Nested sampling for Fully Bayesian Unfolding (PENDING TITLE)

by

Jens Bratten Due

THESIS

for the degree of

MASTER OF SCIENCE

Faculty of Mathematics and Natural Sciences
University of Oslo

August 2021

Abstract

In this thesis we explore a Bayesian approach to the problem of unfolding, namely Fully Bayesian Unfolding (FBU) [1]. With the overarching goal of providing an alternative method to the default unfolding method in use by the nuclear physics group at UiO (the folding iteration method [2]), FBU is explained and used for unfolding real experimental γ-ray spectra. An explanation of the inner process in the PyFBU-package [3] is provided in order to demystify a relatively abstract implementation of underlying libraries and logic. This is done to yield a better understanding and confidence of the final results from FBU. This explanation is accompanied by a few tests of assumptions, specifically finding that the likelihood function indeed takes the form we assume, and is used in the way we expect. Furthermore, a modification to the package is formulated and implemented, with the purpose of facilitating an essential part of Bayesian thinking, the freedom of choice (of prior knowledge).

The experimental spectra in question is those of the ^{28}Si(p,p'γ) and ^{146}Nd(p,p'γ) reactions, the second of which has not before been unfolded using FBU. For ^{28}Si, our results have been compared with earlier results produced by Valsdóttir [4]. A newer response matrix more closely representing the experimental conditions has been used. By cutting out a low γ-energy area of the raw spectra corresponding to errors in the response matrices, more accurate results than before are produced. The results are evaluated using error metrics (R^2-score and Mean Absolute Error (MAE)) and comparisons between refolded spectra and observed data (raw spectra). For ^{146}Nd, we have unfolded both the first excited state, and a high excitation energy area, i.e. a spectrum with a higher degree of complexity. Both results have been compared with the folding iteration method in the OMpy library [5]. It is found that FBU is consistently more accurate, especially with the mentioned cutting of low-energy bins associated with response matrix errors. Both refolded vs. raw comparisons and error metrics are better for FBU for both investigated spectra. Along with the uncertainty estimates from calculated credibility intervals and built into the posterior distributions, this makes a good case for FBU as a powerful and general unfolding method.

Acknowledgements

Contents

CHAPTER 1

Introduction

In this thesis, we will explore the realm of Bayesian thinking and its application to an existing problem. The problem in question is that of unfolding, i.e. interpreting the output from an imperfect detector and attempting to reconstruct the true signal. No detectors are perfect, and determining the exact source of a specific output is of great interest for the experimental process. More specifically, the aim is to investigate the unfolding of γ-ray spectra using Fully Bayesian Unfolding (FBU) [1], and compare with the Folding Iteration Method. This is the current method used for unfolding by the nuclear physics group at UiO [2].

On nuclear physics spectra, FBU is a mostly unexplored method, one of the first (if not the very first) attempts being done by Valsdóttir [4]. In her thesis, she performed FBU on spectra from the ^{28}Si(p,p'γ) reaction using the PyFBU package [3] and observed similar results to the folding iteration method. When including the background data, it was found that FBU showed some improved accuracy versus the folding iteration method which requires a background-subtracted input, as compared to the built-in background handling in FBU.

In the present work we explore the method of FBU further. Firstly, we dive into the inner workings of the PyFBU package in order to make the unfolding process more transparent and the results more understandable. We identify the Bayesian terms, particularly the likelihood, which may be the most abstract concept of Bayes theorem, see equation 2.3. Secondly, we perform modifications to PyFBU in order to increase its flexibility towards a larger variety of problems. This is mainly done by giving the user the ability to directly define the prior distribution, an essential part of Bayesian statistics.

Continuing on to actual unfolding of experimental spectra, we take a look at two different sources of data. First, we utilize a new response matrix for the first excited state of

the ^{28}Si spectrum and compare the corresponding FBU results with Valsdóttir's. The new response matrix is found in the OMpy library [5] and represents the actual experimental conditions more closely than the previous matrix used by Valsdóttir. Secondly, we perform FBU on entirely new data from the ^{146}Nd(p,p'γ) reaction. This data contains spectra which exhibit more complex structures than the first excited state of ^{28}Si and will show the generality of FBU. The results are directly compared with the folding iteration method and evaluated visually and numerically.

The different parts of the thesis is presented as follows:

- In part I, the general theory of Bayesian statistics is described, including the fundamental Bayes' theorem and related concepts.
- In part II, firstly the folding iteration method and its components is described. Thereafter, we describe FBU and its application of Bayes' theorem, with a discussion of the Bayesian terms. From there, we discuss how to summarize and visualize the results from FBU and lastly how to evaluate their accuracy with certain error metrics.
- Next, we show the implementation of FBU in part III, with our suggested modifications to PyFBU in order to increase its flexibility and thus, possibly better its accuracy. Here, we also perform some investigative tests in order to verify our assumptions about the package, more specifically about the relatively mysterious likelihood.
- Part IV shows the results on the experimental spectra. First we compare the results of using the new response matrix on the ^{28}Si first excited state, with the result from Valsdóttir. Then, we focus on the ^{146}Nd data and evaluate the results from unfolding both the first excited state, as well as higher energy levels. We compare the results with the folding iteration method in OMpy using the mentioned error metrics.
- Finally, in part V, we summarize our discussion and make our conclusions, as well as discuss possible future work related to the topics discussed in the thesis.

Part I

Theory

CHAPTER 2

Bayesian statistics

> Probability theory is nothing but common sense reduced to calculation.
>
> Pierre-Simon Laplace [6]

Sivia's book *Data Analysis - a Bayesian tutorial* [6] is a great read, and provides the theoretical foundation for the majority of topics discussed in this chapter.

2.1 Bayes' theorem

The realm of probability is commonly considered to be split into two main camps of interpretation:

- The frequentist view, which defines probability as a number representing the relative frequency of which an outcome occurs, after performing an infinite amount of experiments. This view only considers probability of the data given a hypothesis, and does not allow talking about a probability of the hypothesis itself.
- The Bayesian view, which defines probability as a degree of belief. Applying a Bayesian probability means to make a statement about what the outcome of an experiment will be, and how certain we are. We are able to use whatever *prior* knowledge and experience (or lack thereof) we possess to make this statement, as well as making any changes depending on the result. Frequentists can only base such a statement on the result of the experiment itself, infinitely repeated to be certain.

As one might guess from the titles, we will be assuming the second interpretation, the Bayesian view, and explore how it is used to describe the happenings of nature. First, we consider probability theory and its basic algebra which includes the sum rule

$$P(X|I) + P(\overline{X}|I) = 1 \tag{2.1}$$

and the product rule

$$P(X, Y|I) = P(X|Y, I) \times P(Y|I). \tag{2.2}$$

Here P stands for probability, the bar "|" means "given" and $\overline{X}$ means "not X". Lastly, we have the symbol I, meaning all relevant background information. The sum rule can then be stated as "the probability of X being true plus the probability of X not being true, both given all relevant background, equals 1".

Using the product rule, and the fact that $P(X, Y|I) = P(Y, X|I)$ we get the following.

$$P(X|Y, I) \times P(Y|I) = P(Y|X, I) \times P(X|I)$$

Rearranging this leads to *Bayes' theorem*

$$P(X|Y, I) = \frac{P(Y|X, I) \times P(X|I)}{P(Y|I)} \tag{2.3}$$

To get a clearer picture of the significance of Bayes' theorem, we can replace X and Y with *hypothesis* and *data*. $P(hypothesis|data, I)$ is then given the formal name *posterior probability*, $P(data|hypothesis, I)$ is called the *likelihood* and $P(hypothesis|I)$ is called the *prior probability*, representing our knowledge about the truth of the hypothesis before any data has been analysed. The term in the denominator, $P(data|I)$, often called the *evidence*, is in many cases not shown, due to it often being absorbed by a normalization constant. We can then replace the equality sign with a proportionality.

$$P(hypothesis|data, I) \propto P(data|hypothesis, I) \times P(hypothesis|I) \tag{2.4}$$

In summary, Bayes' theorem describes a learning process, showing how a probability should be augmented by the introduction of data.

Another useful result from using the sum and product rule is the *marginalization* equation

$$P(X|I) = \int_{-\infty}^{\infty} P(X, Y|I) dY \tag{2.5}$$

with a normalization condition

$$\int_{-\infty}^{\infty} P(Y|X,I)dY = 1. \tag{2.6}$$

The marginalization equation gives us the ability to integrate out so-called nuisance parameters, that is parameters of no interest to the question we are investigating, such as parameters describing experiment backgrounds and measurement byproducts. These rules of probability are generally applicable and provide a strong foundation for tackling data analysis problems. [6]

2.2 Parameter estimation

We will now look at the act of estimating a single parameter using Bayes' theorem, such as the mass of a planet, or the charge of the electron. We will firstly go through the example of deducting the fairness of a coin. This can be represented by the *bias-weighting* H. $H = 1/2$ will mean the coin is fair, while $H = 1$ and $H = 0$ means the coin is showing only heads or tails every flip. This value is continuous on the range $[0, 1]$, and $P(H|\{data\}, I)$ describes how much we believe H to be true. For a range of H-values, $P(H|\{data\}, I)$ is a *probability density function* (pdf). To find this, we use Bayes' theorem.

$$P(H|\{data\}, I) \propto P(\{data\}|H, I) \times P(H|I) \tag{2.7}$$

We can, if needed, find the normalization constant using equation (2.6). To express ultimate ignorance, we can assign a flat pdf for the prior.

$$P(H|I) = \begin{cases} 1 & 0 \leq H \leq 1 \\ 0 & \text{otherwise} \end{cases} \tag{2.8}$$

meaning we assume every value of H to be equally probable. Assuming each flip is an independent event, the likelihood function takes the form of the binomial distribution.

$$P(\{data\}|H, I) \propto H^R(1-H)^{N-R} \tag{2.9}$$

where R is the number of heads and N is the number of flips.

Plugging (8) and (9) into Bayes' theorem results in the posterior probability, the shape of which varies significantly for the first few data points. When the number of data increases however, the pdf becomes sharper and converges to the most likely value. The choice of prior becomes mostly irrelevant when we have a large of number of data, as the majority of propositions will converge to the same solution, but the speed of convergence may vary. A very confident, but wrong prior will often approach the correct solution more slowly than an ignorant one.

2.2.1 Summarizing distributions

One way to summarize the posterior pdf is with two quantities: the best estimate and its reliability. The best estimate is given by the maximum value of the pdf

$$\left.\frac{dP}{dX}\right|_{X_O} = 0 \tag{2.10}$$

where X_O denotes the best estimate. To make sure we have a maximum, we also need to check the second derivative

$$\left.\frac{d^2P}{dX^2}\right|_{X_O} < 0. \tag{2.11}$$

This is assuming X is continuous. If not, the best estimate will still be the value corresponding to the max of the pdf.

