Improved Bayesian unfolding

G. D'Agostini
Università "La Sapienza" and INFN, Rome, Italy
(giulio.dagostini@roma1.infn.it, `http://www.roma1.infn.it/~dagos`)

Abstract

This paper reviews the basic ideas behind a Bayesian unfolding published some years ago and improves their implementation. In particular, uncertainties are treated at all levels by probability density functions and their propagation is performed by Monte Carlo integration. Thus, small numbers are safely handled and the final uncertainty does not rely on the assumption of normality. Theoretical and practical issues concerning the iterative use of the algorithm are also treated. The new program, implemented in the R language, is freely available, together with tests scripts to play with toy models.

"Einmal ist keinmal"

1 Introduction

Physicists like to think of 'true' distributions of physics quantities, i.e those distributions (graphically represented by histograms and here also called 'spectra') one would observe under idealized conditions that seldom — never, strictly speaking — happen in real live (ideal detector, no physical or instrumental background). The 'observed' distribution is then seen as a 'noisy distortion' of the true one. An important task of the experimental method is then to estimate the true distribution from the observed one, i.e. to correct the observed spectrum for distortion and noise. This can be done by different methods that follow different approaches. Since this is not a review paper, I just outline the two different classes of strategies and then focus on the specific issues.

In a first kind of approach a mathematical function for the true distribution (and the noise) is assumed, and the task becomes that of estimating the free parameters of the function(s). Indeed, we fall in the so called domain of *parametric inference* ('fits'), usually associated to names like 'least-squares' or 'maximum likelihood'.[1]

Parametric inference, especially when the conditions to use least-squares methods hold, is certainly the way to prefer, if we have good reasons to believe the hypothesized family of functions. In parametric inference all information contained in the observed spectrum is, to say, 'distilled' into the model parameters. However, sometimes we wish to interpret the data as little as possible and just public 'something equivalent' to an experimental distribution, with the bin contents fluctuating according to an underlying multinomial distribution, but having possibly got rid of physical and instrumental distortions, as well as of background. In particle physics this second approach goes under the name of *unfolding* (or deconvolution, or restoration).

Several years ago a simple algorithm based on Bayes' theorem was presented [2] with which it was possible to achieve rather 'good' results ('good' compared with the difficulty of the task). The main improvements presented here concern the handling of small numbers and the evaluation of the uncertainty on the unfolded distribution, while the guiding ideas and the basic assumptions are unchanged. To be more clear, the algorithm of Ref. [2] was relying on the underlying hypotheses of normality and 'small relative errors': 'best estimates' were provided, with uncertainty calculated from standard 'error propagation' formulas.[2] The new algorithm handles better small numbers, in the way it will be described in Sec. 3.2, and performs the propagation of uncertainty by sampling, i.e. by Monte Carlo (MC) integration. The algorithm has been implemented in a R language [4] code available on the author's URL [5].

The paper is organized in the following way. Section 2 gives a short introduction to Bayesian inference and to the specific issue of unfolding. Then the algorithm of Ref. [2] is reminded and the improvements are presented.

[1]However one can show that it is preferable to base the data analysis on more solid probabilistic ground, from which the mentioned methods might also result (see e.g. Ref. [1]), but only under particular conditions, which fortunately hold in the large majority of practical cases (but it is better one knows when the conditions of validity hold and what to do when they do not!).

[2]It is worth remembering that these formulas rely on an hypothesis of linearity between *input* and *output quantities*, an hypothesis that holds approximately, in non-strictly linear problems, if the uncertainties of the input quantities are small enough, such that the dependence input$\rightarrow$output is locally linearized [3].

The issue of iterative use of the algorithm is also discussed, although the program now gives also the possibility to use priors, provided by the user, over all possible true spectra. This option should avoid the need of iterations (but I anticipate that it might be not that easy to model such priors and then the iterative strategy remains a pragmatic solution). Finally, some results on toy models are presented.

2 Bayesian inference and Bayesian unfolding: from first principles to real life

The so called Bayesian inference is a way to learn about physical quantities and their relationships (things we cannot directly see) from experimental data (what we can actually observe with our senses, usually mediated by more or less sophisticated detectors) using probability theory. This game is *conceptually* rather simple, proviso we accept that the intuitive meaning of probability — a scale to rank our beliefs that several events might happen, or that several hypotheses might be true — is suitable in scientific reasoning (see Refs. [6] and [7] and references therein for an introduction).

The starting point of a Bayesian inference is to build up a model of the deterministic ("*from this follows that*") and probabilistic ("*from this might follow that*") connections that relate the several entities that enter the problem. This model has the interesting graphical representation of a network, usually called *Bayesian network* or *belief network* (to get an idea of their meaning and their application *google* these keywords, or browse the *Wikipedia*). For example, Fig. 1, taken from Ref. [1], shows a Bayesian network to model two physical quantities μ_x and μ_y connected to each other with a 'law', whose parameters are denoted by $\boldsymbol{\theta}$. In the very elementary case depicted in the left hand network of Fig. 1 the solution to the problem can be rather simple, under some assumptions that, fortunately, hold very often in routine cases. But, in general, the solution is not that simple. Nevertheless, as stressed in Ref. [1], after one has built the graphical model, one is often more than half the way to the solution, thanks to the great progresses made recently in Bayesian network computing (see e.g. Ref. [8] and references therein). However, the problem of unfolding, in the very general terms as introduced in Ref. [2], does not fit easily in a straight Bayesian network methodology, for the reasons we shall see in a while, and some 'tricks' are required.

The essence of the Bayesian unfolding of Ref. [2], that is the starting point also of this new version, is to make the problem discrete and to treat

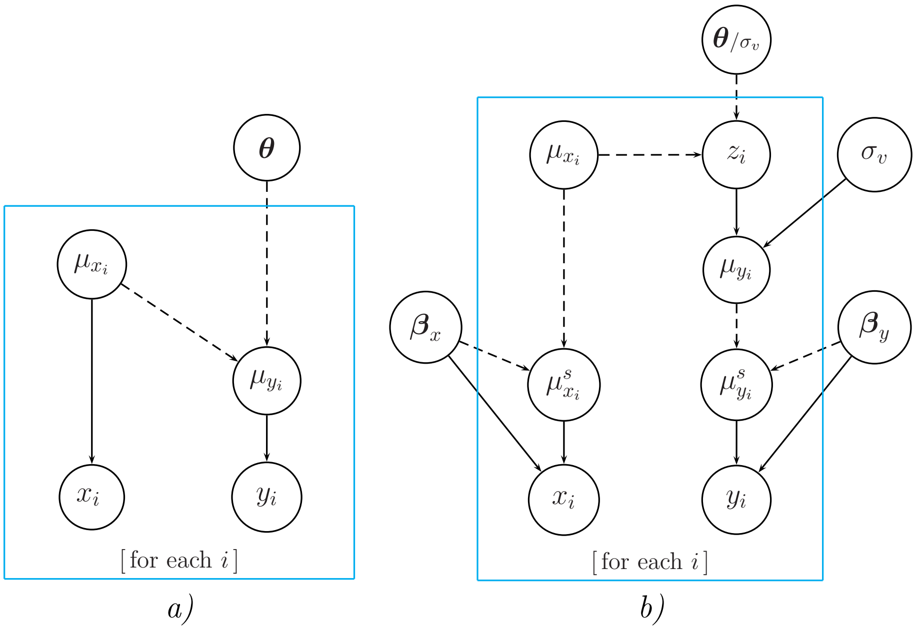

Figure 1: *Bayesian network describing a fit model [1]. x and y are the experimental observations, related to μ_x and μ_y by experimental errors. Instead, a deterministic 'law' connects the 'true' values μ_y to μ_x via the model parameters $\boldsymbol{\theta}$ (solid arrows stand for probabilistic links, dashed for deterministic). Network a) describes a simple model with errors only on the ordinate. Network b) takes into account errors on both axes, extra variability of the data points around the believed physical low and systematic effects.*

the 'cause' bins as independent degrees of freedom, i.e. without constraints among each other. For this reason the algorithm can virtually handle any kind of 'smearing' and it is easily extensible, at least in principle and with the only limit due to computer power, to multidimensional problems. In fact, the core of the algorithm only knows about 'cause-cells' and 'effect-cells',[3] but it does not know the location of the cells in the configuration space of the problem. For the same reason, the treatment of background, and even of several independent sources of background, can be easily embodied in the algorithm by just adding extra cause-cells, one cell per source of background. As a by-product, the algorithm also provides the number

[3]Here 'bin' and 'cell' are used as synonyms, although 'cell' refers to a region on the configuration space and 'bin' to histograms representing them.

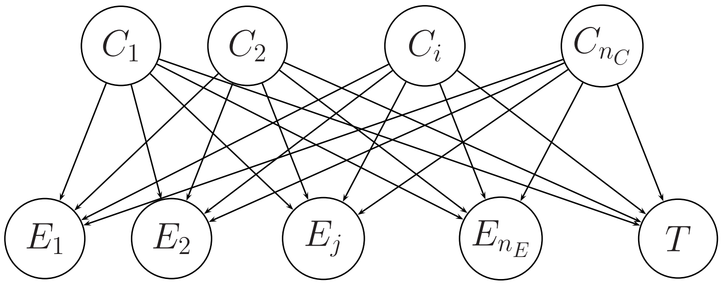

Figure 2: *Probabilistic links from causes to effects. The node indicated by T ('trash') stand for the inefficiency bin and corresponds to E_{n_E+1}*

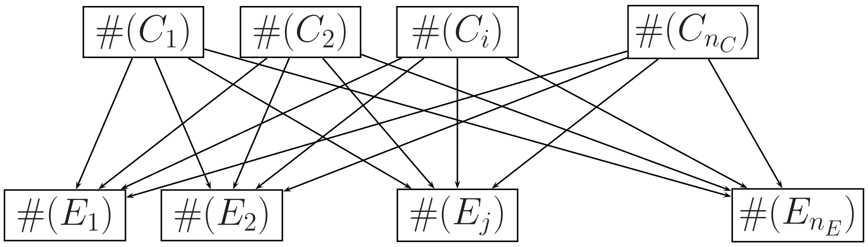

Figure 3: From the 'true' distribution (numbers of events in the cause-bins) to the observed distribution (numbers of events in the effect-bins). The number of events $\#(\,)$ is indicated with $x(\,)$ in the text.

of events to be attributed to each source of background — it is worth remembering that background might have an interesting physical meaning, and thus the estimation of the level 'noise' might provides indeed a physics measurement, as in the analysis of Ref. [9].

