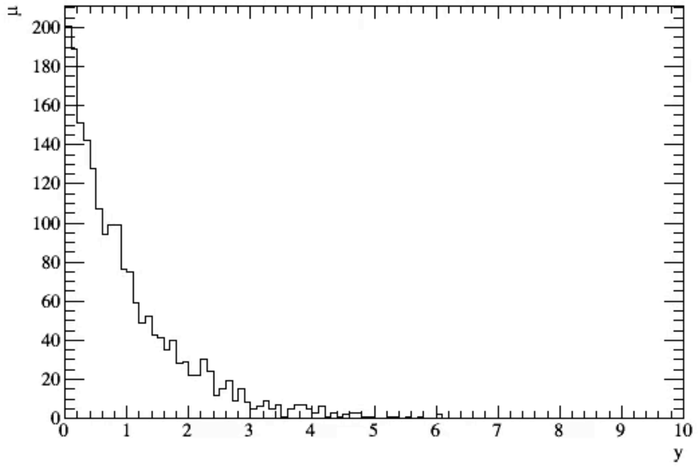
$$C_i = \mu_i^{\mathrm{MC}} / \nu_i^{\mathrm{MC}}$$

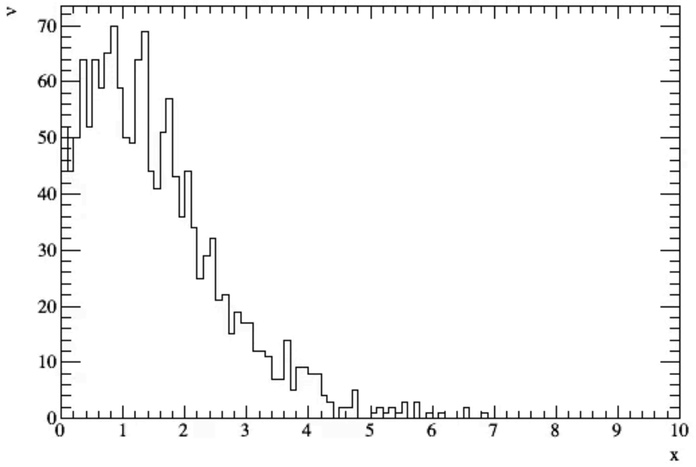
$$\mu_i^{\mathrm{MC}} = \sum_{j=1}^{N_{\mathrm{ev}}} g_0(y_i)\Pi\left(\frac{y_i - \tilde{y}_i}{\Delta y_i}\right)$$

$$\nu_i^{\mathrm{MC}} = \sum_{j=1}^{N_{\mathrm{ev}}} g_0(y_i)\Pi\left(\frac{x_i - \tilde{x}_i}{\Delta x_i}\right)$$









ITERATIVE CORRECTION FACTOR

$$\mu_i^0 = C_i^0(n_i - \beta_i) \quad , \quad C_i^0 = \mu_i^{\mathrm{MC}}/\nu_i^{\mathrm{MC}}$$