The reliability of the best estimate is found by considering the width of the pdf about X_O. We take the logarithm of the pdf as this varies more slowly with X, making it easier to work with.

$$P_{ln} = ln[P(X|\{data\}, I)]. \tag{2.12}$$

Doing a Taylor expansion about X_O and using the condition

$$\left.\frac{dP_{ln}}{dX}\right|_{X_O} = 0, \tag{2.13}$$

which is equivalent to (10), leads to

$$P(X|\{data\}, I) \approx A \exp\left[\frac{1}{2}\left.\frac{d^2P_{ln}}{dX^2}\right|_{X_O}(X - X_O)^2\right]. \tag{2.14}$$

Here, we only show the dominating quadratic term of the expansion, with A being a normalization constant. We have now approximated our pdf by the *normal distribution*, typically expressed in the form

$$P(x|\mu, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left[-\frac{(x-\mu)^2}{2\sigma^2}\right], \tag{2.15}$$

with μ being the mean value. The parameter σ describes the width of the distribution and is related to P_{ln} through

$$\sigma = \left(-\left.\frac{d^2P_{ln}}{dX^2}\right|_{X_O}\right)^{-1/2}. \tag{2.16}$$

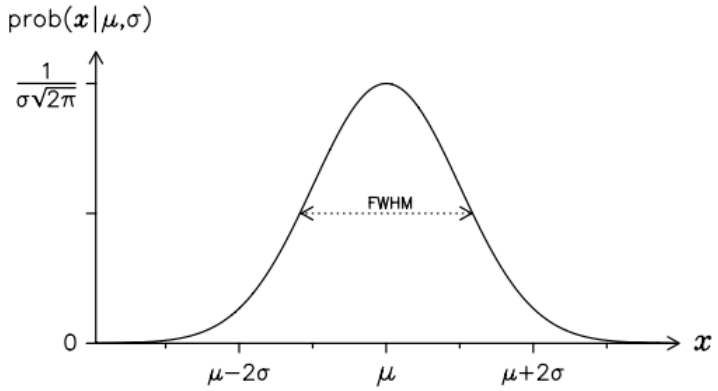


Figure 2.1: The normal distribution with a maximum at $x = \mu$ and a full width at half maximum (FWHM) of 2.35σ. (Sivia, 2006, p. 22) [6]

Combining this with the best estimate allows us to summarize the distribution:

$$X = X_O \pm \sigma. \tag{2.17}$$

In this context, σ is often called the *error-bar*. By calculating the integral of the normal distribution in this range, we get approximately a 68% chance that X lies within $X_O \pm \sigma$ and approximately 95% within $X_O \pm 2\sigma$.

2.2.1.1 Asymmetric pdfs

The error-bar needs a symmetric pdf to be valid, something that is often not the case. This is solved by replacing the error-bar with a *credible interval* as a measure of reliability. It is defined as the shortest interval that encloses a given percentage of the total probability, conventionally set to 68% or 95%. In the case of 95%, we find X_1 and X_2 such that

$$P(X_1 \leq X \leq X_2|\{data\}, I) = \int_{X_1}^{X_2} P(X|\{data\}, I)dX \approx 0.95, \tag{2.18}$$

assuming the pdf is normalized.

In the case of an asymmetric pdf, we may consider using the *mean* or *expectation* as the best estimate. This quantity takes the skewness of the pdf into account, and is given by

$$\langle X \rangle = \int XP(X|\{data\}, I)dX. \tag{2.19}$$

If the pdf is not normalized, we also need to divide the right-hand side by $\int P(X|\{data\}, I)dX$.

If the pdf is *multimodal*, meaning it has multiple maxima, it becomes more difficult to calculate a best estimate and its reliability. If one maximum is much greater than the

others, we can ignore those other contributions and focus on the largest. However, if multiple peaks are of similar size, we would be better off displaying the pdf itself.

Another common pdf is the Cauchy distribution, shown below.

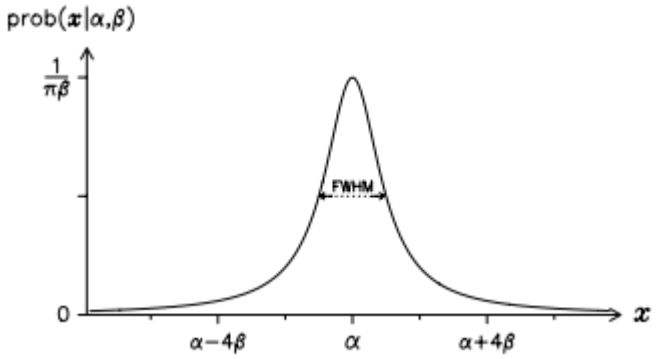


Figure 2.2: The Cauchy distribution, symmetric about $x = \alpha$ and has a FWHM of 2β. (Sivia, 2006, p. 31) [6]

This distribution has very wide wings and is in this case given by

$$P(x|\alpha, \beta, I) = \frac{\beta}{\pi[\beta^2 + (x - \alpha)^2]} \tag{2.20}$$

2.3 Notation

The terms in eq. 2.3 are notationally very similar, forcing us to keep track of vertical bars and the order of parameters. Thus, for ease of reading, the following notation will be used for the Bayesian terms from now on:

- The posterior distribution: $P(H|D)$
- The likelihood: $L(H)$
- The prior distribution: $\pi(H)$

Bayes' theorem then takes the following form:

$$P(H|D) \propto L(H) \times \pi(H). \tag{2.21}$$

Part II

Methods

CHAPTER 3

Unfolding Methods

3.1 What is unfolding

Unfolding is the act of reconstructing a true signal, based on observations made by an imperfect detector. It can be described as an inverse problem, where the observed data, or folded spectrum, is given by:

$$f = \boldsymbol{R}u \tag{3.1}$$

where f and u are vectors containing count values for corresponding channels of the detector. f and u represent the observed data and the unfolded spectrum (expected true spectrum), respectively. $\boldsymbol{R}$ is the response matrix, i.e. the impact on the data by the detector imperfections. The solution is not straightforward, as multiplying with $\boldsymbol{R}^{-1}$ leads to fluctuations because we cannot assume the observed data equals the *expectation values* for the data. Statistical fluctuations in the data is assumed to come from a real structure in the true spectrum and will be magnified [7]. We will now discuss two different methods for unfolding, firstly the default method in use at the nuclear physics group at UiO [2]. Secondly, we jump into the focus of this thesis, Fully Bayesian Unfolding.

3.1.1 Data source

The data used in this thesis is produced from γ-ray spectra observed by the OSCAR detector array. The detector array consists of multiple $LaBr_3$:Ce detectors which detect γ-rays through different interactions. A detailed description of the experimental setup is given by Valsdóttir [4]. The resulting data is on the form of a 'raw' matrix, with binned γ-energies on the x-axis, and corresponding source excitation energies on the y-axis. The raw matrices used in this thesis are provided by Ann-Cecilie Larsen.

3.1.2 The response matrix

The response matrix represents the detector performance, i.e. describing how the detector may redistribute counts to other areas in the energy range. The response matrix has elements given by:

$$R_{tr} = P(\text{reconstructed in bin r} \mid \text{true in bin t}) \tag{3.2}$$

[8]. This can be read as the probability of observing an event in energy bin r, given the true event happening in bin t. In a nutshell, the response matrix describes how a signal is smeared over the other bins in the spectrum. The response matrices used in this thesis are found in the OMpy library [9][10].

3.2 The folding iteration method

The following section describes the methods developed by Guttormsen et al. [2].

The folded spectrum f is on the form

$$f = \boldsymbol{R}u, \tag{3.3}$$

where $\boldsymbol{R}$ is the response matrix and u is the expectation values for the true spectrum. The iterative method can then be described in 4 parts.

- First we use the observed spectrum $\boldsymbol{D}$ as an initial guess for a trial spectrum u_0,
  $u_0 = \boldsymbol{D}$
- We then fold this with the response matrix,
  $f_0 = \boldsymbol{R}u_0$
- The difference between the folded and the raw spectrum is calculated and added to the initial guess, and we end up with the next trial spectrum,
  $u_1 = u_0 + (\boldsymbol{D} - f_0)$
- This is then repeated according to the following iteration scheme,
  $u_{i+1} = u_i + (\boldsymbol{D} - f_i)$

This method is performed until $f_i \approx \boldsymbol{D}$ within the experimental uncertainties [2]. It is important to note that for each new iteration, the oscillations between channels increase, as the solution approaches the inverted matrix solution $u = \boldsymbol{R}^{-1}\boldsymbol{D}$, which exhibits large oscillations [8][2].

3.2.1 The Compton subtraction method

As the resulting spectrum from the folding iteration method often contains some degree of fluctuations, the Compton subtraction method is performed to obtain a significantly more stable spectrum.

The first step is to define a new spectrum v as the observed data excluding the Compton contribution:

$$v = p_f u + w\,, \tag{3.4}$$

where u is the spectrum obtained from the folding iteration method, which multiplied with p_f gives the full-energy contribution (the tallest peak in the spectrum, representing a complete photoelectric absorption of a γ photon). The remaining contributions are contained in $w = u_s + u_d + u_a = p_s u + p_d u + \sum p_{511} u$, representing single escape, double escape and annihilation (note the missing Compton contribution "u_c"). These other possible contributions to the spectrum represent phenomena which may hinder a full photon absorption, like pair production, scattering and the mentioned annihilation. To match the observed energy resolution, each contribution is then smoothed with a Gaussian function. Next, we subtract this from the raw spectrum to obtain the Compton background spectrum:

$$c = \boldsymbol{D} - v\,. \tag{3.5}$$

This spectrum may exhibit significant oscillations, and is thus further smoothed. This smoothing carries a low risk of loss of important information due to the nature of the spectrum not containing any sharp, narrow peaks. After this smoothing procedure on the individual contributions, we now "return" to the unfolded spectrum like so:

$$u = \frac{\boldsymbol{D} - c - w}{p_f}\,. \tag{3.6}$$

Finally, to get closer to the true number of events, we correct for the total detector efficiency:

$$U = \frac{u}{\epsilon_{tot}}\,. \tag{3.7}$$

This final spectrum shows higher stability compared to the result of the folding iteration method, while keeping similar statistical fluctuations to the raw spectrum [2].

3.3 Fully Bayesian Unfolding

Choudalakis created the method of *Fully Bayesian unfolding* (FBU) by applying Bayesian thinking to the problem of unfolding. He states that the method provides the ability to observe all possible answers to a given unfolding problem via the posterior distribution, as opposed to other methods which result in point estimates of one of the possible answers through iteration. Below, we describe FBU and its components as developed by Choudalakis [1].