Given the discretization of the algorithm, the Bayesian network relating causes and effects is that shown in Fig. 2, where we use the same notation of Ref. [2], with the addition of the effect bin T ('trash'), equivalent to E_{n_E+1}, to describe inefficiency (the reason to introduce this extra bin will become clear later).

Rephrasing the problem in probabilistic terms, the purpose of the unfolding is to find the 'true' number of events in each cause bin [$\#(C_i)$ in Fig. 3, indicated by $x(C_i)$ in the text], given the observed spectrum and assuming some knowledge about the smearing.

Because of the probabilistic links cause→effects, also the links effect→causes will be probabilistic, and therefore it will be uncertain the number of events to be attributed to the cause-cells. We can only attempt to rank in probability all possible spectra that might have caused the observed one. In other words, the realistic goal of our analysis is not to determine the *the* true spectrum, but rather to assess

$$P(\boldsymbol{x}_C \,|\, \boldsymbol{x}_E,\, \Lambda,\, I)\,, \tag{1}$$

where:[4]

- $\boldsymbol{x}_C = \{x(C_1),\, x(C_2),\, \ldots,\, x(C_{n_C})\}$ is the number of events in each bin of the true distribution, i.e. *a* true spectrum.
- $\boldsymbol{x}_E = \{x(E_1),\, x(E_2),\, \ldots,\, x(E_{n_E})\}$ is the observed spectrum.
- Λ stands for the smearing matrix, whose elements λ_{ji} (see remarks in footnote 4 concerning symbols) are defined in probabilistic terms as

  $$\lambda_{ji} \;\equiv\; P(E_j \,|\, C_i,\, I)\,. \tag{2}$$

  The knowledge of Λ comes from MC simulation and it is therefore affected by uncertainty, an uncertainty modeled in general by a pdf $f(\Lambda \,|\, \mathrm{MC},\, I)$.
- I stands for the state of information under which the analysis is performed (this underlying condition is often implicit in the pdf's).

Once we have stated clearly and in probabilistic terms our question (*"what is $P(\boldsymbol{x}_C \,|\, \boldsymbol{x}_E,\, I)$?"*), probability theory provides us the answer,[5] at least in principle. In fact,

[4] In Ref. [2] $\boldsymbol{x}_C$, $\boldsymbol{x}_E$ and Λ were respectively indicated by $\underline{n}_C$, $\underline{n}_E$ and $\boldsymbol{S}$. As in Fig. 3 and in Ref. [2], the symbols i and j are used to index causes and effects, respectively, no matter if this choice leads to the unusual convention of indexing the Λ rows by j and the columns by i, as in Eq. (2). Note that, when we refers to MC simulations to infer the smearing matrix, $\boldsymbol{x}_E$, we also need include the trash bin, as it will be reminded at the proper place.

[5] Instead, the idea of inverting the smearing matrix (assuming it square and not singular) is wrong in principle (i.e. besides the ascertainment that such a method yields unacceptable results). In fact unfolding (or at least what it is meant by this word in experimental physics — folding and unfolding might have other meanings in other fields) is a probabilistic problem and not a deterministic linear one (or, equivalently, a geometric problem of rotating vectors) that needs to be solved by probabilistic tools and not by linear algebra methods ('rotations'). It easy to show that 'rotation' works only when we have an 'infinite' number of events, such that stochastic effects are negligible and the

1. Bayes' theorem states that, given the observation $\boldsymbol{x}_E$ and the smearing matrix Λ,

$$P(\boldsymbol{x}_C \,|\, \boldsymbol{x}_E,\, \Lambda,\, I) \;=\; \frac{P(\boldsymbol{x}_E \,|\, \boldsymbol{x}_C,\, \Lambda,\, I) \cdot P(\boldsymbol{x}_C \,|\, I)}{\sum_{\boldsymbol{x}_C} P(\boldsymbol{x}_E \,|\, \boldsymbol{x}_C,\, \Lambda,\, I) \cdot P(\boldsymbol{x}_C \,|\, I)} \tag{3}$$

   (the formula will be explained in a while).

2. We can take into account of the uncertainty about Λ using another theorem of probability theory, namely

$$P(\boldsymbol{x}_C \,|\, \boldsymbol{x}_E, I) \;=\; \int P(\boldsymbol{x}_C \,|\, \boldsymbol{x}_E, \Lambda,\, I)\, f(\Lambda \,|\, \mathrm{MC}, I)\, \mathrm{d}\Lambda\,. \tag{4}$$

Let us now go through the several ingredients needed to get $P(\boldsymbol{x}_C \,|\, \boldsymbol{x}_E, I)$ and try to understand were the problems arise.

---

observations coincides with the expectations. In fact, each product $\lambda_{ji}\, x(C_i)$ is nothing but the expected number of events in effect-cell E_j due to cause C_i alone. Summing up the contributions from all cause-cells, we get the expected value in effect-cell E_j

$$\begin{aligned} \mathrm{E}[x(E_j)|_{x(C_i)}] &= P(E_j \,|\, C_i,\, I)\, x(C_i) = \lambda_{ji}\, x(C_i) \\ \mathrm{E}[x(E_j)|_{\boldsymbol{x}_C}] &= \sum_i \lambda_{ji}\, x(C_i)\,, \end{aligned}$$

that can be written in matrix form as

$$\boldsymbol{\mu}_E \equiv \mathrm{E}[\boldsymbol{x}_E] \;=\; \Lambda\, \boldsymbol{x}_C\,.$$

Then, if Λ is square and not singular, we get

$$\boldsymbol{x}_C \;=\; \Lambda^{-1}\, \boldsymbol{\mu}_E\,.$$

But this might be, at most, the solution of a text book exercise in mathematical probability theory, and does not help to solve real problems. This is the reason why, besides the fact that the matrix inversion gives notoriously bad results, *the very idea of inverting the smearing matrix is logically flawed*: we can certainly apply Λ^{-1} to a vector of numbers *already known* to be sums of expected values of binomials, but we cannot apply it to a vector of numbers that *might be* (we are not even sure of this, because some counts could be due to background we do not take into account!) sums of binomial random variables. If we do it, there is no guarantee that $\Lambda^{-1}\boldsymbol{x}_E$ yields a vector of 'valid numbers' of the n-parameters of binomials (the question that they might have the meaning of a physical spectrum for the problem under study is secondary at this level) and, in fact, even negative numbers can be obtained! It follows that unfolding methods which use the matrix inversion as starting point and try to cure its bad features with some kitchen are not appealing from a theoretical point view, although they might even provide acceptable results because of 'mysterious' reasons (cooks might be extremely clever!) I do not want to enter into.

- First of all, it is easy to realize that the denominator of Eq. (3) is just a normalization factor, and then we can rewrite the Bayes' formula in a way that highlights the main ingredients:

$$P(\boldsymbol{x}_C \,|\, \boldsymbol{x}_E,\, \Lambda,\, I) \;\;\propto\;\; P(\boldsymbol{x}_E \,|\, \boldsymbol{x}_C,\, \Lambda,\, I) \cdot P(\boldsymbol{x}_C \,|\, I)\,, \tag{5}$$

  where $P(\boldsymbol{x}_E \,|\, \boldsymbol{x}_C,\, \Lambda,\, I)$ is the *likelihood* and $P(\boldsymbol{x}_C \,|\, I)$ the *prior*. The left hand side of the Bayes's formula takes the name of *posterior*. As we see, in probabilistic inference the likelihoods have the role of updating probabilities.

- Although the presence of priors in the formula might cause anxiety in those who approach this kind of reasoning for the first time, it is a matter of fact that: 1) priors are logically necessary to get $P(\boldsymbol{x}_C \,|\, \boldsymbol{x}_E,\, \Lambda,\, I)$ starting from the likelihood, i.e. to perform the so-called 'probability inversion'; 2) they allow to plug into the model all relevant prior information, that might come from previous data or from theoretical prejudices; 3) priors are often so vague (or there are so many data — that is the same for the relative balance of prior and likelihood in the Bayes' formula) that they influence negligibly the posterior, and the inference is dominated by likelihood.

- Let us assume this last case of prior vagueness applies. This is equivalent to have

$$P(\boldsymbol{x}_C \,|\, I) \;\;=\;\; \textit{constant} \tag{6}$$

  and the inference is then performed according to the rule

$$P(\boldsymbol{x}_C \,|\, \boldsymbol{x}_E,\, \Lambda,\, I) \;\;\propto\;\; P(\boldsymbol{x}_E \,|\, \boldsymbol{x}_C,\, \Lambda,\, I)\,. \tag{7}$$

  It is then not a surprise that the most probable spectrum $\boldsymbol{x}_C$ is the one which maximizes the likelihood, *if* we have no other good reason to believe that this is not the case. (This is the meaning of the expression "recovering maximum likelihood estimators" from the Bayesian approach.)

- If we were able to give a closed expression for the likelihood, or even write a simple algorithmic expression of it, we could somehow scan all possible spectra $\boldsymbol{x}_C$, find the most probable one (or an 'average' spectrum that summarizes in some sense the variety of true spectra compatible with the data) and assess somehow the uncertainty about the result. But the technical problem comes from the fact that it is not possible to write with an analytic formula, or in a way that can be simply handled by computers, $P(\boldsymbol{x}_E \,|\, \boldsymbol{x}_C,\, \Lambda,\, I)$. Let us see why. Given a certain number of events in a cause-bin $x(C_i)$, the number of events in the effect-bins, included the 'trash' one, is described by a *multinomial distribution* (see Appendix A):

$$P(\boldsymbol{x}_E \,|\, x(C_i), \Lambda, I) \;=\; \frac{x(C_i)!}{\prod_j^{n_E+1} x(E_j)!} \prod_j^{n_E+1} \lambda_{ji}^{x(E_j)} \,. \qquad (8)$$

It follows that $P(\boldsymbol{x}_E \,|\, \boldsymbol{x}_C,\, \Lambda,\, I)$ is the sum of independent multinomial distributions, for which, unfortunately, a closed formula does not exists.[6] This is the real serious technical problem that prevents a straight application of the Bayes' formula, not even by numerical methods, to be used.