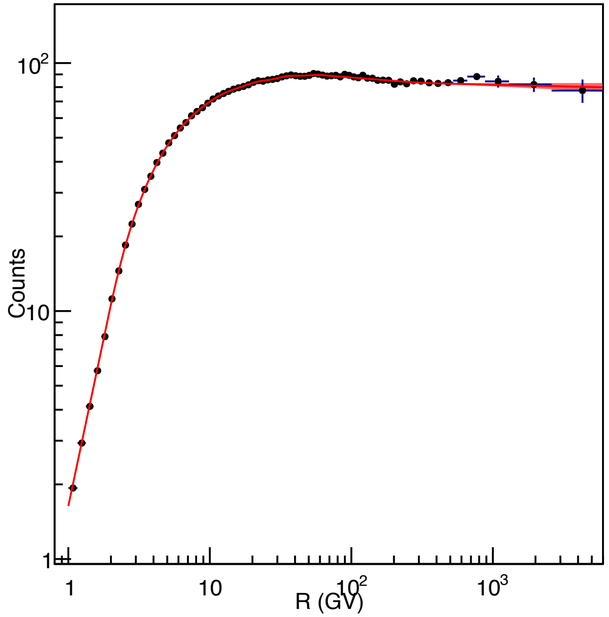
Define a new weighting function

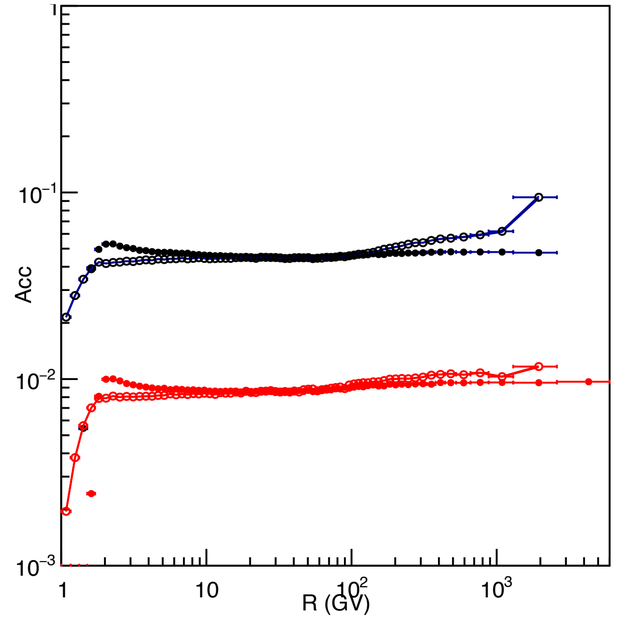
$$g_1(y) = f(\tilde{\boldsymbol{\mu}}) \quad , \quad \tilde{\mu}_i = \frac{\mu_i}{\sum_{j=1}^{N} \mu_j}$$

such as it is a sufficiently smooth representation of the step 0 result.
We use this to compute the correction factor.

$$\mu_i^{\mathrm{MC}} = \sum_{j=1}^{N_{\mathrm{ev}}} g_1(y_i)\Pi\left(\frac{y_i - \tilde{y}_i}{\Delta y_i}\right) \quad , \quad \nu_i^{\mathrm{MC}} = \sum_{j=1}^{N_{\mathrm{ev}}} g_1(y_i)\Pi\left(\frac{x_i - \tilde{x}_i}{\Delta x_i}\right)$$

and we repeat the procedure until the difference between two consecutive iterations is below a given threshold.





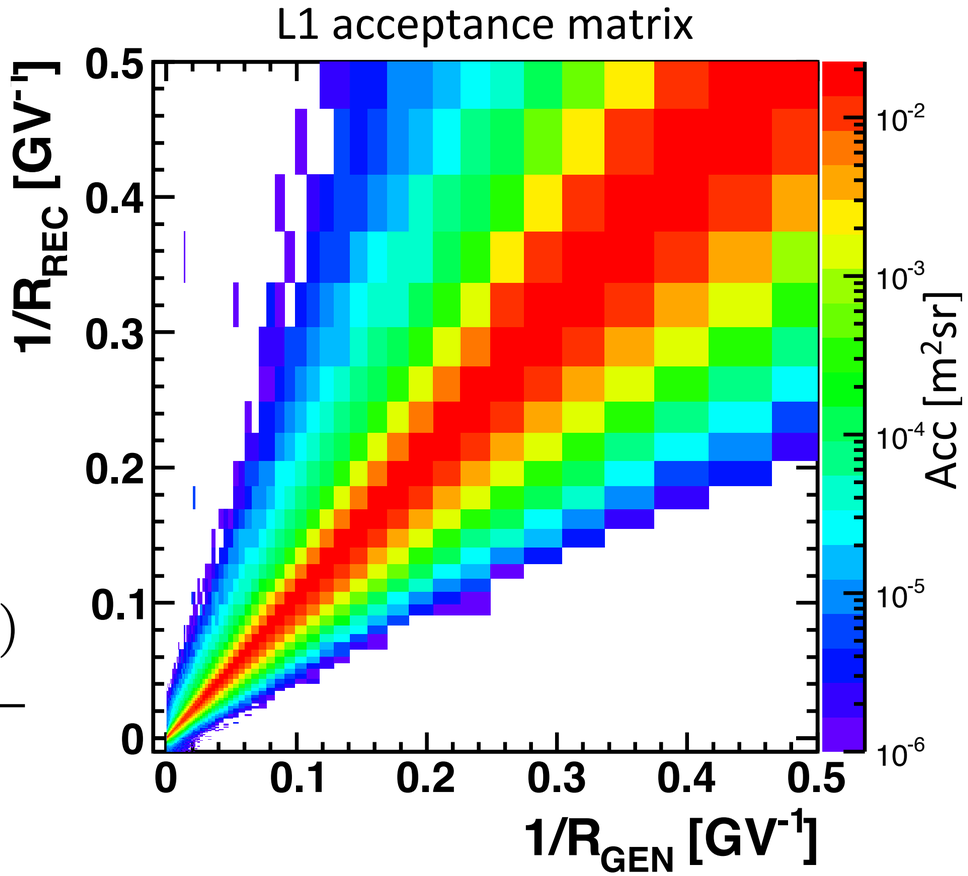
FORWARD FOLDING

If you can write down an ansatz for the target distribution you can try to define a likelihood such as:

$$\hat{\mu}_i(\boldsymbol{\theta}) = \int_{y_i}^{y_{i+1}} f(y;\boldsymbol{\theta})\, dy$$

$$\hat{\nu}_i(\boldsymbol{\theta}) = \sum_{j=1}^{N} R_{ij}\hat{\mu}_j(\boldsymbol{\theta})$$

$$\mathcal{L}(\boldsymbol{\theta}|\mathbf{n}) = \prod_{i=1}^{N} \mathbb{P}(n_i|\hat{\nu}_i(\boldsymbol{\theta})) = \prod_{i=1}^{N} \hat{\nu}_i(\boldsymbol{\theta})^{n_i} \frac{e^{-\hat{\nu}_i(\boldsymbol{\theta})}}{n_i!}$$



Now the difference is in the fact that $f(y)$ is smooth by definition!

FORWARD FOLDING

If you can write down an ansatz for the target distribution you can try to define a likelihood such as:

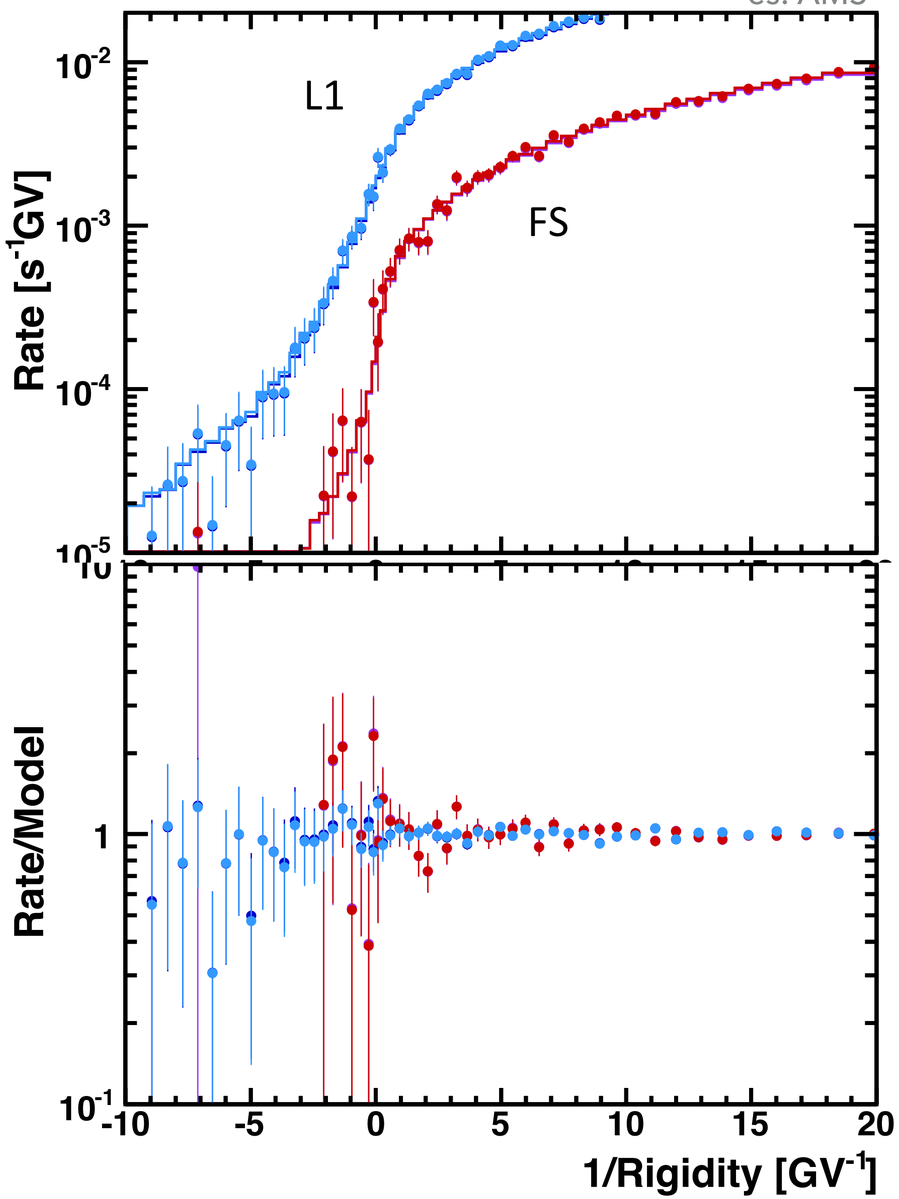
$$\hat{\mu}_i(\boldsymbol{\theta}) = \int_{y_i}^{y_{i+1}} f(y;\boldsymbol{\theta})\, dy$$