Bayes' theorem succinctly describes what we are asking for in the problem of unfolding, showing the relation between the expected truth spectrum $\boldsymbol{T}$, and the data we have obtained $\boldsymbol{D}$.

$$P(\boldsymbol{T}|\boldsymbol{D}) \propto L(\boldsymbol{T}) \times \pi(\boldsymbol{T}) \tag{3.8}$$

The expected truth spectrum $\boldsymbol{T}$ and the raw spectrum $\boldsymbol{D}$ are binned with N_t and N_r bins, respectively. In this thesis we operate with $N_t = N_r = N$ as we do not expect the number of energy bins to change during an experiment, but the mathematics do permit such a difference either way. Each bin in $\boldsymbol{T}$ is assigned a prior probability distribution $\pi(\boldsymbol{T})$, describing our belief of the number of events expected to be present. We assume the data follows a Poisson distribution, meaning

$$L(\boldsymbol{T}) = \prod_{r=1}^{N_r} \frac{f_r^{D_r}}{D_r!} e^{-f_r} \tag{3.9}$$

where

$$f_r = \sum_{t=1}^{N_t} T_t \times R_{tr}. \tag{3.10}$$

This f_r parameter represents the expectation value for the number of reconstructed counts in bin r. Here, R_{tr} is the element of the response matrix $R_{N_t \times N_r}$, corresponding to the probability that an event produced in the truth bin t is observed in the reconstructed bin r: $P(r|t)$. If we wish to include the background spectrum, all we have to do is add it to the sum:

$$f_r = B_r + \sum_{t=1}^{N_t} T_t \times R_{tr}. \tag{3.11}$$

The next step is to employ a sampling scheme of the parameter space, usually a MCMC algorithm, to calculate $L(\boldsymbol{T}) \times \pi(\boldsymbol{T})$ and arrive at a posterior distribution per bin in the expected truth spectrum.

With the likelihood function defined, the next step is to numerically sample the N_t-dimensional parameter space of possible truth spectra $\boldsymbol{T}$, usually with a MCMC algorithm, to obtain samples from the posterior distribution $P(\boldsymbol{T}|\boldsymbol{D}) \propto L(\boldsymbol{T})\pi(\boldsymbol{T})$. By histogramming these $\boldsymbol{T}$ samples in different ways, we can visualize the final result in terms of one- or two-dimensional marginal posterior distributions. In particular, we will present our main results in terms of 1D posterior distributions, one for each bin in the truth spectrum.

3.3.1 Priors

As mentioned above, we assign a prior $\pi(T_t)$ for the truth expectation T_t in each bin of the spectrum. This means we are choosing the exact probabilities of T_t values we believe to be possible for that bin, independent of other bins. In fact, since the prior has to equal 0 outside its defined range, we say that T_t values beyond these boundaries are impossible. Since we are dealing with physical experiments, these boundaries need to be finite, and thus we are forced to restrict the realm of possibilities to whatever we deem reasonable. There is practically an infinite number of choices one can make for assigning a prior, depending on what knowledge one has beforehand. We will now take a look at two possible prior distributions.

3.3.1.1 Uniform prior

If one wishes to make only weak assumptions about the truth, a *uniform* prior is suitable. The pdf of the uniform distribution is given as:

$$\pi(x) = \begin{cases} \frac{1}{b-a} & \text{for } a \leq x \leq b \\ 0 & \text{otherwise} \end{cases} \tag{3.12}$$

This flat distribution assigns equal probability to every outcome in the space of possibilities. The only assumption to be made here is determining the boundaries on this space. Complete ignorance would strictly be represented with a uniform prior without any boundaries. This would mean we believe all numbers on the interval $(-\infty, \infty)$ to be equally likely in a one-dimensional space. Such a space is of course not possible to explore completely, and otherwise extremely large limits will be computationally unfeasible. This is especially true considering the fact that many problems are complex and demand multidimensional parameter spaces. One thing to note is that unfolding in physics is often related to physical experiments pertaining to the counting of a number of events measured by a detector. In these cases, the existence of negative counts is unphysical, meaning a lower prior limit can safely be set to 0.

Choosing the upper limit is not as straightforward. The ideal choice would be the largest possible limit that still allows for reasonable computational performance. Of course, if we have some knowledge about the size and location of the domain of the possible truth-values, there is no need to pick a limit located significantly beyond this domain. Computational resources are wasted if spent on exploring a region we strongly believe will not improve our estimate. A good check is to fold the upper prior limit with the response and make sure the raw data is contained within.

In this thesis, an upper prior limit of 10 times the raw data will be used for the uniform prior. In other words, we believe that the true value must be contained within an area with size and limits dependent on the observed value.

- Image of uniform distribution

3.3.1.2 Log-uniform prior

Another prior distribution we will use is the *log-uniform* distribution, also called the reciprocal distribution ref?. This distribution has the characteristic that its logarithm is uniformly distributed. What this means for our prior belief is that each order of magnitude is given equal probability. In the case of a logarithm with base 10, we say that it is equally probable for our value of interest to lie between limits $a = 10^0$ and $b = 10^1$, as between $a = 10^6$ and $b = 10^7$, even though the second range is much larger. The pdf of the log-uniform distribution is defined as:

$$\pi(x) = \frac{1}{x \ln(b/a)} \quad \text{for } a \leq x \leq b \text{ and } a > 0. \tag{3.13}$$

Ref?

- Image of logarithmic distribution

Using the log-uniform distribution allows us to define a very large range of possible truth-values while keeping a high probability for values close to 0. The amount of counts per experiment is finite, and in cases with significant differences between peaks and valleys, we do not want to 'dampen' these by probabilistic distribution of counts into bins which should be containing none.

Say we know there exists one or multiple peaks in our truth-spectrum consisting of a very large amount of counts, i.e. $\sim 10^{10}$. If we also know that other bins in the spectrum should have close to 0 counts, how do we make sure both of these conditions are met? If

we were to use a uniform prior between 0 and 10^{10}, we would firstly have an incredibly large space to explore, with 10^{10} possible values for each bin in the spectrum. Secondly, the probability of sampling a value close to 0 would be very small. Let's say that any value between 0 and 10000 is considered 'close' to 0, which itself seems very imprecise. According to the **uniform** prior, the probability of the true value being 'close' to 0 is thus:

$$P(0 \leq T_t \leq 10^4) = \int_0^{10^4} \frac{1}{10^{10} - 0} dT_t = \frac{10^4}{10^{10}} = 10^{-6} = 0.0001\%, \tag{3.14}$$

and the same result holds for values equally 'close' to the maximum of 10^{10}. This low probability means we might run out of computational resources long before our algorithm gets to explore those areas.

Now, if we instead use the **log-uniform** distribution as our prior, the probability of a value close to 0 is much higher, while still allowing for those tall peaks.

$$P(1 \leq T_t \leq 10^4) = \int_1^{10^4} \frac{1}{T_t \ln(10^{10}/1)} dT_t = 0.4 = 40\%. \tag{3.15}$$

Note here that we integrate from a count value of 1 instead of 0, as the distribution is undefined at $T_t = 0$. Since the logarithm of this function is uniformly distributed, looking at the exponents will give a good indication of the probability value. In the simple case above, the exponents are 4 and 10, $4/10 = 0.4 = 40\%$. Similarly, we can estimate the probability of the area between 10^6 and 10^{10} to be $(10 - 6)/10 = 0.4 = 40\%$, the same result as in eq. (3.15), even though this range is much larger. The estimation turns out to be correct when we perform the proper calculation:

$$P(10^6 \leq T_t \leq 10^{10}) = \int_{10^6}^{10^{10}} \frac{1}{T_t \ln(10^{10}/1)} dT_t = 0.4 = 40\% \tag{3.16}$$

In summary, when we believe the truth spectrum to exhibit a very large difference between minima and maxima, the log-uniform distribution is a good candidate for the prior. This way, we increase the chance of reaching the proper relation between peaks and valleys, at the price of lower precision for higher values. The high probability for values close to 0 will also allow for less distribution of counts into bins where there should be almost none, due to the nature of probability. Should the true spectrum instead be composed of peaks with similar magnitudes and smaller differences, a uniform prior may be better suited.

For the cases examined in this thesis, similar limits to the uniform distribution has been chosen as the range has been sufficiently wide. Expanded limits have been found not to

make a significant impact on the unfolded results, but different spectra may warrant such changes. The lower limit cannot be set to zero for which the distribution is not defined, thus we set it to 10^{-1} to be close enough. The upper prior limit is set to the following:

$$\text{upper} = \max(10 \times \text{raw}, 100 \times \text{lower} = 100 \times 10^{-1} = 10). \tag{3.17}$$

This makes sure the upper limit is never smaller than the lower in cases with low raw values. The implementation of the prior distributions is discussed in part III.

3.3.2 Likelihood

As mentioned above, the likelihood used in FBU is given by the Poisson distribution, given by eq. (3.9). It is important to note that the likelihood is a function rather than a pdf, meaning it does not necessarily integrate to 1.

When we assign the prior probability for our problem, we do so on a per-bin basis, meaning we end up with a set of N independent distributions, represented by histograms, each describing the probability of possible truth-values for one bin. The same applies to the posterior probability, the only difference being the histograms having different shapes, due to the fact that we have been provided new knowledge from the data. This reshaping stems from the multiplication of the prior with the likelihood. One might then be tempted to construct a 1-dimensional Poisson distribution for a given bin, multiply with the prior and call this product the posterior. This is incorrect due to two reasons:

- Firstly, one must remember that the posterior is only proportional to the product likelihood$\times$prior, meaning it may need normalization to be a proper pdf and integrate to 1.
- Secondly, and most importantly, the likelihood cannot be assigned on a per-bin basis like the prior, as it does not consist of N independent distributions of which we can aggregate.

The likelihood is an N-dimensional function dependent on the total collection of data as well as the entire response matrix. For a given bin, a 1-dimensional Poisson distribution based on that data does not equal the contribution from the actual likelihood in that bin (this has been tested in appendix V, unless our spectrum consists of only one bin. This also means we have no easy way of plotting the likelihood, should we wish to compare with the prior and posterior in a model test, unless we restrict the spectrum to contain a maximum of $N = 3$ bins and plot the complete multidimensional function. Most experiments

are conducted with many more bins than this, however there is still some value to be had from performing such a visualization. Mainly, this will help us achieve an increased understanding of the process of Fully Bayesian Unfolding, its components, as well as the inner workings of the PyFBU-package. Due to its low amount of documentation and several layers of abstraction, both in itself and through PyMC3 and Theano, the important elements are not immediately apparent. The symbolic variables and objects, while computationally efficient, do not allow for simple printing or plotting during intermediate steps. Understanding why the results appear as they do is therefore not straightforward, but we are able to use what we know about the prior and likelihood. The prior is defined by the user, but the likelihood is not. In fact, there is no simple multiplication of prior and likelihood performed in the source code of PyFBU at all. This is due to the way Bayes' theorem is being implemented. In the analythical formula, eq. (3.8), we see the posterior as a rescaled version of the prior, through multiplication with the likelihood. In the code, there is instead of this product, a definition of the space for which a sampling algorithm explores. By PyMC3 convention, this space is defined as the likelihood evaluated on the prior, i.e. plugging the prior-values in for T_t in eq. (3.10). This is shown by creating an object of class 'Poisson' from the PyMC3 package, with the prior (folded with the response) and data as arguments. The class refers to one of the built-in distribution classes found in PyMC3. To make sure we are correct about this indeed representing the likelihood, we will want to compare the resulting posterior with a Poisson distribution we construct ourselves, multiplied with the prior. If the posterior has the same shape and location as this product, we have verified our knowledge of the likelihood and its parameters. As mentioned above, we will have to plot the entire multidimensional Poisson distribution to show a correct picture. Therefore, a 2-bin constructed spectrum will be used for this purpose. This implementation will be discussed in part III.