- The elements of the smearing matrix Λ are obtained by MC: we generate a large number of events in each cell C_i and count where 'they end' after a realistic simulation. Intuitively, we expect $\lambda_{ji} \approx x(E_j)^{MC}/x(C_i)^{MC}$, but we also know that this is just an estimate, with some uncertainty. Fortunately, in this case we can make direct use of the Bayes' formula applied to MC events, if we model the prior using a convenient pdf. In fact, if we indicate by $\boldsymbol{\lambda}_i$ the i-th column of Λ, i.e. $\boldsymbol{\lambda}_i = \{\lambda_{1,i},\, \lambda_{2,i}, \ldots,\, \lambda_{n_E+1,i}\}$, we have

$$f[\boldsymbol{\lambda}_i \,|\, \boldsymbol{x}_E^{MC},\, x(C_i)^{MC},\, I] \;\propto\; P[\boldsymbol{x}_E^{MC} \,|\, x(C_i)^{MC},\, \boldsymbol{\lambda}_i,\, I] \cdot f(\boldsymbol{\lambda}_i \,|\, I) \,. \qquad (9)$$

  Since $P[\boldsymbol{x}_E^{MC} \,|\, x(C_i)^{MC},\, \boldsymbol{\lambda}_i,\, I]$ is a multinomial, choosing a Dirichlet prior (see Appendix A), an extension of the Beta distribution to many dimensions, we get a Dirichlet posterior, i.e.

$$\boldsymbol{\lambda}_i \;\sim\; \mathrm{Dir}(\boldsymbol{\alpha}_{posterior_i}) \,, \qquad (10)$$

  with

$$\boldsymbol{\alpha}_{posterior_i} \;=\; \boldsymbol{\alpha}_{prior_i} + \boldsymbol{x}_E^{MC}\big|_{x(C_i)} \,. \qquad (11)$$

[6]It is well know that the sum of two Poisson variables is still a Poissonian. Equally, the sum of two binomial variables having *the same* parameter 'p' is still a binomial. (This 'reproductive property' applies to few other distributions). But the sum of two binomial variables with different p does not have a closed expression (this can be understood either intuitively, just thinking to the meaning of a binomial, or, more formally, analyzing the product of the characteristic functions of each binomial). The same happens with a multinomial, that is just an extension of the binomial to more than two possible outcomes.

- Finally there is the integral (4), which, as we said above, could be performed by MC methods. But, again, the real problem is $P(\boldsymbol{x}_C \,|\, \boldsymbol{x}_E, \Lambda,\, I)$ [instead, $f(\Lambda \,|\, \mathrm{MC}, I)$ is easy to sample, being made of several independent Dirichlet distributions].

In conclusion, being unable to tackle the problem from the main door, we need some 'tricks' to circumvent the obstacle, but still under the guidance of probability theory.

3 Practical algorithm to perform an independent-bin Bayesian unfolding

The basic trick to avoid the mentioned difficulty is to apply Bayes' theorem to causes and effects, instead than to true and observed spectrum. In practice, instead of using of Eq. (3) we start from

$$P(C_i \,|E_j,\, I) \;=\; \frac{P(E_j \,|\, C_i,\, I) \cdot P(C_i \,|\, I)}{\sum_i P(E_j \,|\, C_i,\, I) \cdot P(C_i \,|\, I)}\,, \tag{12}$$

i.e.

$$\theta_{ij} \;=\; \frac{\lambda_{ji} \cdot P(C_i \,|\, I)}{\sum_i \lambda_{ji} \cdot P(C_i \,|\, I)}\,, \tag{13}$$

having defined $\theta_{ij} \equiv P(C_i \,|E_j,\, I)$.

At this point a very important remark is in order. The prior in Eqs. (12)-(13) has a different meaning than that of Eq. (3). In Eq. (3) $P(\boldsymbol{x}_C \,|\, I)$ assigns different probabilities to all possible spectra. In Eqs. (12)-(13) $P(C_i \,|\, I)$ stands for *a single spectrum* (more precisely, all spectra that differ from each other just by normalization). This can be better understood analyzing the meaning of 'uniform' (or 'flat') referred to $P(\boldsymbol{x}_C \,|\, I)$ and to $P(C_i \,|\, I)$.

- $P(\boldsymbol{x}_C \,|\, I) = \textit{constant}$ means *all spectra* are equally likely.
- $P(C_i \,|\, I) = \textit{constant}$ means we consider the cause bins equally likely, i.e. the prior assess an initial belief in *flat spectra*.

In other words, while a flat $P(\boldsymbol{x}_C \,|\, I)$ means indifference about all possible spectra in order to 'let the data speak by themselves', a flat $P(C_i \,|\, I)$ is a strong assumption that usually does not correspond to our priors concerning the physics case. This implies that we have to tune somehow the algorithm in order to take into account of this gross approximation. We will come back to this issue in Sec. 4.

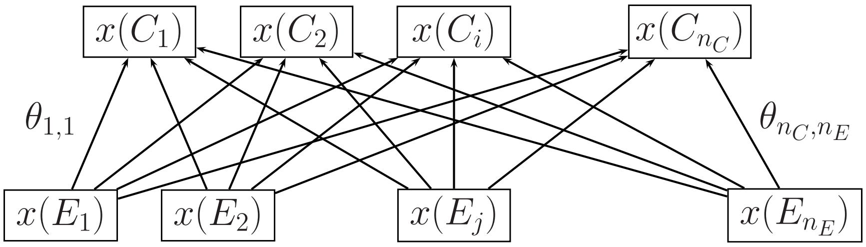

Figure 4: *Sharing counts observed in effect-cells among cause-cells according to* $\theta_{ij} = P(C_i \,|\, E_j,\, I)$.

At this point, having evaluated $P(C_i \,|\, E_j,\, I)$, we can use it to share the counts observed in each effect-bin among all cause-bins (see Fig. 4). The estimate of the true spectrum is obtained repeating this sharing for all observed bins and taking into account inefficiency. An uncertainty on the unfolded spectrum is also evaluated. Let us see how all this was done in the old algorithm and how it has been improved.

3.1 Old algorithm (Ref. [2])

In the old algorithm the share was done only considering expectations, and applying an inefficiency correction, going through the following steps:

- number of counts in C_i due to the observation in E_j:

$$x(C_i)|_{x(E_j)} \approx P(C_i \,|E_j,\, I) \cdot x(E_j) = \theta_{ij} \cdot x(E_j)\,; \tag{14}$$

- number of counts in C_i due to all observations:

$$x(C_i)|_{\boldsymbol{x}_E} \approx \sum_{j=1}^{n_E} P(C_i \,|E_j,\, I) \cdot x(E_j) = \sum_{j=1}^{n_E} \theta_{ij} \cdot x(E_j)\,; \tag{15}$$

- number of counts in C_i that also takes into account efficiency (ϵ_i):

$$x(C_i) \approx \frac{1}{\epsilon_i}\, x(C_i)|_{\boldsymbol{x}_E} = \frac{1}{\epsilon_i} \sum_{j=1}^{n_E} P(C_i \,|E_j,\, I) \cdot x(E_j) \tag{16}$$

$$= \frac{1}{\epsilon_i} \sum_{j=1}^{n_E} \theta_{ij} \cdot x(E_j) \tag{17}$$

where

$$[\,\epsilon_i \;=\; \sum_{j=1}^{n_E} P(E_j\,|\,C_i,\,I) = \sum_{j=1}^{n_E} \lambda_{ji} = 1 - \lambda_{n_E+1,\,i}\,]\;. \tag{18}$$

- Finally, we have to remember that λ_{ij} were estimated by MC simulation as

  $$\lambda_{ji} \approx x(E_j)^{MC}/x(C_i)^{MC}\,, \tag{19}$$

  from which ϵ_i and θ_{ij} are calculated according to Eqs. (18) and (13).

Essentially, $x(C_i)$ of Eq. (16) was considered an 'estimator', whose 'error' was calculated by standard 'error propagation' (this was a pragmatic compromise between 'standard methods' I was at that time accustomed and the Bayesian approach I had just learned — anyhow, not bad to begin, and not worst than other methods.)

At this point, there is still open the issue of the inappropriate priors, that we have encountered at the beginning of this section. Since this question remains in the improved algorithm, it will be discussed later in Sec. 4.

3.2 Improvements

As it is easy to understand from the previous description, weak points of the old algorithm are the treatment of small numbers and the calculation of uncertainty by 'error propagation formulas'. Let us see how we can improve them.

- Instead of just 'estimate' λ_{ij} as $x(E_j)^{MC}/x(C_i)^{MC}$, we can model their pdf by a Dirichlet (see Appendix A):

  $$\boldsymbol{\lambda}_i \;\sim\; \mathrm{Dir}[\boldsymbol{\alpha}_{prior} \,+\, \boldsymbol{x}_E^{MC}\big|_{x(C_i)^{MC}}]\,, \tag{20}$$

  where

  - $\boldsymbol{\lambda}_i = \{\lambda_{1,i}\,,\,\lambda_{2,i}\,,\,\ldots\,,\,\lambda_{n_E+1\,,i}\}$;
  - $\boldsymbol{\alpha}_{prior}$ is the initial set of Dirichlet parameters — typically $\alpha_{prior_i} \approx \mathcal{O}(1)$ for vague priors (see details in Appendix B);
  - $\boldsymbol{x}_E^{MC}\big|_{x(C_i)^{MC}}$ stands for the numbers of MC counts generated in C_i and which end in each of *all* possible n_E+1 'effects' (i.e. we also have to consider the inefficiency bin — see Fig. 2 — in order to satisfy the condition $\sum_i \lambda_i = 1$ of the Dirichlet variables).