$$\hat{\nu}_i(\boldsymbol{\theta}) = \sum_{j=1}^{N} R_{ij}\hat{\mu}_j(\boldsymbol{\theta})$$

$$\mathcal{L}(\boldsymbol{\theta}|\mathbf{n}) = \prod_{i=1}^{N} \mathbb{P}(n_i|\hat{\nu}_i(\boldsymbol{\theta})) = \prod_{i=1}^{N} \hat{\nu}_i(\boldsymbol{\theta})^{n_i} \frac{e^{-\hat{\nu}_i(\boldsymbol{\theta})}}{n_i!}$$

Now the difference is in the fact that $f(y)$ is smooth by definition!

es: AMS



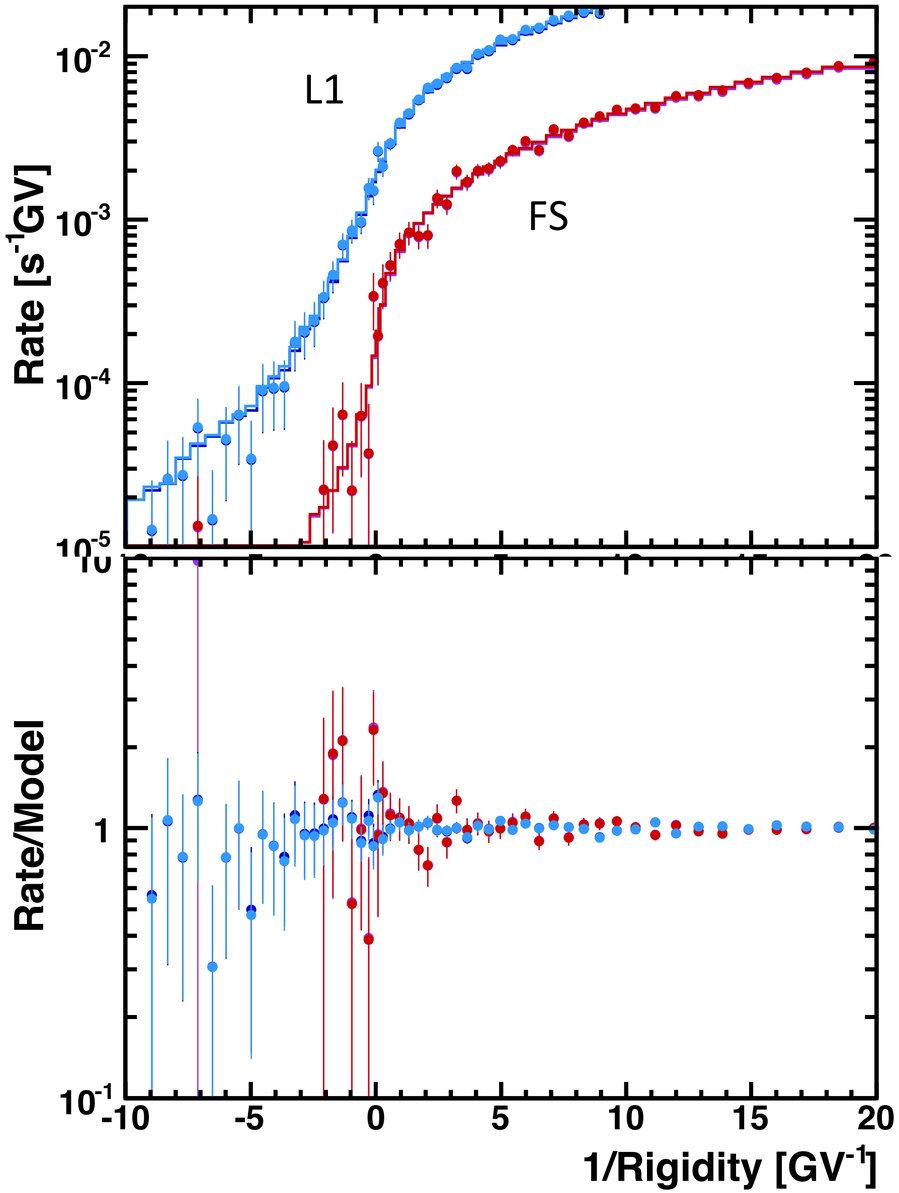
FORWARD FOLDING

$$\mathcal{L}(\boldsymbol{\theta}|\mathbf{n}) = \prod_{i=1}^{N} \mathbb{P}(n_i|\hat{\nu}_i(\boldsymbol{\theta})) = \prod_{i=1}^{N} \hat{\nu}_i(\boldsymbol{\theta})^{n_i} \frac{e^{-\hat{\nu}_i(\boldsymbol{\theta})}}{n_i!}$$

Of course there are some caveat:

- This works only if the underlying distributions are naturally smooth.
- If you can define a smooth function whose integral is directly tied to event counts you're all set. Generally speaking every transform of the p.d.f. breaks the assumption of poisson counts
  (e.g.: in AMS we use the count rate as input function. The count rate is obtained dividing the counts by the geomagnetic transmission function -> the integral of the count rate in a given bin, no longer follows poisson statistics!)
- Hard t oget uncertainty on the result
  (toy MC, error propagation on function parameters; not a trivial problem, but solvable)
- Getting minimization to converge can be hard, depending on the problem.

es: AMS



MORE EXAMPLES...

- IDS: iterative dynamically stabilized, B. Malaescu, arxiv: 0907.3791 [phys.data-an]
  - used in ATLAS paper http://arxiv.org/abs/1112.6297
- Binning free Iterative Deconvolution, Lindemann, Zech, Nucl.Instr. Meth A 354 (1995) 516-521
- Satellite Method, see G. Bohm and G. Zech, Introduction to Statistics and Data Analysis for Physicists, Verlag Deutsches Elektronen-Synchrotron (2010), available at http://www-library.desy.de/elbook.html
- SPlot, M Pivk, F. Le Diberder, arXiv:physics/0402083v

References:

- Bayes
  - https://www.roma1.infn.it/~dagos/199408011.pdf (original)
    https://arxiv.org/pdf/1010.0632.pdf (improved)
- SVD
  - https://arxiv.org/pdf/hep-ph/9509307.pdf

THE EXERCISE

You get two datasets:

A sample of cosmic-ray events