- Modified Poisson to take into account the total amount of counts

With a greater understanding of what happens under the hood, we can experiment with modification of the likelihood. We believe the Poisson distribution to be a good representation of the data, and will not switch it out completely. Such a switch is possible through PyFBU and PyMC3, but we will only perform a modification of the standard Poisson.

3.3.3 Sampling

There are several sampling methods possible for the problem of unfolding, a common example being Markov Chain Monte Carlo (MCMC) algorithms such as the Metropolis-

Hastings algorithm. In the PyFBU-package, a variant of a Hamiltonian Monte Carlo (HMC) Markov Chain Monte Carlo algorithm is the default sampler. HMC aims to be much more efficient than regular MCMC algorithms by avoiding both sensitivity to correlated parameters and random walk tendencies [11]. A drawback to this is a significant sensitivity to step size as well as the number of steps, requiring manual tuning of these parameters. To circumvent this, Hoffman and Gelman created the No U-turn Sampler (NUTS), a variant of HMC which removes having to specify the number of steps. They also implemented an adaptive step size, meaning no manual tuning is necessary for running NUTS. Furthermore, they observed similar to better performance than other fine-tuned HMC algorithms [11]. The NUTS algorithm is implemented in the PyMC3 package [12] and is the default sampling algorithm in PyFBU. PyMC3 has several methods for initializing NUTS, the default being named 'jitter+adapt_diag'. Another initialization method which will be used in this thesis is called Automatic Differentiation Variational Inference (ADVI) [13]. In some cases, the use of this initialization will help when PyFBU would otherwise crash. This might be due to the 'jitter' part of the default method, which according to PyMC3, applies a "uniform jitter in [-1,1] as a starting point in each sampling chain" [12]. Negative numbers have shown to cause some issues in PyFBU, perhaps connected to the unphysicality of allowing negative counts. The results in this thesis have been produced with the ADVI initialization, due to no observed instances of crashing, and no discernible differences in results.

3.3.4 Posterior inference

Now that the unfolding has been performed, how do we interpret the resulting posterior distribution? While other methods may only return a point value, not necessarily accompanied by the uncertainties, FBU allows us to directly look at the final distribution per bin, and thus observe the result and its corresponding degree of belief. Of course, we are able to quantify these concepts in multiple ways. Here, we take a look at some of the methods of posterior inference.

3.3.4.1 Point estimates

Since we are dealing with 1-dimensional raw spectra visualized as plots with counts on the y-axis, and energy (bins) on the x-axis, it is desirable to represent the unfolded result the same way. The final output from FBU is a list containing N sets of posterior samples, allowing us to create N histograms representing the respective posterior distribution in

each bin. One can then simply stack these histograms to form a band through the entire energy range, where higher probabilities can be shown with higher color intensities. However, it is customary to operate with point values for the unfolded spectrum when performing further analysis, like the results OMpy supplies. Point estimates will also allow us to directly compare performance with the folding iteration method, of which point values are the only output. Finally, error metrics such as the R^2-score are evaluated on point values, and allows for a quantitative measure of performance. Since we have N posterior distributions, we create N point estimates which we aggregate and represent as the unfolded spectrum. We can use this to directly compare the **folded** representation with the original raw spectrum, as well as the regular representation with the true spectrum, should we possess it.

It is important to remember that a point estimate does not summarize an entire distribution, and may in many cases paint a wrong picture. In these cases, the fact that we can access and look at the complete posterior at any time may be the greatest advantage of using FBU.

We will consider two different point estimates, the posterior mean and median:

- Posterior mean: The mean of the posterior distribution which minimizes the mean squared error (MSE) [14].
- Posterior median: The median of the posterior distribution which minimizes the expected absolute error [14].
- A final possibility is the posterior mode. This represents the most likely value for the parameter in question, but does not take into account any skewness of the posterior nor the existence of multiple modes of similar magnitudes. Furthermore, the mode may be computationally expensive to calculate, often requiring approximation algorithms which may not always be correct.

3.3.4.2 Credible intervals

The credible interval is the Bayesian version of the frequentist confidence interval. It depends on the posterior and is defined as any interval that encompasses a certain percent of the posterior density. The difference between confidence and credible intervals is subtle, but not negligible. In the case of frequentist inference, the parameter in question, lets say θ, is treated as an unknown, but fixed value. The limits of the confidence interval are treated as random variables. Therefore, a confidence level of 95% means that for 100 re-

peated experiments, 95 of the confidence intervals will contain θ. Note that this does not mean there is a 0.95 probability of finding θ in every confidence interval. [15][16]

For bayesian inference however, the random-trait is switched, with the credible interval limits being fixed, and θ treated as the random variable. The credible interval takes our prior belief into account, while the confidence interval relies only upon the data. A 95% credible interval covers 95% of the posterior and can then be said to contain θ with a probability of 0.95. [15][16]

There are many types of credible intervals, the only requirement being that it covers a certain amount of area of the posterior. Some examples of ways of constructing credible intervals are:

- Using the posterior mean as the interval center.
- Making sure the probability of being outside the interval is equal on all sides (equal tailed).
- Making the interval as narrow as possible, the Highest Posterior Density interval (HPD). This will include the most likely values, as well as the mode of the posterior if it is unimodal.

We will be using the HPD interval which, together with the point estimates mentioned above, will give a solid estimate of the true energies of the γ-ray spectrum.

3.3.4.3 Variance

Each bin in the truth space are assigned a set of samples approaching the posterior distribution for that bin. We calculate the variance of these samples for each bin and take the mean, to represent a mean posterior spread for the whole unfolded spectrum.

3.4 Error metrics

Now that we have a candidate unfolded spectrum consisting of expectation values for the counts in each bin, we need to assess the accuracy. If we should possess the true spectrum, we can directly compare our unfolded result with this. Usually however, we do not have the true spectrum, only the observed data. The solution is to refold our result with the response matrix, generating a candidate for the observed spectrum. Doing this assumes the response matrix perfectly represents the detector attributes, which will likely lead to

some errors. Keeping this in mind, we use the following error metrics for comparing our model with the observed data.

3.4.1 Mean absolute error (MAE)

Given an estimated spectrum $\tilde{y}$ and an observed spectrum y, the mean absolute error is given by:

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \tilde{y}_i| \tag{3.18}$$

[17]. In our context, $\tilde{y}$ is the unfolded result refolded with the response matrix to estimate the observed data y. MAE is a simple measure of average error, allowing us to see how many counts we expect to be off the mark per bin. The lower this value, the better model, with 0 signifying a perfect match. Note that this does not take scale into account, as it does not measure relative difference. A model may output a value of $\tilde{y}_i = 90$ where the observed value is $y_i = 100$, while also outputting $\tilde{y}_i = 900$ for an observed value $y_i = 1000$. While the relative differences are the same, the MAE of the second case is 10 times larger than the first, falsely pointing to a worse accuracy. Therefore, we must take care to not blindly trust the MAE when comparing model accuracy on different scales. The MAE remains a good metric as long as we make sure to examine it within the context of scale.

3.4.2 R^2-score

Another metric of how well our estimated spectrum fits with the observed data is the R^2-*score*, also called the coefficient of determination. It is given by:

$$R^2(y, \tilde{y}) = 1 - \frac{\sum_{i=1}^{n}(y_i - \tilde{y}_i)^2}{\sum_{i=1}^{n}(y_i - \overline{y})^2}, \tag{3.19}$$

[18] where $\overline{y}$ is the mean of the observed spectrum. The R^2-score is a measure of the overall similarity between the estimate and the observed, a perfect match resulting in a score of 1. The R^2-score is scale invariant, meaning the mentioned limitation of MAE does not apply. This allows for direct comparison of model accuracy on different data sets, without the need to worry about scale differences. What constitutes as a 'good' R^2-score depends on the case and data, but in general the closer to 1, the better.

3.4.3 Residual plots

The MAE and R^2-score are summarizing metrics, boiling all errors down to a single number. They do not describe the actual error for each bin in the spectrum, nor if the total error is spread out over many bins, or focused around a few. To directly observe the error for each bin, we plot the residuals, which is simply given by the difference $y_i - \tilde{y}_i$. This way, we can determine the per-bin model accuracy and assess if there are any local dependencies affecting the total result.

3.4.4 Relative uncertainty plots

An interesting way to assess the uncertainty of the results can be by plotting the relative uncertainties for the unfolded spectrum. Since we expect a statistical variation already present in the data, we can investigate the systematic discrepancies by calculating the following:

$$\frac{\text{raw - mean(folded) - background}}{\sigma_{\text{tot}}}, \tag{3.20}$$

where $\sigma_{\text{tot}} = \sqrt{(\sigma^2_{\text{stat}} + \sigma^2_{\text{folded}} + \sigma^2_{\text{background}})}$. These are then the following uncertainties: $\sigma_{\text{stat}} = \sqrt{\text{raw}}$ (expected statistical variation for Poisson-distributed data), σ_{folded} is the standard deviation for the output samples from FBU, folded with the response, and $\sigma_{\text{background}}$ is the uncertainty in the background estimate. This last uncertainty is unavailable through the supplied background matrices at the time of writing, and will be set to 0. The final plot shows the discrepancy between raw and expectation measured in the number of standard deviations. Pure statistical variations should then result in swings of a couple standard deviations, while systematic discrepancies will more significantly affect this number. Credit to Anders Kvellestad for this formulation.

Part III

Implementation

CHAPTER 4

PyFBU and PyMC3

For the implementation of FBU, we utilize the PyFBU package, made by Gerbaudo, Helsens and Rubbo [3]. It is directly based on the original FBU article by Choudalakis [1], and it is made to receive the observed data, response matrix and prior limits as inputs. Using these and other optional inputs such as the background spectrum or a specified number of sampling steps, it performs the modeling and sampling with the PyMC3 package [12]. PyMC3 is a statistical modeling library which has built-in probability distributions usable for both priors and likelihoods, e.g. Uniform, Normal, Poisson etc., as well as truncated versions of some distributions. PyMC3 leverages the Theano package for array operations and linear algebra with the use of symbolic variables [19].