- The uncertainty about Λ is taken into account by *sampling* its values. For each sampling:
  - extract smearing the matrix elements λ_{ji} according to a Dirichlet distribution;
  - calculate the efficiencies $\epsilon_i = \sum_{j=1}^{n_E} \lambda_{ji}$;
  - calculate the inverse probabilities θ_{ij} by Bayes' theorem.
- The share of the number of counts $x(E_j)$ observed in an effect-cell among all cause-bins is performed using a multinomial distribution (see Appendix A):

$$\boldsymbol{x}_C|_{x(E_j)} \sim \mathrm{Mult}[x(E_j),\, \boldsymbol{\theta}_j]\,, \tag{21}$$

  with $\boldsymbol{\theta}_j = \{\theta_{1\,j},\, \theta_{2\,j},\, \ldots,\, \theta_{n_C\,j},\, \}$.
- The contributions of all *observed* effect-bins are summed up, i.e.

$$\boldsymbol{x}_C|_{\boldsymbol{x}_E} = \sum_{j=1}^{n_E} \boldsymbol{x}_C|_{x(E_j)}\,. \tag{22}$$

  Inefficiency are taken into account as in the old algorithm:

$$\boldsymbol{x}_C = \frac{\boldsymbol{x}_C|_{\boldsymbol{x}_E}}{\boldsymbol{\epsilon}}\,, \tag{23}$$

  where the ratio in the r.h.s is done component by component.

However, we need to remember that the observed number of events in each effect bin, $x(E_j)$, comes from a Poisson distribution with unknown parameter μ_j, whose inference is usually modeled by a conjugate gamma distribution (see e.g. Ref. [6]),

$$\mu_j \sim \mathrm{Gamma}[c_j + x(E_j),\, r_j + 1]\,, \tag{24}$$

where c_j and r_j are the initial parameters of the gamma. [The case of a flat prior corresponds to $c_j = 1$ and $r_j \to 0$, i.e. an exponential with vanishing rate.] Some issues related to the choice of c_j and r_j will be discussed in Appendix B.

Therefore, what should be shared among the cause-bins is not $x(E_j)$ but μ_j, extracted in each sampling according to Eq. (24). However, μ_j is a continuous variable and we cannot use it as 'trial parameter' of the multinomial distribution to share the events. Nevertheless, fractional events are shared with a little trick.

- Extract μ_j according to Eq. (24) and round it to the closest *positive* integer, that we indicate here with m_j. This will be the number of trials used in the multinomial random generator.

- Extract an $\boldsymbol{x}_C|_{m_j}$ and rescale it according to μ_j/m_j, i.e.

$$\boldsymbol{x}_C|_{m_j} \sim \mathrm{Mult}(m_j,\, \boldsymbol{\theta}_j)\,, \tag{25}$$

$$\boldsymbol{x}_C|_{\mu_j} = \frac{\mu_j}{m_j}\, \boldsymbol{x}_C|_{m_j}\ . \tag{26}$$

- This means that Eq. (22) has to be replaced by

$$\boldsymbol{x}_C|_{\boldsymbol{x}_E} = \sum_{j=1}^{n_E} \boldsymbol{x}_C|_{\mu_j}\ ; \tag{27}$$

  then the inefficiency corrections are applied as usual.

Each sampling provides a spectrum $\boldsymbol{x}_C^{(t)}$, where t is the 'time' index of the sampling. After N samplings we can calculate an average spectrum, variance and covariance for each cause-bin, as well as covariances among bins, and any other statistical summary; or we can inspect graphically each $x(C_i)$

This algorithm has been implemented in R [4] and it is freely available on the web [5] under GPL license [10]. The R language has been chosen because it is a powerful scripting language, open source, multi-platform, oriented towards statistics applications and well maintained.[7] In order to test the algorithm, also a simple event generator has been included, that reproduces the toy models of Ref. [2]. An overview of the program is provided in Appendix C, while some results will be presented in Sec. 5

[7] I take the opportunity to make a point about the teaching of scripting languages, and in particular of *R*, in the physics courses. Surely physics students have to learn C, and perhaps C++, but it is a matter of fact that writing some C code to solve little/medium problems that occur everyday, and that might also need some graphics, is a pain, not only for students. The consequence is that students usually do not write little programs in C to solve the programs they meet in the general physics courses or in the laboratory courses, using instead, at most, spread-sheets(!). In my opinion, teaching a scripting language like R, possibly in parallel to C (but if I really had to chose, I would opt for R, as the language to begin), would be a good solution. The students have to thing anyhow in terms of *loops*, *logical conditions* and so on (and even of *objects*), but they could also use very-high level functions to perform complicated things with a couple of commands, using then a real programming language for everyday tasks. (Personally, since I have discovered scripting languages, I use C only for though professional programs. After I have discovered R, I have even almost abandoned *Mathematica*, unless I really need symbolic calculations.)

4 Iterations and smoothing

Let us now go back to the issue of the priors, that we have left open from the beginning of the previous section. The crucial thing to understand is that, as we said above, *instead of using a prior over the possible spectra, we are using a particular spectrum as prior*. Therefore, the posterior [i.e. the ensemble of $\boldsymbol{x}_C^{(t)}$ obtained by sampling] does not corresponds to our initial belief of indifference over all possible spectra. Obviously, the first idea a practical physicist has in order to fix the problem is to do some *fine tuning*. In fact, simulation on toy models shows that the unfolded distribution reproduces rather well the true one, even with a flat spectrum as prior. However, it still 'remembers' the 'flat spectrum prior'. This effect can be cured by iteration, using the posterior as prior in a subsequent unfolding. From simulations we learn that, in 'normal' cases, just two or three steps are sufficient to recover quite accurately the true spectrum.

As it has been discussed in length in Ref. [2], we cannot repeat the iterations for a very long time, otherwise there will be a kind of positive feedback, driven by fluctuations, that makes the asymptotically unfolded spectrum 'crazy' (it simply means that it is far away from all reasonable expectations). Again, as we understand it, it is a question of judging the outcome by our physics priors,[8] that, although rather vague, tell us that, given the kind of measurement, a spectrum with wild oscillations is far from our beliefs. This kind of problems happens also with other kind of unfolding algorithms and it is usually cured by *regularization*. Regularization is based on the subjective scientific prejudice that 'wild' distributions are not physical (and, as the reader knows or might imagine, this very reasonable subjective ingredient is unavoidably also embedded in all non-Bayesian methods). In practice, regularization can be implemented constraining adjacent bins not to be 'too discontinuous', for example limiting the absolute value of the derivatives calculated numerically on the unfolded spectrum. Or one might fit the spectrum with some orthonormal functions, but dumping 'high frequencies'. And so on.

I do not have a strong opinion on the matter. My recommendation is one should know what one is doing. My preferred regularization method for one-dimensional problems consists in smoothing the posterior before injecting it as prior of the next iteration (but not after the last step!). This smoothing can be performed by a low order polynomial fit, as it is implemented in the demo scripts accompanying the program. I find this technique quite

[8]Good physicists do have priors and always use them!

simple, well consistent with the spirit of this unfolding method and with the reasons to go through the iterations. Moreover, it is easy to understand that the procedure unfolding-smoothing-unfolding converges rapidly (unless one has some kinds of pathological situations, that might eventually spotted analyzing the convergence of the algorithm).

In conclusion, although the idea of iteration does not seem very Bayesian (we cannot squeeze the same data twice!), it is in fact just an adaptive way to recover what could have been obtained by a more reasonable uniform prior over the possible spectra (stated with other words, it would be non-Bayesian to stick to the first iteration, obtained from a prior that rarely corresponds to physics spectra!). For this reason the idea of iteration has been kept, although some effort has been done to get rid of it. In fact, in the new code the user can, instead of providing a (typically flat) spectrum as prior, set up a function [`priorf()`] that returns a normalized spectrum chosen at random according to a probability distribution describing prior knowledge. In this way, in each sampling also the prior is sampled. However, I anticipate that implementing the function `priorf()` might not be that trivial, and iterative procedure still remains a pragmatic compromise to slalom among the difficulties.

5 Results on toy models

The program has been checked with the same toy smearing matrices used in Ref. [2], reproduced here in Tab. 1.

Smearing 1

	C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8	C_9	C_{10}
E_1	0.010	0	0	0	0	0	0	0	0	0
E_2	0.025	0	0	0	0	0	0	0	0	0
E_3	0.040	0	0	0	0	0	0	0	0	0
E_4	0.025	0.050	0	0	0	0	0	0	0	0
E_5	0	0.100	0	0	0	0	0	0	0	0
E_6	0	0.050	0.120	0	0	0	0	0	0	0
E_7	0	0	0.120	0	0	0	0	0	0	0
E_8	0	0	0.060	0.160	0.025	0	0	0	0	0
E_9	0	0	0	0.160	0.100	0.120	0.140	0	0	0
E_{10}	0	0	0	0.080	0.250	0.360	0.280	0.200	0	0
E_{11}	0	0	0	0	0.125	0.120	0.280	0.400	0.225	0
E_{12}	0	0	0	0	0	0	0	0.200	0.450	0.250
E_{13}	0	0	0	0	0	0	0	0	0.225	0.500
E_{14}	0	0	0	0	0	0	0	0	0	0.250
ϵ	*0.1*	*0.2*	*0.3*	*0.4*	*0.5*	*0.6*	*0.7*	*0.8*	*0.9*	*1*