- For each event you have a measurement of its rigidity (= momentum/charge)
- You have a MC simulation sample with 5x more events where for each event you have both "generated" and "measured" rigidity
- You should unfold the rigidity distribution and reconstruct the flux (a histogram with the geomagnetic transmission function is also provided)
- Fit the spectrum with the function

$$\Phi = C\left(\frac{R}{45\ \mathrm{GV}}\right)^{\gamma}\left[1+\left(\frac{R}{R_0}\right)^{\Delta\gamma/s}\right]^{s},$$

  and measure both the spectral index and its change

A sample of muon pairs from some beam collisions

- For each event you have the 4-momentum of each muon
- You have a MC simulation sample with 10x where for each event you have both "generated" and "measured" 4-momenta
- You should unfold the invariant mass distribution
- Fit the peak in the mass distribution with a gaussian function (beware of the background) and measure its position, its width, and the total number of events associated to it.

Choose **TWO METHODS** from: Iterative correction factor, Regularized SVD, Bayesian (bonus points if improved), Forward folding, and compare the results from each method

THE EXERCISE

You can get the data (and these slides) from here: https://github.com/valerioformato/statistical-data-analysis-unfolding-exercise

In each dataset folder you also have a file with some setup code to get the trees and setup the branches.

Some tips and rules:

- Don't use RooUnfold. The goal of this exercise is for you to try and implement two unfolding procedures. Existing libraries and tools are good and fine, but won't teach you anything.
- You can compute the invariant mass by hand, or you can store the muon momenta in some TLorentzVector objects and use those to compute the invariant mass.
- If you choose to implement SVD unfolding, you can use the class TDecompSVD to help you with the decomposition. Also, follow the paper, the steps are not that hard once you understand the procedure.
- If you choose to implement Forward Folding you need to write a Likelihood function and hook it to a ROOT::Math::Minimizer object, you might need some help with this so feel free to come and ask :)