A significant effort has been made attempting to fully understand the packages used in this thesis. Externally, they (PyFBU and PyMC3) are moderately simple to learn and the experience of using them is quite pleasant, should you be content with the limits they pose and the results you receive. Due to the multiple layers of class-references and abstraction, it is not immediately apparent how the code relates to the analytical procedure of FBU, and hence, why the results look as they do. Now, the authors may never have intended for the direct manipulation of their source code, and to expect them to facilitate the possibility would be unfair, seeing as the likely intended use is completely functioning. However, we are in a search of a greater understanding of the process and the results. If we should receive a result we do not expect, we want to know how it came to be, as well as the ability to fine-tune the individual elements in the name of improvement. This chapter focuses on how PyFBU and PyMC3 is modified to achieve an increased versatility of the unfolding process, by enabling more direct control over the individual Bayesian terms.

The following sections are based on the source code of the PyFBU and PyMC3 packages, as well as the documentation available for PyMC3 [3][12].

4.1 Usage and modification

We start by describing the general setup of PyFBU and how PyMC3 comes into play with the unfolding process. In practice, PyFBU can be said to be used in the following way:

- Create an object of the PyFBU class
- Supply the necessary variables, i.e. observed data, response matrix and the upper and lower prior limits
- Optionally supply parameters such as background data, systematic uncertainties, number of sampling chains and steps, etc.
- Run unfolding

Many things happen behind the scenes which the user does not see, of which the main parts will be discussed here, starting with the prior distribution.

4.1.1 Creating the prior

An important part of the process is how the prior is defined. The user only has to supply the limits for the prior, not the distribution itself. This is due to the fact that the uniform distribution is the default prior in PyFBU, and we have some ability to change that by supplying a string with the name of a different distribution. This string is then used to collect one of the built-in distribution classes in PyMC3. There is a good variety of these classes representing many popular distributions used in statistics. Unfortunately, there are only four of these that accept the lower and upper limits as parameters, namely the classes `Uniform` (default), `DiscreteUniform`, `Triangular` and `TruncatedNormal`. If we are to attempt to input the name of any other distribution, we will be met by errors due to the `lower` and `upper` parameters. PyMC3 does however include a class `Bound` that takes the limit parameters and constrains any of the built-in distributions. The resulting distribution is not normalized anymore, and we are still restricted to using the distributions included in PyMC3, unfortunately ruling out the log-uniform distribution (part II, subsection 3.3.1).

Another possibility is the `DensityDist` class, made for supporting custom distributions. This requires supplying a function returning the log-probability of the distribution you want to use, as well as a `random` method if the distribution is to be sampled from. These functions are not straightforward to implement, as complex distributions may not

easily be represented as analytical formulas, and attempts to use this class have not been successful by the author of this thesis.

It is possible to create an entirely new distribution class that mimics the functionality of the other classes, which of course requires some effort to correctly implement the underlying methods, Theano logic and inheritance to parent classes. Luckily, we can avoid this due to the final possibility for implementing custom distributions; the `Interpolated` class. This class belongs to the collection of continuous distributions in PyMC3 and allows for a higher degree of user influence. The parameter inputs are two arrays, one containing a lattice of `x_points` (counts) and one containing the corresponding `pdf_points` (probability densities). The distribution is then generated by linear interpolation of these probabilities. The `Interpolated` class can be found here (link to the PyMC3 GitHub). Now we are free to design whichever distribution shape we want, by directly controlling the probability height for each count in our assigned prior range. Furthermore, the prior limits are collected from the first and last element of the `x_points` array, meaning the previously mentioned `lower` and `upper` arguments are unnecessary. Lastly, the resulting distribution is automatically normalized by PyMC3, allowing for the direct use as a prior in PyFBU. The `Interpolated` class is very promising, and the integration of this class into PyFBU will now be described.

When the `run()` method is called, a PyMC3 model is created, wherein the main math and sampling is performed. The prior distribution is created here, using an external `wrapper` method which returns a PyMC3 object representing a stack of N tensors, a prior distribution for each bin. The important part is found inside this `wrapper`, where the type of distribution is determined by the `priorname` parameter. Originally, this creates a new distribution object from PyMC3 for each bin in the spectrum, assuming there exists one for the current `priorname`, and that it can take the `lower` and `upper` arguments. All these distributions are then stacked and passed on to the main PyFBU program. The suggested changes to this method is including an alternative creation of distribution objects if `priorname = Interpolated`, where `lower` and `upper` are not used. The `x_points` and `pdf_points` arguments are passed through to the `other_args` dictionary by assigning them to the `priorparams` variable accessible in PyFBU. The original code and suggested changes are shown in figure 4.1 and 4.2, respectively.

priors_original.py > ...

```python
import pymc3 as mc

priors = {
    }

def wrapper(priorname='',low=[],up=[],other_args={},optimized=False):


    if priorname in priors:
        priormethod = priors[priorname]
    elif hasattr(mc,priorname):
        priormethod = getattr(mc,priorname)
    else:
        print( 'WARNING: prior name not found! Falling back to DiscreteUniform...' )
        priormethod = mc.DiscreteUniform

    truthprior = []
    for bin,(l,u) in enumerate(zip(low,up)):
        name = 'truth%d'%bin
        default_args = dict(name=name,lower=l,upper=u)
        args = dict(list(default_args.items())+list(other_args.items()))
        prior = priormethod(**args)
        truthprior.append(prior)

    return mc.math.stack(truthprior) #https://github.com/pymc-devs/pymc3/issues/502
```

Figure 4.1: The original prior-creation function in PyFBU [3], which returns a stack of N prior distributions, one for each bin in the data. The file has been renamed from `priors.py` to `priors_original.py` to distinguish from the modified file in figure 4.2.

priors.py > ...

```python
import pymc3 as mc
priors = {
    }

def wrapper(priorname='',low=[],up=[],other_args={},optimized=False):
    # Suggested changes are in blocks enclosed by #---# borders
    #-------------------------------------------------------------------------#
    # Get non-keyword arguments from other_args, return empty list if not found
    non_kwargs = other_args.get('non_kwargs', [])
    #-------------------------------------------------------------------------#

    if priorname in priors:
        priormethod = priors[priorname]
    elif hasattr(mc,priorname):
        priormethod = getattr(mc,priorname)
    else:
        print( 'WARNING: prior name not found! Falling back to DiscreteUniform...' )
        priormethod = mc.DiscreteUniform

    truthprior = []
    #-------------------------------------------------------------------------#
    # If the Interpolated class is to be used, use arguments from non_kwargs
    if priorname == 'Interpolated':
        for bin,(l,u) in enumerate(zip(low,up)):
            name = 'truth%d'%bin
            prior = priormethod(name, non_kwargs[0][bin], non_kwargs[1][bin])
            truthprior.append(prior)
    else:
    #-------------------------------------------------------------------------#
        for bin,(l,u) in enumerate(zip(low,up)):
            name = 'truth%d'%bin
            default_args = dict(name=name,lower=l,upper=u)
            args = dict(list(default_args.items())+list(other_args.items()))
            prior = priormethod(**args)
            truthprior.append(prior)

    return mc.math.stack(truthprior) #https://github.com/pymc-devs/pymc3/issues/502
```

Figure 4.2: Modified version of the `priors.py` file in PyFBU [3], shown in figure 4.1. The changes are shown in blocks enclosed by comment borders, allowing for the use of the `Interpolated` class in PyMC3. This enables a much greater freedom in designing the shape of the prior distribution, done by determining prior range and corresponding pdf-values in the users code and passing to PyFBU.

The user is now able to externally define the exact shape of the prior distribution which makes it possible to use an endless variety of distributions like the log-uniform distribution discussed in part II, subsection 3.3.1.

4.1.2 The likelihood

Picking up the thread from the creation of the prior, the next step in PyFBU is to incorporate the likelihood function. By PyMC3 convention this is done by creating another distribution object, from the Poisson class in our case, and evaluating it on the **folded** prior object. The reason for this can be understood by considering the spaces where our Bayesian terms are defined. The prior $\pi(\boldsymbol{T})$ is an assumption of the truth, meaning it is defined in the truth-space, it represents what we believe the true count-value can be in each bin. However, the Poisson distribution we define lives in the folded space, along with the observed data, as it is dependent on the folded parameter f_r, see eq. 3.9. This means it describes the spread of possible observed values given the already supplied observed values. We can transform this to a function in the truth space by specifying a truth-range and making the f_r-parameter dependent on that range, see eq. 3.10. Evaluating the likelihood on this truth-dependent range makes it describe the spread of possible **truth**-values that can lead to the given observed data. This Poisson distribution object evaluated on the folded prior will then represent a space of likelihood-weighted prior-values, which when sampled will result in the posterior distribution. After this, the next step in PyFBU is running the NUTS sampling algorithm, which when finished, outputs the final posterior samples for each bin in the spectrum.

4.1.2.1 The modified likelihood

keep or drop?

CHAPTER 5

Synthetic spectra

Here, we perform unfolding on a raw spectrum consisting of 2 bins, to display all Bayesian terms and their relation. Even though experimental spectra usually consist of a much larger amount of bins, we have chosen 2 to be able to show the likelihood properly. As mentioned in part II, the likelihood is an N-dimensional function which is not easily decomposed into 1-dimensional contributions (an attempt of this is shown in appendix V. Using 2 bins allows us to plot the likelihood in its entirety on the plane and compare with the posterior. We do this to firstly, display and compare the complete prior and posterior distributions, and the effect of the response matrix. Secondly, we wish to confirm what we believe to be correct of the built-in, symbolized likelihood, by comparing the posterior with an independent, externally constructed Poisson distribution. If we are correct, we expect our Poisson distribution to overlap and exhibit a similar shape as the posterior distribution from PyFBU.

We choose a simple true spectrum $\boldsymbol{T} = (120, 120)$ and construct a 2×2 response matrix with arbitrary values and normalized rows (preferably not the identity matrix, as no changes would happen when folding, i.e. a perfect detector):

$$R_{2\times 2} = \begin{bmatrix} 1 & 0 \\ 0.5 & 0.5 \end{bmatrix} \tag{5.1}$$

The response matrix is visualized in figure 5.1.