Smearing 2

	C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8	C_9	C_{10}
E_1	0.045	0	0	0	0	0	0	0	0	0.225
E_2	0.090	0.054	0	0	0	0	0	0	0.225	0.405
E_3	0.045	0.072	0.045	0	0	0	0	0.225	0.360	0.225
E_4	0	0.054	0.045	0.045	0.225	0.180	0.360	0.405	0.225	0
E_5	0	0	0	0.180	0.360	0.450	0.315	0.225	0	0
E_6	0	0	0	0.315	0.180	0.180	0.180	0	0.045	0
E_7	0	0	0.135	0.270	0.045	0	0	0.045	0	0
E_8	0	0.045	0.270	0.045	0	0	0.045	0	0.045	0
E_9	0	0.135	0.270	0	0.045	0.045	0	0	0	0.045
E_{10}	0	0.360	0.045	0	0.045	0.045	0	0	0	0
E_{11}	0.180	0.180	0	0	0	0	0	0	0	0
E_{12}	0.270	0	0	0.045	0	0	0	0	0	0
E_{13}	0.270	0	0.045	0	0	0	0	0	0	0
E_{14}	0.090	0	0.045	0	0	0	0	0	0	0
ϵ	*0.9*	*0.9*	*0.9*	*0.9*	*0.9*	*0.9*	*0.9*	*0.9*	*0.9*	*0.9*

Table 1: *Smearing matrices $P(E_j \,|\, C_i)$ of the toy models*

Appendix A – Multinomial and Dirichlet distributions

A1 Multinomial distribution

The Multinomial distribution is the extension of the Binomial from 2 to k outcomes. If we assign the probability p_i to the i-th outcome and think of n trials, then we are interested in the joint probability to observe x_1 times the outcome 1, x_2 times the outcome 2, and so on. From basic probability theory rules and some combinatorics we get

$$f[\,\boldsymbol{x}\,|\,\mathrm{Mult}(n,\,\boldsymbol{p})\,] \;=\; \frac{n!}{x_1!\,x_2!\cdots x_k!}\,p_1^{x_1}p_2^{x_2}\cdots p_k^{x_k}\,, \tag{28}$$

with

$$\begin{aligned}\boldsymbol{x} &= \{x_1, x_2,\,\ldots,\,x_k\} \qquad [\,x_i = 0, 1, \ldots, n;\;\; \sum_i x_i = n\,]\\ \boldsymbol{p} &= \{p_1, p_2,\,\ldots,\,p_k\} \qquad [\,0 < p < 1;\;\; \sum_i p_i = 1\,]\,.\end{aligned}$$

It is easy to see that the binomial distribution is recovered for $k = 2$. Note that, given the constraint $\sum_i x_i = n$, the multinomial has to be seen as a distribution for $k-1$ variables $x_1, x_2, \ldots, x_{k-1}$, as it is clear from the binomial case. Expected values, variances, covariances and correlation coefficients are given by

$$\mathrm{E}(X_i) \;=\; n\,p_i \tag{29}$$

$$\mathrm{Var}(X_i) \;=\; n\,p_i\,(1-p_i) \tag{30}$$

$$\mathrm{Cov}(X_i, X_j) \;=\; -n\,p_i\,p_j \qquad [i \neq j] \tag{31}$$

$$\rho(X_i, X_j) \;=\; \frac{-p_i\,p_j}{\sqrt{p_i\,(1-p_i)\,p_j\,(1-p_j)}} \qquad [i \neq j]\,. \tag{32}$$

Each marginal, i.e. $f[x_i\,|\,\mathrm{Mult}(n,\,\boldsymbol{p})]$, is a binomial with parameters n and p_i:

$$f[x_i\,|\,\mathrm{Mult}(n,\,\boldsymbol{p})] \;=\; \frac{n!}{x_i!\,(n-x_i)}\,p_i^{x_i}\,(1-p_i)^{n-x_i}\,. \tag{33}$$

For $k = 2$ the two variables X_1 and X_2 are 100% anticorrelated, as it is quit obvious. In fact, since $p_2 = 1 - p_1$, we get from Eq. (32)

$$\rho(X_1, X_2\,|\,k=2) \;=\; \frac{-p_1\,(1-p_1)}{\sqrt{p_1\,(1-p_1)\,(1-p_1)\,p_1}} = -1 \tag{34}$$