Chapter 5. Synthetic spectra

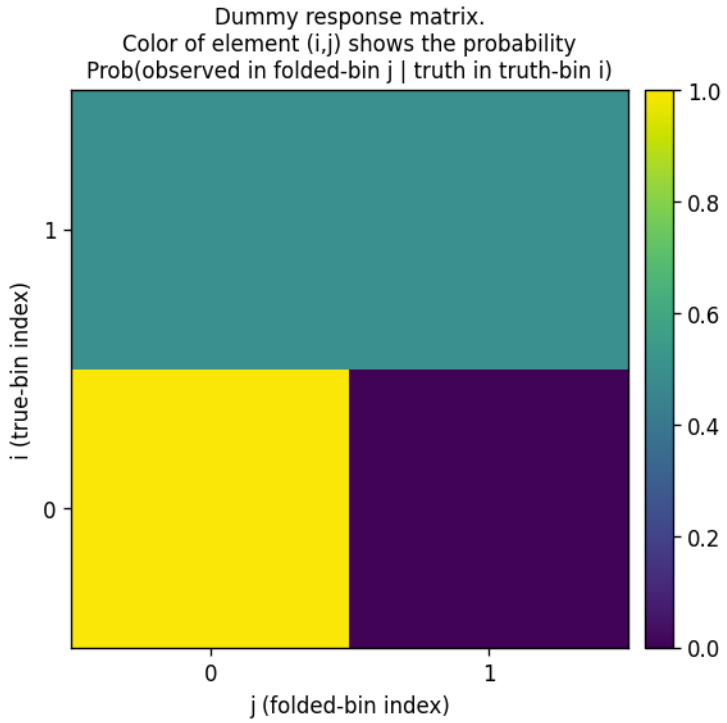


Figure 5.1: The constructed 2-bin response matrix. Note that when plotting the response, the origin for the indices is at the bottom left corner, as opposed to a mathematical matrix which has indices starting at the top left.

Next, we generate an 'observed' spectrum by folding the true spectrum with the response:

$$\boldsymbol{D} = \boldsymbol{TR} = \begin{bmatrix} 120 & 120 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0.5 & 0.5 \end{bmatrix} = \begin{bmatrix} 180 \\ 60 \end{bmatrix} \tag{5.2}$$

Note that by generating the data this way, we take away the inherent randomness of the response matrix. Since it consists of probabilities of an event being reconstructed in bin r given that it originated in truth-bin t, the data would not remain constant for repeat experiments; flipping 10 fair coins does not result in 5 tails and 5 heads every time. A more realistic generation of observed data can be performed by randomly sampling a value in each bin according to the given probabilities. The mean value of these samples will be the same as the result we get from direct multiplication with the response. That is sufficient in this case, where the focus is on investigating the Bayesian terms and verifying our knowledge of the likelihood, for which the actual values of the observed data does not matter. For both bins, we assign a uniform prior in the range $[0, 200]$ since we know the true values, and perform the unfolding. The resulting posteriors, together with priors and true values are shown in figure 5.2.

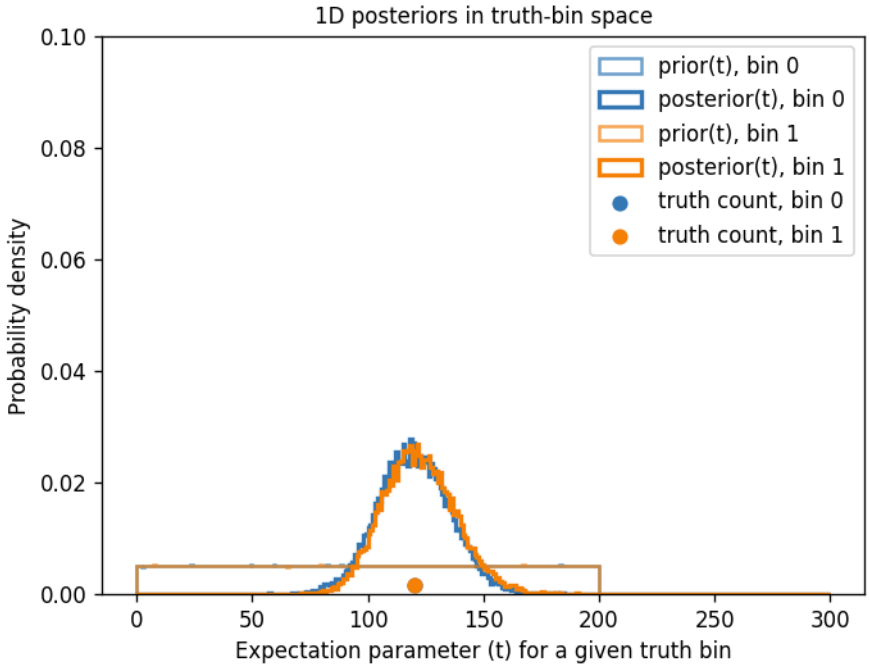


Figure 5.2: Prosterior distributions after unfolding a 2-bin constructed spectrum, along with corresponding priors and true values. Both posteriors point to the true value being located around 120, which is correct. The spread of the posteriors represents the uncertainty. We can also see that our uniform priors were suitable choices, as the true values are located within, and the posteriors are not truncated.

The plotted priors and posteriors are histograms consisting of samples from their respective distributions, and we may combine both bins to show the complete 2-dimensional histograms with 1 bin per axis. The complete prior is shown in figure 5.3.

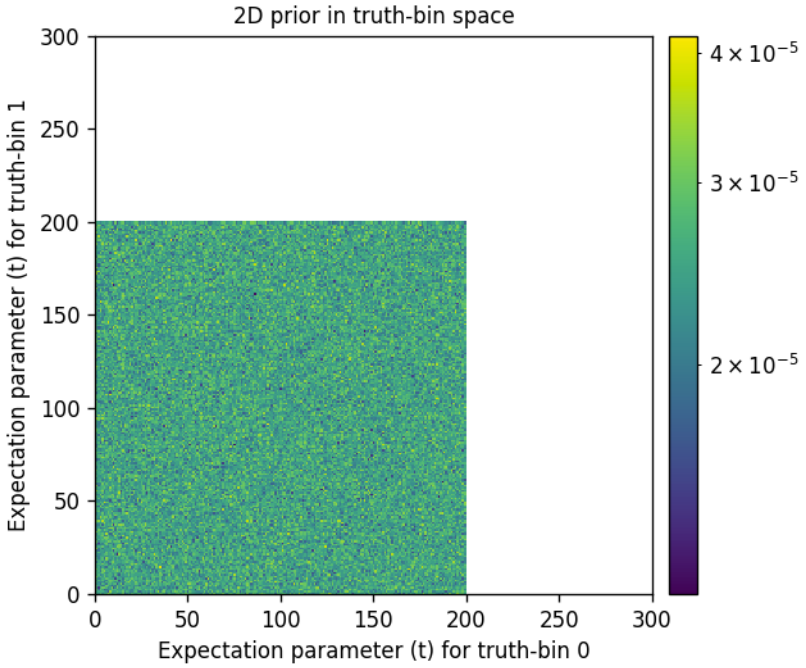


Figure 5.3: 2-dimensional complete prior distribution for both bins in the spectrum, consisting of random samples from the uniform distribution in the range $[0, 200]$.

Next, we plot our 2-dimensional Poisson distribution in the truth-bin space and see that it belongs to the same domain as the combined prior. This is shown in figure 5.4.

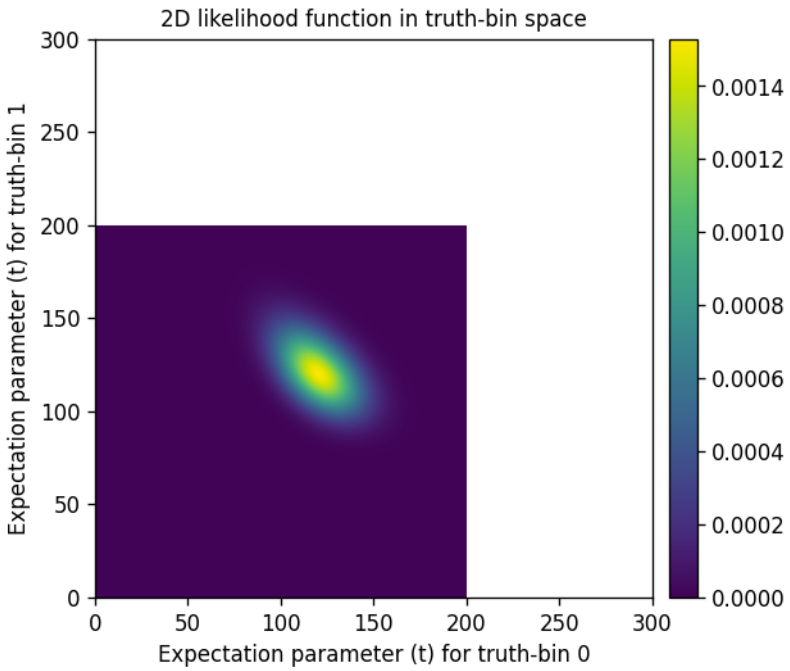


Figure 5.4: 2-dimensional Poisson distribution in the truth-bin space, dependent on the observed data and our defined ranges of possible true values. Note that this, as opposed to the prior and posterior, is a function defined over both dimensions, rather than a collection of samples.

With these pieces in place we can examine whether our Poisson distribution corresponds to the likelihood that is built into PyFBU. We do this by combining the posteriors in the same way we did with the priors (figure 5.3) and plot this together with the Poisson distribution, shown in figure 5.5.

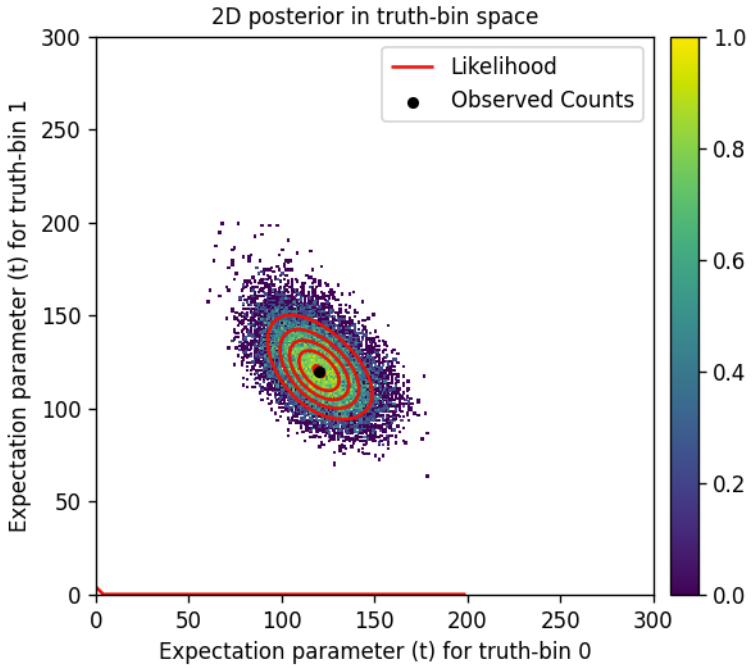


Figure 5.5: 2-dimensional complete posterior distribution for both bins, output from PyFBU, as well as contour lines from the 2-dimensional Poisson distribution shown in figure 5.4. Remember that the contours belong to a function we have defined based only on our available data and assumptions, meaning there is no connection to PyFBU. Yet, the contours exhibit a similar shape and location of the Poisson distribution as the posterior, indicating that our externally constructed function matches the internal likelihood of PyFBU. As mentioned in part II, subsection 3.3.2, we strictly have to compare the posterior with Poisson$\times$prior. However, since our prior is a uniform distribution, neither the shape nor the location of the posterior is affected by the prior, meaning we can directly compare with the likelihood candidate.

We see that our assumption about the likelihood was correct, and we have gained a stronger understanding of how the PyFBU-package is built up. This reduction of the black-box trait lets us have a greater confidence in future results, and how they actually are produced. Next, in figures 5.6 - 5.9, we examine the corresponding plots of the distributions in the folded-bin space, i.e. the space where the observed data is contained. This is where the likelihood function is originally defined, i.e. being dependent on the variable f_r in eq. 3.10 and eq. 3.9, and we will see the effect of the response matrix on the prior and posterior.

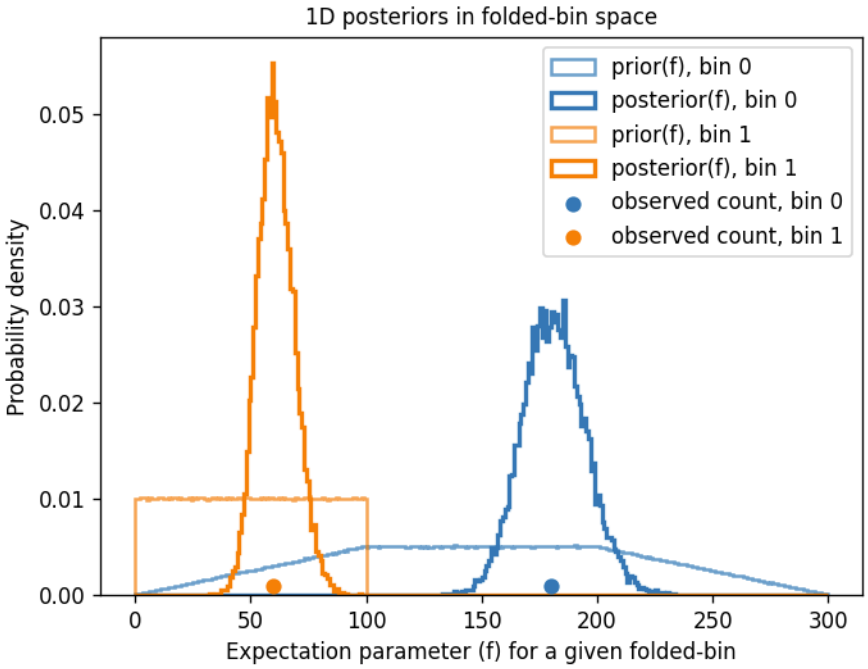


Figure 5.6: Posterior and prior distributions from figure 5.2 folded with the response matrix, along with the observed data, in folded-bin space. We see the smearing effect, how counts can be redistributed due to the detector response. The range for the prior for bin 0 is seen to have been resized from $[0, 200]$ to $[0, 300]$, and to $[0, 100]$ for bin 1. The corresponding heights have thus changed, preserving the total probability of 1 for both priors. The posteriors are centered around their respective observed counts, and show the spread for which possible observed values can lead to the true counts of 120 in this case.

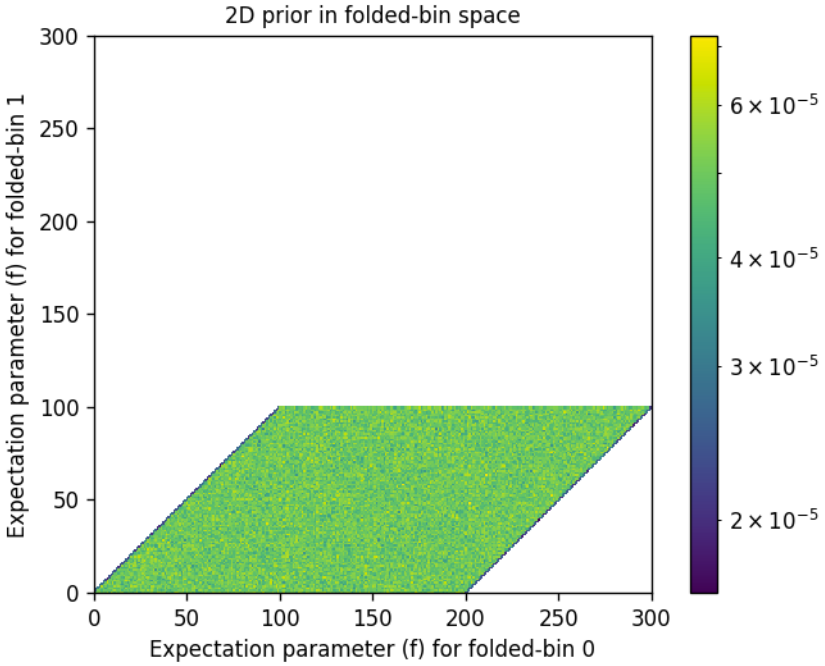


Figure 5.7: The complete 2-dimensional prior distribution from figure 5.3 in folded-bin space. We see that the prior has been skewed and occupies a smaller area than in the truth-bin space. We can also see that the prior heights, shown in color, are larger to compensate.

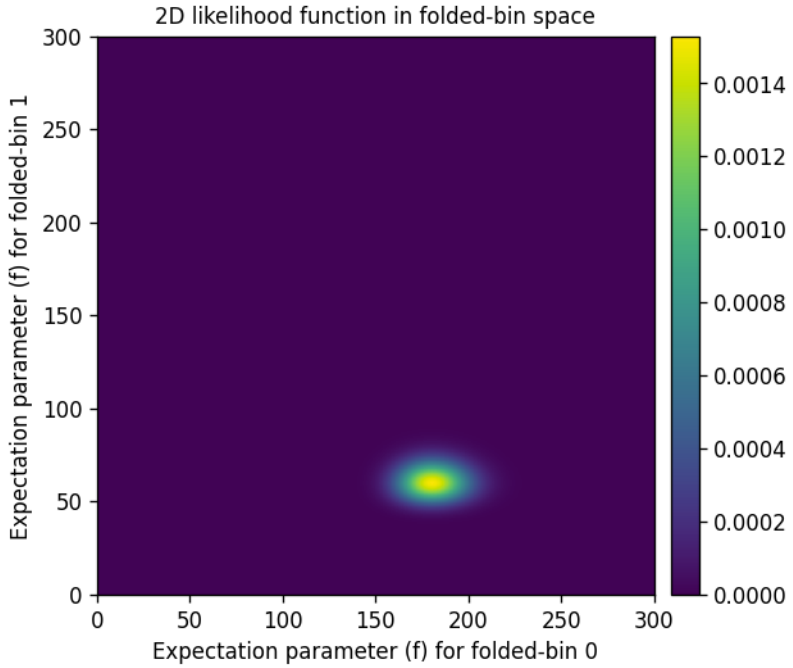


Figure 5.8: The 2-dimensional Poisson distribution from figure 5.4 in the folded-bin space, corresponding to the likelihood in PyFBU. We see a more concentrated peak here than in the truth space.

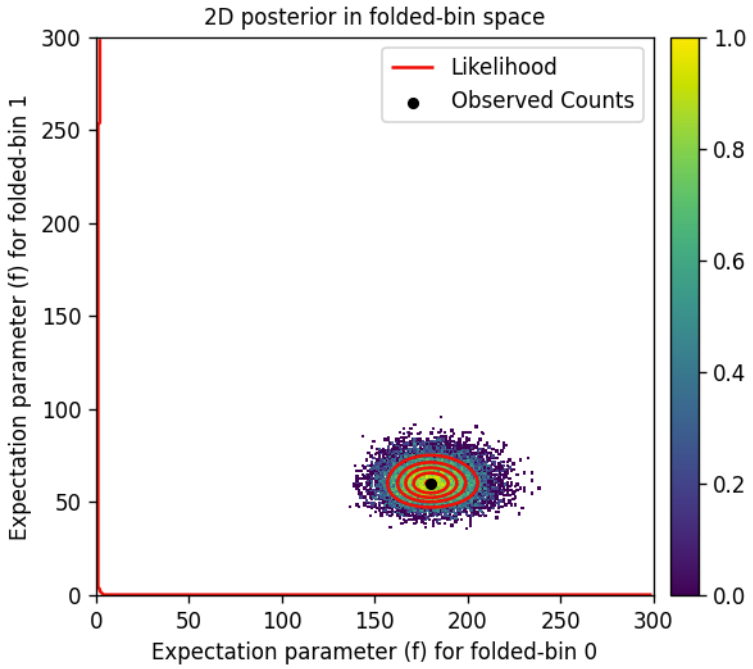


Figure 5.9: The 2-dimensional complete posterior distribution from figure 5.5, as well as contour lines from the likelihood function, in folded space. Here too, the likelihood and posterior shapes and locations match very well, confirming again our assumption about the likelihood. The narrower distribution, especially for bin 1, shows that there are less possible observed counts leading to a truth count of 120, than possible truth counts leading to the observed value from the detector.

Part IV

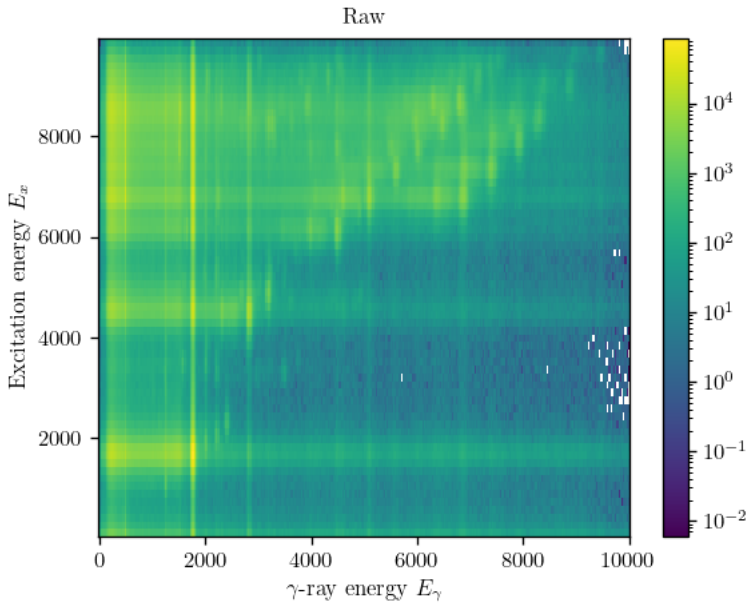
Results & Discussion

CHAPTER 6

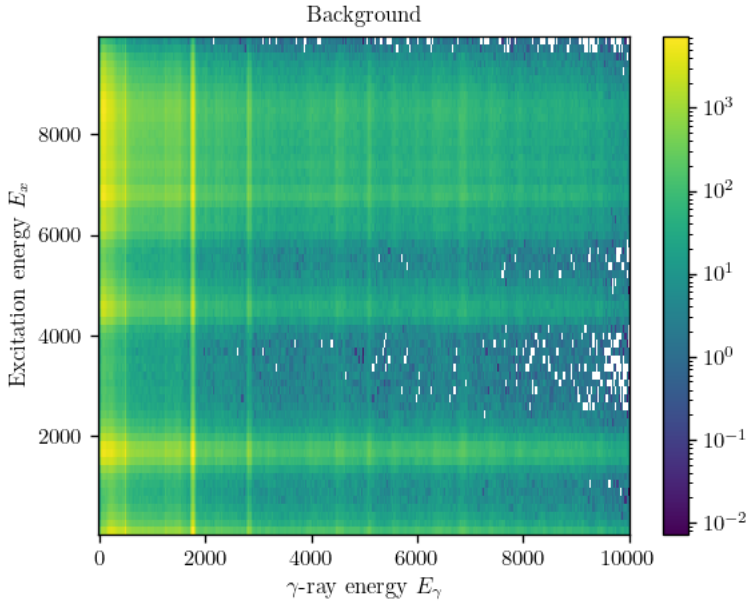
Experimental spectra

6.1 The ^{28}Si spectrum

Here we take a look at the unfolded result using FBU with the new response matrix (2020) from the OMpy library, which should more closely match the original experimental conditions than the response matrix used by Valsdóttir (2017). First, we present the complete raw matrix as well as the background matrix in figure 6.1. These have been rebinned with a factor 3, the same as done by Valsdóttir, to reduce computation time. Both new and old response matrices are then shown in figure 6.2.