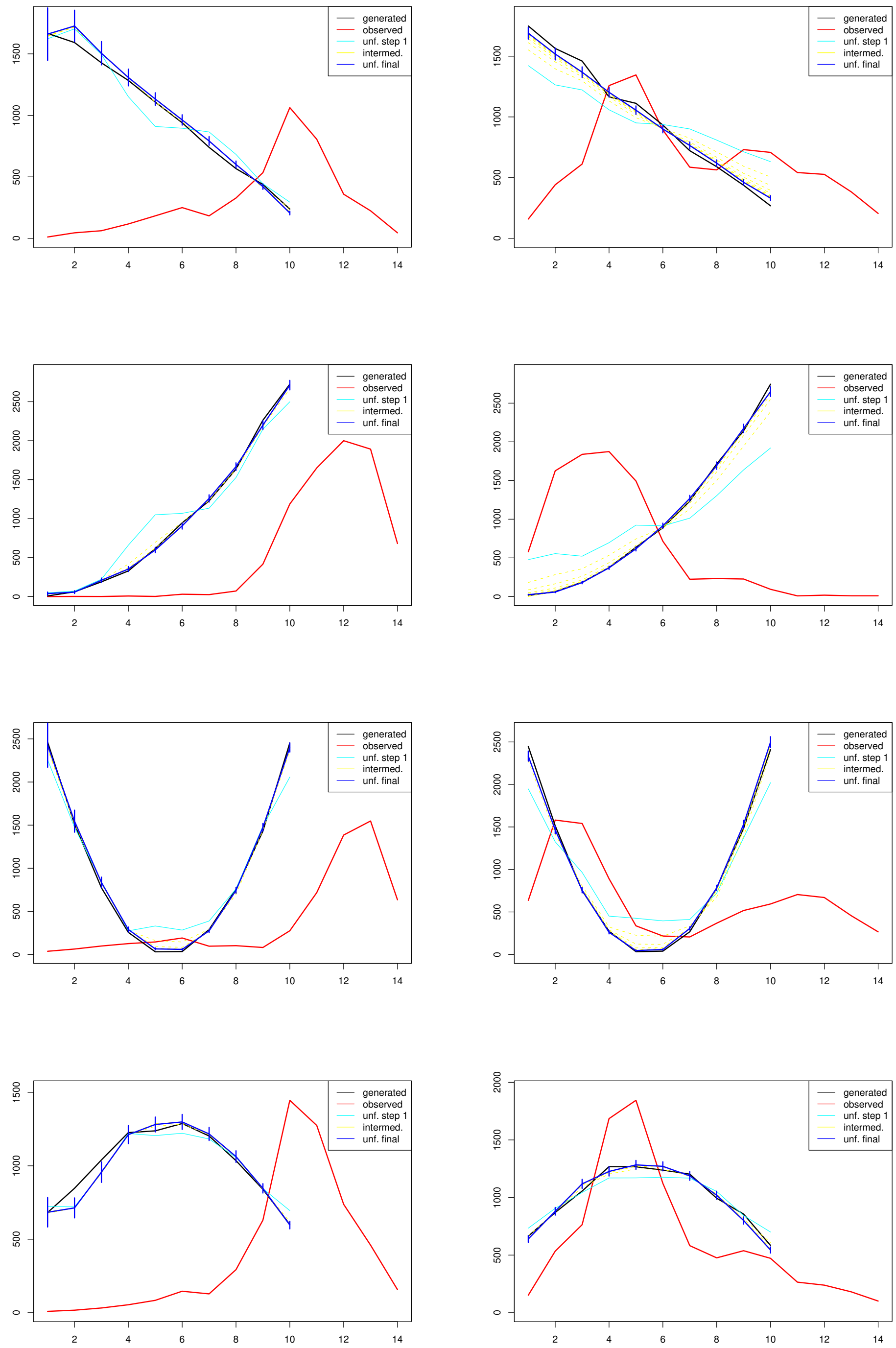

Figure 5: *Result of a toy model*

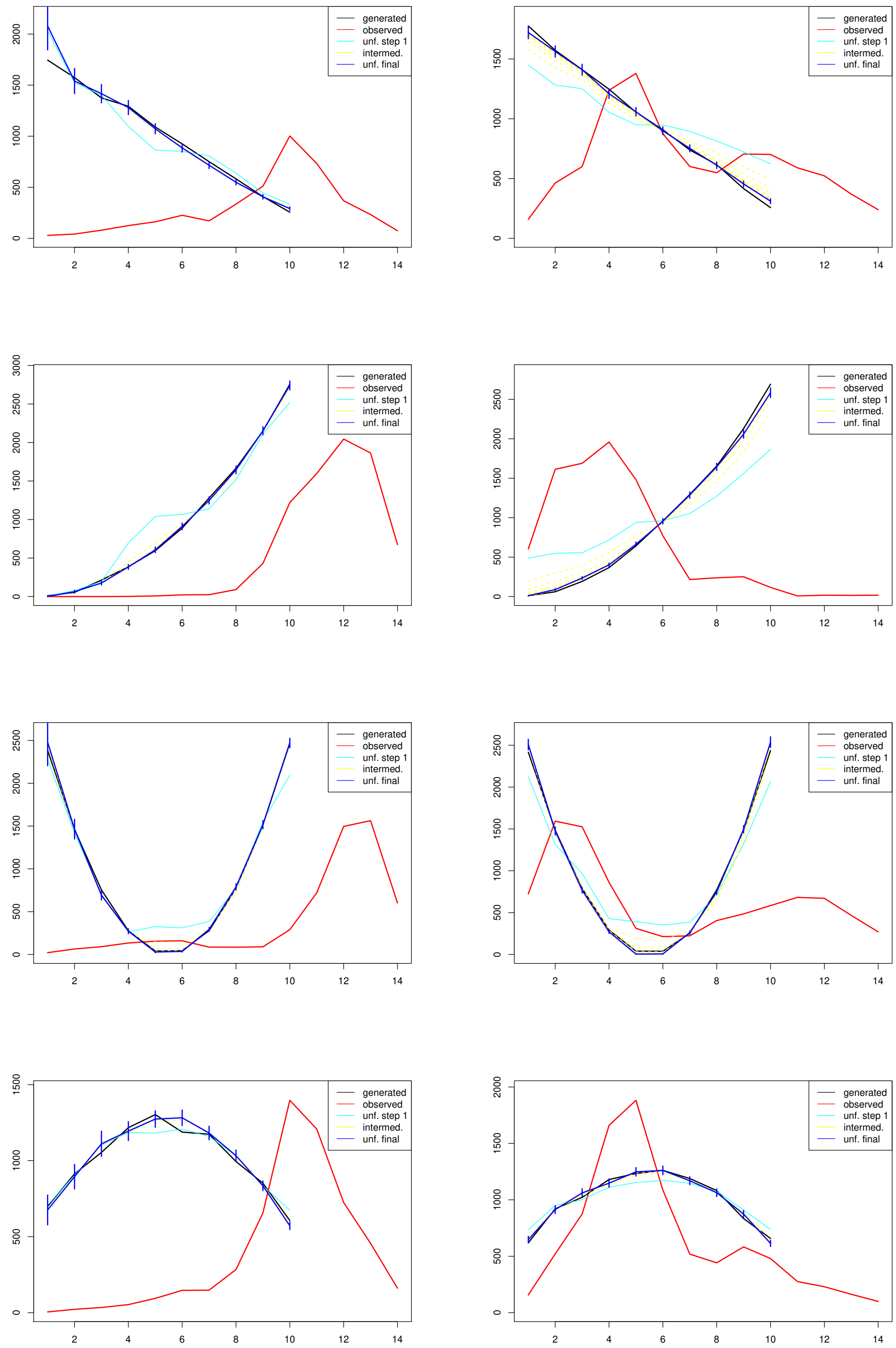

Figure 6: *Same as Fig. 5. Independent complete simulation.*

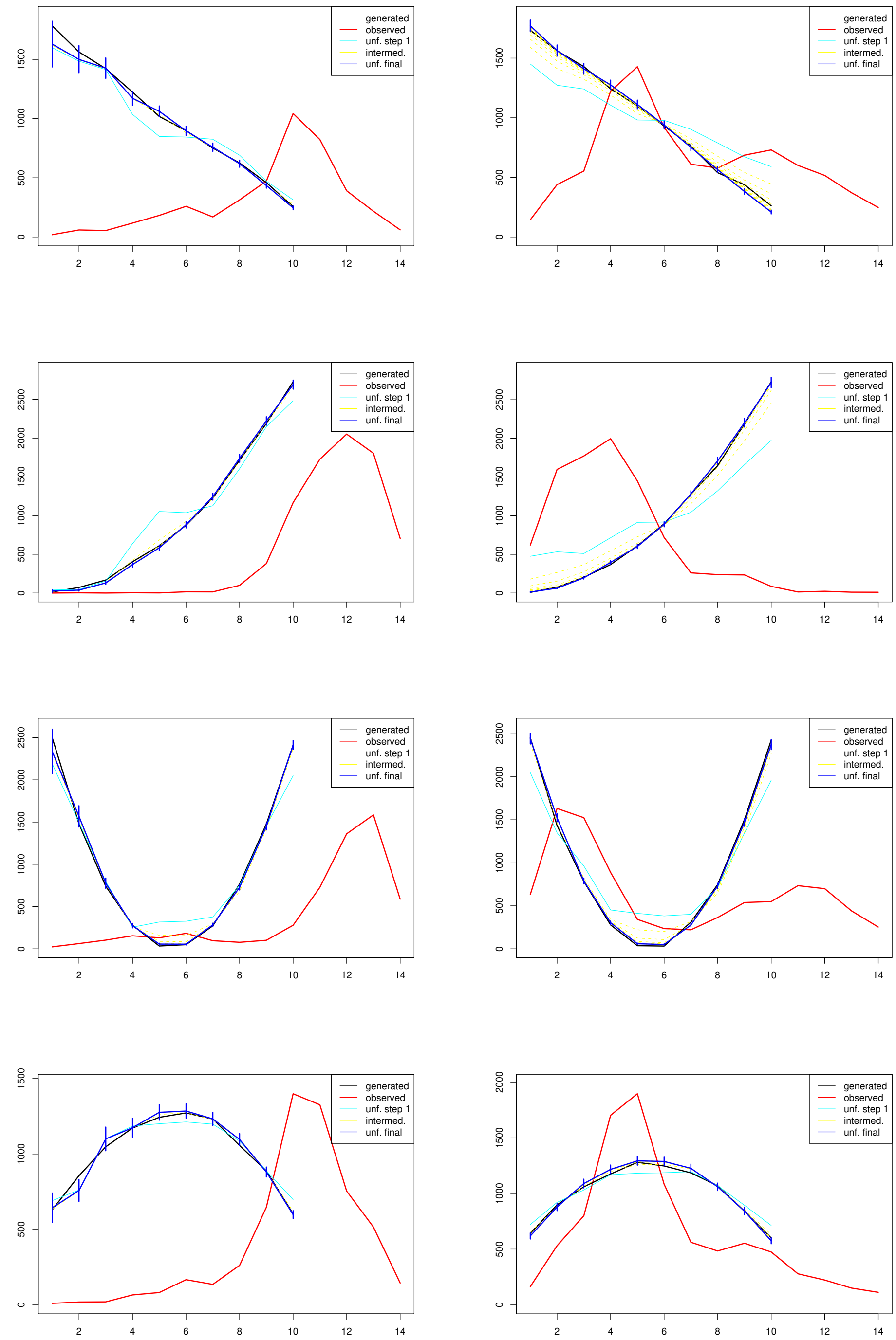

Figure 7: *Same as Fig. 5. Independent complete simulation.*

A.2 Dirichlet distribution

In this case we start from the formal definition. Given k continuous variables whose value can range between zero and one, the Dirichlet distribution is defined by the following joint pdf:

$$f[\,\boldsymbol{x}\,|\,\mathrm{Dir}(\boldsymbol{\alpha})\,] \;=\; \frac{\Gamma(\alpha_1+\alpha_2+\cdots+\alpha_k)}{\Gamma(\alpha_1)\,\Gamma(\alpha_2)\,\cdots\,\Gamma(\alpha_k)}\,x_1^{\alpha_1-1}\,x_2^{\alpha_2-1}\,\ldots\,x_k^{\alpha_k-1} \tag{35}$$

with

$$\begin{aligned} \boldsymbol{x} &= \{x_1, x_2, \ldots, x_k\} \qquad [\,0 < x_i < 1;\ \sum_i x_i = 1\,] \\ \boldsymbol{\alpha} &= \{\alpha_1, \alpha_2, \ldots, \alpha_k\} \qquad [\,\alpha_i > 0\,]\,, \end{aligned}$$

where $\Gamma(\,)$ is the usual gamma function. If $k = 2$, then the Beta is recovered. Also in this case there is a constraint on the variables, $\sum_{i=1}^{k} x_i = 1$, that makes the distribution in fact $(k-1)$-dimensional. Expected values, variances and covariances are

$$\mathrm{E}(X_i) \;=\; \frac{\alpha_i}{\alpha} \tag{36}$$

$$\mathrm{Var}(X_i) \;=\; \frac{\alpha_i\,(\alpha-\alpha_i)}{\alpha^2\,(\alpha+1)} \tag{37}$$

$$\mathrm{Cov}(X_i, X_j) \;=\; \frac{\alpha_i\,\alpha_j}{\alpha^2\,(\alpha+1)} \qquad [i \neq j]\,, \tag{38}$$

where $\alpha = \sum_i \alpha_i$. Each marginal is a Beta with parameters $r = \alpha_i$ and $s = \alpha - \alpha_i$:

$$f[x_i\,|\,\mathrm{Beta}(r=\alpha_i, s=\alpha-\alpha_i)] \;=\; \frac{x_i^{\alpha_i-1}(1-x_i)^{\alpha-\alpha_i-1}}{\beta(\alpha_i,\,\alpha-\alpha_i)}\,, \tag{39}$$

where $\beta()$ is the beta function (hence the name to the distribution), defined as

$$\beta(r,s) = \int_0^1 z^{r-1}(1-z)^{s-1}\,\mathrm{d}z\ \left[= \frac{\Gamma(r)\,\Gamma(s)}{\Gamma(r+s)}\right]. \tag{40}$$

Note that the Beta pdf vanishes for $x_i = 0$ if $r > 1$, and for $x_i = 1$ if $s > 1$, respectively. This observation will be important for some considerations concerning the treatment of bins with zero counts (see Appendix B.1).

A.3 Dirichlet as prior conjugate of the multinomial

If the data $\boldsymbol{x}$ have been observed in n trials and we are interested in inferring the multinomial parameters $\boldsymbol{p}$, applying Bayes' theorem we get

$$\begin{aligned} f[\,\boldsymbol{p}\,|\,\mathrm{Mult}(n),\,\boldsymbol{x}\,] &\propto f[\,\boldsymbol{x}\,|\,\mathrm{Mult}(n,\,\boldsymbol{p})\,]\,\cdot\,f(\boldsymbol{p}) && (41)\\ &\propto p_1^{x_1}p_2^{x_2}\cdots p_k^{x_k}\,\cdot\,f(\boldsymbol{p})\,, && (42)\end{aligned}$$

having absorbed in the normalization all factors that do not depend on $\boldsymbol{p}$. If we model the prior $f(\boldsymbol{p})$ with a Dirichlet, i.e. $f(\boldsymbol{p}) \propto p_1^{\alpha_1-1}\,p_2^{\alpha_2-1}\,\ldots\,p_k^{\alpha_k-1}$, we get

$$\begin{aligned} f[\,\boldsymbol{p}\,|\,\mathrm{Mult}(n,\,\boldsymbol{x})\,] &\propto (\,p_1^{x_1}p_2^{x_2}\cdots p_k^{x_k}\,)\,\cdot\,(\,p_1^{\alpha_1-1}\,p_2^{\alpha_2-1}\,\ldots\,p_k^{\alpha_k-1}\,) && (43)\\ &\propto p_1^{\alpha_1+x_1-1}\,p_2^{\alpha_2+x_2-1}\,\ldots\,p_k^{\alpha_k+x_k-1}\,, && (44)\end{aligned}$$

that is still a Dirichlet. In practice, the observation $\boldsymbol{x}$ has update the Dirichlet parameters according to the rule

$$\boldsymbol{\alpha}_{posterior} = \boldsymbol{\alpha}_{prior} + \boldsymbol{x}\,. \tag{45}$$

For this reason the Dirichlet is known as the *prior conjugate* of the multinomial. In the simplest case of $k = 2$ we have the Beta prior conjugate of the Binomial (see e.g. Ref. [6]).

Appendix B – Handling the zeros

Bins with null counts might occur either in the Monte Carlo results used to infer the smearing matrix, or in the observed spectrum. As it is true that when we deal with large numbers we can neglect fluctuations and uncertainties (this is well understood in statistical mechanics), while small numbers are somewhat problematic, it is also true that handling zeros is particularly challenging and you should mistrust abstract mathematical arguments of any kind. In fact, when dealing with zeros, subjective prior knowledge becomes crucial, because we need to have an idea of what a zero 'might mean': for some bins we could reasonably think that if we would repeat the same experiment or the same simulation the zero might turn into one count, or even in two o three: for other bins we are quite confident that we would need to repeat the experiment or the simulation many many times before a zero turns into one count, or perhaps we have good reasons to believe this will never happen. In other words, experienced physicists think that 'not all zeros are the same'. Think for examples of an histogram in which, after some bins with positive (and perhaps quite large) counts, suddenly there is a row of empty bins (like the second row of Tab. 2). Intuitively we do not think all the empty bins are equivalent, although for a mathematician their are. But we are physicists, and we have formed some opinion about how nature behaves. As a consequence, we do not only tend to smooth the positive numbers of the histogram, but we also tend to make some extrapolations to the region of zero counts, and judge the far zeros to be 'more zero' that the near ones. We need then to model somehow our opinion based on past experience. Let us see how we can treat the zeros in the evaluation of the smearing matrix and in the observed spectrum. (But don't worry: the program gives you the option of treat the zeros as ... zeros, and you might want to skip the following two subsections. But, perhaps, considering, democratically, all zeros on equal foot might be against your beliefs, while certainly you do not want to be incoherent...)

B.1 – Zero counts in effect-bins of MC events used to infer smearing matrix

As explained in Sec. 2, the columns of the smearing matrix are jointly described by vectors of Dirichlet random variables (one vector per cause-bin). The parameters of a Dirichlet pdf are updated according to Eq. (45), that we rewrite here:

$$\boldsymbol{\alpha}_{posterior} \;=\; \boldsymbol{\alpha}_{prior} + \boldsymbol{x}\,,$$

where $\boldsymbol{x}$ stands here for $\boldsymbol{x}_E^{MC}\big|_{x(C_i)}$. [see Eq. (11)]. A usual choice of the Dirichlet prior to model indifference is to set all α's to 1. The *prior* is then characterized by the the following marginals, expectations and variances (we use the symbol p for the variables, for their physical meaning in our contest, and the index j for numbering them, as we do in the text for the effect-bins):

$$f[p_j\,|\,\mathrm{Beta}(1,\,\nu-1)] \;=\; \frac{(1-p_j)^{\nu-2}}{\beta(1,\,\nu-1)} \tag{46}$$

$$\mathrm{E}(p_j) \;=\; \frac{1}{\nu} \tag{47}$$

$$\mathrm{Var}(p_j) \;=\; \frac{\nu-1}{\nu^2\,(\nu+1)} \xrightarrow[\nu\gg 1]{} \frac{1}{\nu^2}\,, \tag{48}$$

where, hereafter, $\nu = n_E + 1$ to improve the readability of the formulas. In the case of $\nu = 2$ we recover a flat prior and hence an expectation of $1/2$.

If we trow n MC events in order to estimate the several p_j, the updated α parameters become $\alpha_j = x_j + 1$. For the marginals we have now:

$$f[p_j \,|\, \mathrm{Beta}(x_j+1,\, n+\nu-x_j-1)] \;=\; \frac{p^{x_j}\,(1-p_j)^{n+\nu-x_j-2}}{\beta(x_j+1,\, n+\nu-x_j-1)}\,, \quad (49)$$

with expected value and variance

$$\mathrm{E}(p_j) \;=\; \frac{x_j+1}{n+\nu} \quad (50)$$

$$\mathrm{Var}(p_j) \;=\; \frac{(x_j+1)\,(n+\nu-x_j-1)}{(n+\nu)^2\,(n+\nu+1)}\,. \quad (51)$$

In the limit of large numbers ($n \gg \nu$ and $x_j \gg 1$) we get an expected value equal to x_j/n, the fraction of events in the cell j, with a variance of $(x_j/n)\,(1-x_j/n)\,/\,n$.

In the case *zero counts and large* n the pdf is given by

$$f(p_j \,|\, x_j = 0) = (n+1)\,(1-p_j)^n\,, \quad (52)$$

having a sharp peak at $p_j = 0$, with sharpness growing with n. Expected number and standard deviation are $\mathrm{E}[p_j \,|\, x_j] = \sigma[p_j \,|\, x_j] = 1/n$ [moreover, in this case the cumulative has the easy expression $F(p_j \,|\, x_j = 0) = 1-(1-p_j)^{n+1}$, from which we can calculate a 95% probability upper limit for p_j to be $1 - 0.05^{1/(n+1)} \approx 3/n$].

5.0.1 Solving a formal paradox with some good sense

The above conclusions seem reasonable when applied to an individual empty bin. But let us see, with the help of Tab. 2, what happens where several elements of $\boldsymbol{x}$ are null: one hundred events have been generated in a cause-cell and the second row of Tab. 2, whose elements are indicated by $x(E_j)$, shows the counts in each effect-bin. The third row shows the priors of $\boldsymbol{\alpha}$ and the forth one the updated $\boldsymbol{\alpha}$. We also calculate in the following two rows expected values and standard deviation of the p_j values.

All elements of the smearing matrix that connect that cause-cell with the effect-cells with zero counts have $\mathrm{E}[p_j] = \sigma[p_j] \approx 0.9\%$. This looks reasonable if we consider each cell individually. But, observing the values of $\mathrm{E}[p_j]$ and $\sigma[p_j]$ in the fifth and sixth row of Tab. 2, any physicist would be uneasy, for a couple of reasons. First, we are have strong prejudices towards regularity of physical laws, including whatever causes the smearing. If the smearing is peaked around bin E_3 and drops in either sides, we do

E_j	1	2	3	4	5	6	7	8	9	10	11	12
$x(E_j)$	10	20	40	15	10	5	0	0	0	0	0	0
	Standard uniform priors											
α_{pr_j}	1	1	1	1	1	1	1	1	1	1	1	1
α_{post_j}	11	21	41	16	11	6	1	1	1	1	1	1
$\mathrm{E}[p_j]$	9.8	18.8	36.6	14.3	9.8	5.4	0.9	0.9	0.9	0.9	0.9	0.9
$\sigma[p_j]$	2.8	3.7	4.5	3.3	2.8	2.1	0.9	0.9	0.9	0.9	0.9	0.9
	$\mathrm{E}[p_{1-6}] = 94.6\%$ $\sigma[p_{1-6}] = 2.1\%$						$\mathrm{E}[p_{7-12}] = 5.4\%$ $\sigma[p_{7-12}] = 2.1\%$					
	$\alpha_j = 2/\nu$											
α_{pr_j}	1/6	1/6	1/6	1/6	1/6	1/6	1/6	1/6	1/6	1/6	1/6	1/6
α_{post_j}	10.2	20.2	40.2	15.2	10.2	5.2	0.2	0.2	0.2	0.2	0.2	0.2
$\mathrm{E}[p_j]$	10.0	19.8	39.4	14.9	10.0	5.1	0.02	0.02	0.02	0.02	0.02	0.02
$\sigma[p_j]$	3.0	3.9	4.8	3.5	3.0	2.2	0.4	0.4	0.4	0.4	0.4	0.4
	$\mathrm{E}[p_{1-6}] = 99.0\%$ $\sigma[p_{1-6}] = 1.0\%$						$\mathrm{E}[p_{7-12}] = 1.0\%$ $\sigma[p_{7-12}] = 1.0\%$					
	Regularized α's in bins with zero counts											
α_{pr_j}	1/6	1/6	1/6	1/6	1/6	1/6	1/6	1/6	1/6	1/6	1/6	1/6
α_{post_j}	10.2	20.2	40.2	15.2	10.2	5.2	0.2	0.2	0.2	0.2	0.2	0.2
f	1	1	1	1	1	1	1	1/2	1/4	1/8	1/16	1/32
$\alpha^{(r)}_{post_j}$	10.2	20.2	40.2	15.2	10.2	5.2	0.2	0.1	0.04	0.02	0.01	0.005
$\mathrm{E}[p_j]$	10.0	19.9	39.6	15.0	10.0	5.1	0.2	0.1	0.04	0.02	0.01	0.005
$\sigma[p_j]$	3.0	3.9	4.8	3.5	3.0	2.2	0.4	0.3	0.2	0.1	0.1	0.1
	$\mathrm{E}[p_{1-6}] = 99.7\%$ $\sigma[p_{1-6}] = 0.6\%$						$\mathrm{E}[p_{7-12}] = 0.3\%$ $\sigma[p_{7-12}] = 0.6\%$					
	Regularized and rescaledα's $[\,*:\ \sum_{k\subset\mathcal{M}}\alpha_k = m/\nu\,]$											
α_{pr_j}	1/6	1/6	1/6	1/6	1/6	1/6	1/6	1/6	1/6	1/6	1/6	1/6
α_{post_j}	10.2	20.2	40.2	15.2	10.2	5.2	0.2	0.2	0.2	0.2	0.2	0.2
f	1	1	1	1	1	1	1	1/2	1/4	1/8	1/16	1/32
$\alpha^{(r\&r)}_{post_j}\ *$	10.2	20.2	40.2	15.2	10.2	5.2	0.5	0.3	0.13	0.06	0.03	0.015
$\mathrm{E}[p_j]$	10.0	19.8	39.4	14.9	10.0	5.1	0.5	0.2	0.1	0.06	0.03	0.02
$\sigma[p_j]$	3.0	3.9	4.8	3.5	3.0	2.2	0.7	0.5	0.3	0.2	0.2	0.1
	$\mathrm{E}[p_{1-6}] = 99.0\%$ $\sigma[p_{1-6}] = 1.0\%$						$\mathrm{E}[p_{7-12}] = 1.0\%$ $\sigma[p_{7-12}] = 1.0\%$					

Table 2: *MC events, Dirichlet parameters* $\boldsymbol{\alpha}$ *and expectations of Dirichlet variables* p_j *(value in %) with several treatments of the zeros (see text).*

not expect the smearing probability $p_{12} = P(E_{12}\,|\,C_i\,,I)$ to be equal to $p_7 = P(E_7\,|\,C_i\,,I)$. Second, if we sum up the probabilities of all bins with no counts, i.e. $\sum_{j=7}^{12}\mathrm{E}[p_j]$, the total is not really negligible. The situation becomes worse if we add more bins in the right side.

Another way to understand the problem is starting from the probability density function of $p_{\mathcal{M}}$, the parameter of a binomial distribution that describes the occurrence of an event in the set $\mathcal{M}$ of m selected bins (not necessarily adjacent, although in Tab. 2 they are). Just extending a well known property of the Dirichlet distribution [see Appendix A.2, and in particular Eq. (39)], we get that $p_{\mathcal{M}}$ is described by a Beta pdf, with parameter $\alpha_{\mathcal{M}} = \sum_{k\subset\mathcal{M}}\alpha_k$ and $\alpha - \alpha_{\mathcal{M}}$, where $\alpha = \sum_j \alpha_j$:

$$f[p_{\mathcal{M}}\,|\,\mathrm{Beta}(\alpha_{\mathcal{M}},\,\alpha-\alpha_{\mathcal{M}})] \;=\; \frac{p_{\mathcal{M}}^{\alpha_{\mathcal{M}}-1}(1-p_{\mathcal{M}})^{\alpha-\alpha_{\mathcal{M}}-1}}{\beta(\alpha_{\mathcal{M}},\,\alpha-\alpha_{\mathcal{M}})}\,. \qquad (53)$$

If we take the set $\mathcal{M}$ to be that of the m bins with zero counts and for each of them we choose the initial value of α to be equal 1, we get:

$$f[p_{\mathcal{M}}\,|\,\mathrm{Beta}(m,\,n+k-m)] \;=\; \frac{p_{\mathcal{M}}^{m-1}(1-p_{\mathcal{M}})^{n+k-m-1}}{\beta(m,\,n+k-m)}\,. \qquad (54)$$

Contrary to (52), this pdf vanishes for $p_{\mathcal{M}} \to 0$, i.e. it states *we are a priori sure that $p_{\mathcal{M}}$ cannot be zero.* But this is a bit too much. We understand then that the required condition in order to have $f(p_{\mathcal{M}} = 0) = 0$ is to apply the following constraint to the α's: $\sum_{k\subset\mathcal{M}}\alpha_k \le 1$, that becomes $\alpha_{k\subset\mathcal{M}} \le 1/m$ in the case we consider all zero count bins on the same foot. Since the final α of the bins with many counts is influenced very little by the initial α, if this does not exceed $\mathcal{O}(1)$, we could simply require $\alpha_j \approx 1/\nu$. But, just to recover the uniform prior when $\nu = 2$, which can be written as a Dirichlet with $\alpha_1 = 1$ and $\alpha_2 = 1$ (that is a Beta with $r = 1$ and $s = 1$), our starting point will be to set $\alpha = 2/\nu$ for all bins. Then this value will modified due to the constraint $\sum_{k\subset\mathcal{M}}\alpha_k = m/\nu$ for the bins with zero counts.

5.0.2 Regularization of the smearing matrix prior

There is still the problem that we do not believe all zeros are the same. We can solve it shaping in some way the α's in a suitable way. Obviously, in order to do this we need to know the bin ordering in the parameter space, that was not taken into account till now. We show here a simple way of how this can be done in 1-dimensional spectrum unfolding. We set all initial α's to $2/\nu$ in all bins that have non-zero counts and in the adjacent ones. Then, as we move far away from the bins with non-zero counts we decrease, the value of alpha, for example with some geometric low (i.e. exponential decrease). In Tab. 2 we have done this exercise, although choosing, as academic exercise, the factor 1/2 from one zero count bin to the next one. That is perhaps a too conservative choice that still takes in too much considerations the tails (indeed, in the program the geometric factor is given as an option in the program, whose default value is equal to $1/e$, i.e. a 'natural' exponential decrease that gives roughly an order of magnitude suppression every two bins). The α's shaped in this way are finally rescaled with the condition $\sum_{k\subset\mathcal{M}}\alpha_k = m/\nu$. To conclude these are the steps made in the default practical method to handle zeros in 1-dimensional unfolding:

- All bins:
  - set $\alpha_j = 2/\nu + x_j$ in all bins.
- Only bins with zero counts:
  - reshape the α's with an exponentially low;[9]
  - rescale the α's by the condition $\sum_{k\subset\mathcal{M}}\alpha_k = m/\nu$.

The last rows of Tab. 2 show the results obtained applying this rule, which seems a reasonable compromise between the several pieces of informations and beliefs as they have been stated here. Needless to say, you are free to adjust the treatments of the zero according to the best knowledge of your physical case. The program offers you the possibility to return your preferred α's using the function `my.alphas()`, or just to decide that zeros are simply zeros, and *basta*.

B.2 – Zero counts in real data effect-bins

Having explained in length the strategy to treat the zero count bins in the inference of the smearing matrix, let us sketch a similar strategy that can be used to treat zero bins in the observed spectrum. Contrary to the case of the smearing matrix, where many zero can really occur, because long range smearing is usually rare, in the case of observed data think to treat the zeros might sound purely academic, because one can re-bin the spectrum to avoid zeros. Nevertheless, there might be cases of interest and it is then

[9] In the case of a sequence of zero counts bins surrounded on both sides by non-zero bins, the two exponential starting at each side are summed up, to mimic the fact that there might be an incoherent sum of the two tails;

important at least to predispose the program to deal with this situation, although requiring intelligent inputs by the user.

Appendix C – Program components

The R code [5] is divided in three small files ($\approx$ 20kB in total): `bayes_unf.R` contains the core functions; `bayes_extra.R` contains extra 'technical' functions, as well as functions used in iteration, smoothing and handling of zeros; `simula.R` contains the toy model event generator. The code has not been optimized in terms of performances, but, I hope, in terms of readability. Several demo scripts are also available to run, with different options, the unfolding on simulated events, as well as the script used to make figures [...]. We just give here a short overview on the main component, referring to the code itself for details.

`bayes_unf.R`

The core code is made of four functions.

`infer.pec(MC.gen, MC.obs)`
: Given the vector of MC event generated in each cause-cell and the matrix of MC event 'observed' in each effect-bin, it setup the Dirichlet parameters, calculate average smearing probabilities and efficiencies.

`Bayes(prior, pec)`
: given the `prior` and the smearing matrix `pec` $= P(E_j \,|\, C_i\, I) = \lambda_{ij}$ it returns a matrix of posteriors $P(C_i \,|\, E_j, I) = \theta_{ij}$. The argument `prior` can either be either a vector of probabilities (numbers that sum up to one, or the name of a function that, at each call returns such a vector.

`MC.unfold(prior, mc.smear, obs, N, HISTORY, ...)`
: This is what does the main work. The argument `prior` has the same meaning as above, while `obs` contains the vector of observations and `N` the number of samplings. Finally, `mc.smear` is an *object* contains several pieces of information regarding the smearing matrix. It might contains directly the smearing probabilities (λ_{ij}), and in this case the program takes these numbers without uncertainty, or it might contain the Dirichlet parameters calculated from the Monte Carlo events (see `infer.pec`).
The function returns an object that summarizes the unfolded distribution (averages, standard deviations, etc.). If the argument `HISTORY` is set, then all the history of the samplings is reported, for a more accurate study of the result.

`guess.distr(pce, eff, obs)`
: Assign events to causes, only using 'expectations'. Essentially what was done in the old algorithm. It might be used during the iterations, but the last one, to speed up the program.

`bayes_extra.R`

Here there are some extra functions, that might be useful for iterative use of the program in one-dimensional problems.

`smooth(true.evts, ORDER)`
: This does the polynomial fit smoothing. If the argument ORDER is negative, the polynomial fit is done use the log of the spectrum.

`myfit(x,y, ORDER)`
: Just a simple least square polynomial fit function in a few lines of code.

`rprior()`
: This is the function, to be provided by the user, that at each call returns a spectrum at random, according to some priors.

`simula.R`

This file contains a few functions to generate and smear the events according to several 'true spectra' and several unfolding matrices.

References

[1] G. D'Agostini, *Fits, and especially linear fits, with errors on both axes, extra variance of the data points and other complications*, physics/0511182.

[2] G. D'Agostini, *A multidimensional unfolding method based on Bayes' theorem*, Meth. in Phys. Res. **A362** (1995) 487

[3] G. D'Agostini, *Asymmetric uncertainties: sources, treatment and possible dangers*, physics/0403086.

[4] R Development Core Team (2005) *R: A language and environment for statistical computing. R Foundation for Statistical Computing*, Vienna, Austria. ISBN 3-900051-07-0, `http://www.R-project.org`.

[5] `http://www.roma1.infn.it/~dagos/prob+stat.html`.

[6] G. D'Agostini, *Bayesian reasoning in data analysis: A critical introduction*, World Scientific Publishing, 2003.

[7] G. D'Agostini, *Bayesian inference in processing experimental data: principles and basic applications*, Rep. Prog. Phys. 66 (2003)1383, physics/0304102.
V. Dose, *Bayesian inference in physics: case studies*, Rep. Prog.. Phys. 66 (2003)1421.

[8] R.E. Neapolitan, *Learning Bayesian Networks*, Prentice Hall, 2003.

[9] M. Raso, *A measurement of the $\sigma_{e\,p}$ for $Q^2 < 0.3\,GeV^2$, as a by product of the F_2^{ep} measurement*, ZEUS-98-075, October 1998 (available at `http://www.roma1.infn.it/~dagos/prob+stat.html#Raso`)

[10] GPL