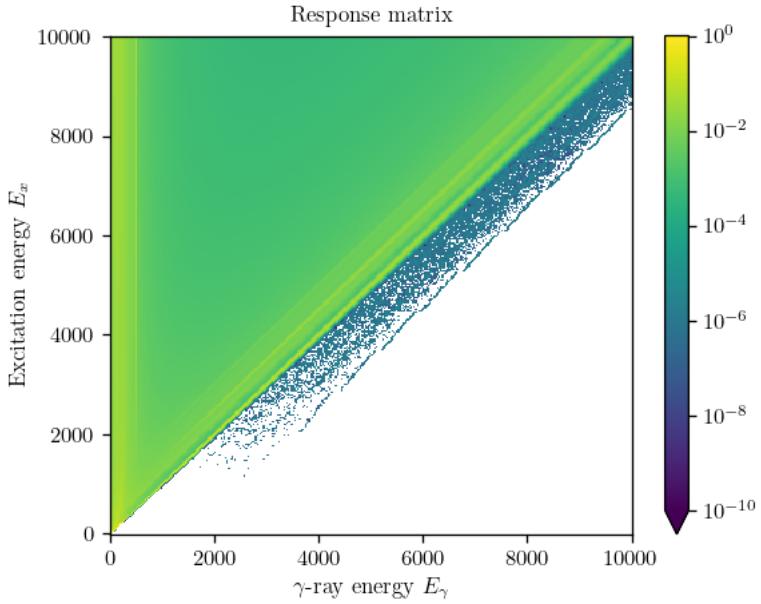


(a)

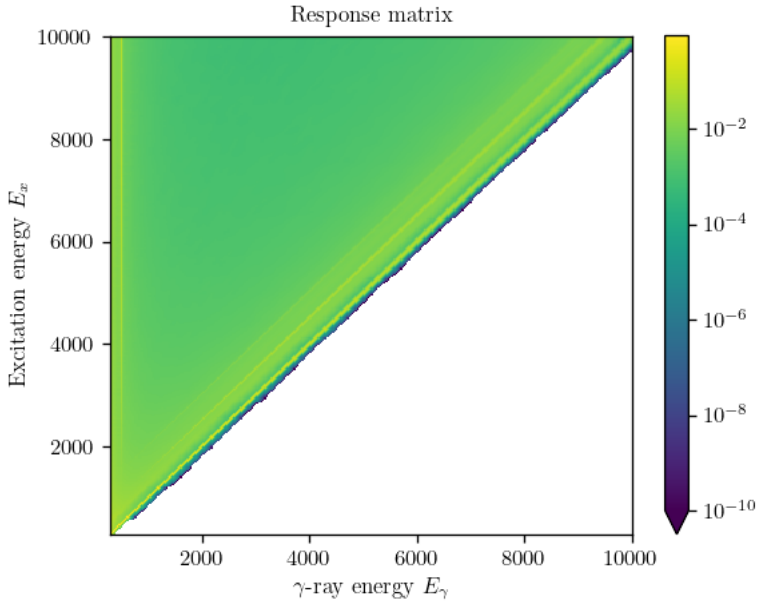


(b)

Figure 6.1: Complete raw spectrum for ^{28}Si, showing the observed γ-ray spectra for each excitation energy E_x, as well as the corresponding background, received from Ann-Cecilie Larsen. These have both been rebinned with a factor 3 on the E_γ axis.



(a)



(b)

Figure 6.2: The new response matrix from 2020 (a), and the old response matrix from 2017, both from the OMpy library [9][10].

6.1.1 The first excited state

The projected spectrum unfolded by Valsdóttir is the first excited state of the ^{28}Si data, i.e the projection of all counts in the interval $E_x = [1400, 2200]$ keV. We use the same projection in order to achieve a direct comparison. This is shown in figure 6.3.

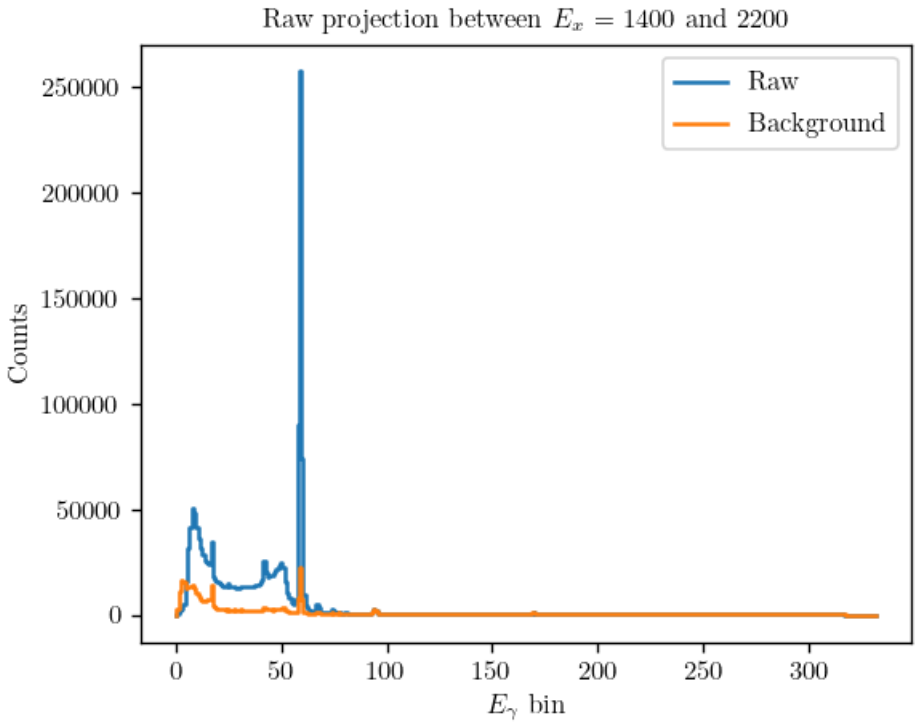
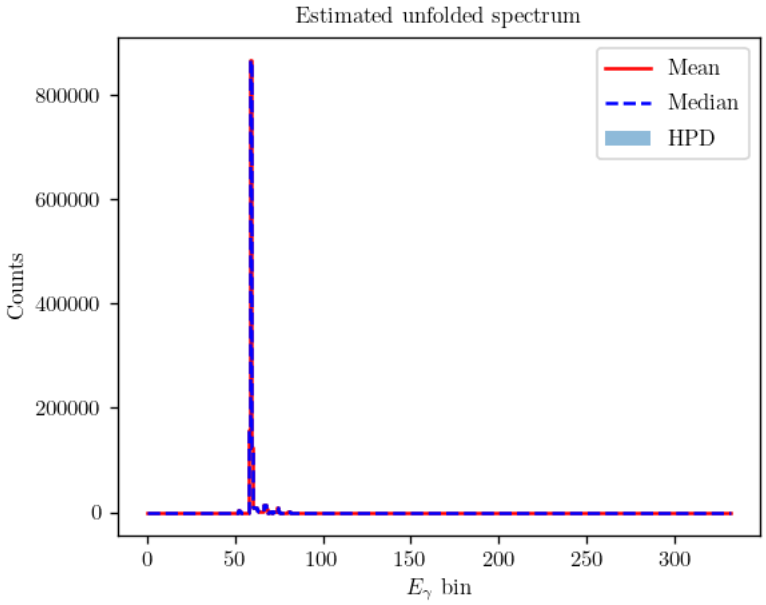
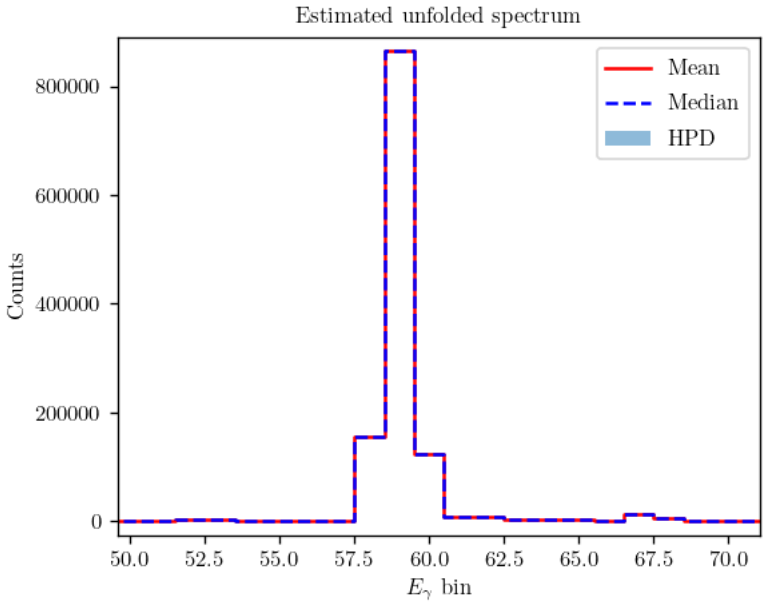


Figure 6.3: Projection of the raw spectrum and background for the first excited state, i.e. between $E_x = 1400$ keV and 2200 keV, see figure 6.1, showing observed counts in each energy bin. Note the background being higher than the raw spectrum at the very start. This is due to increasing the supplied background data by 20% in order to more closely match the actual, unknown background, as discussed by Valsdóttir [4].

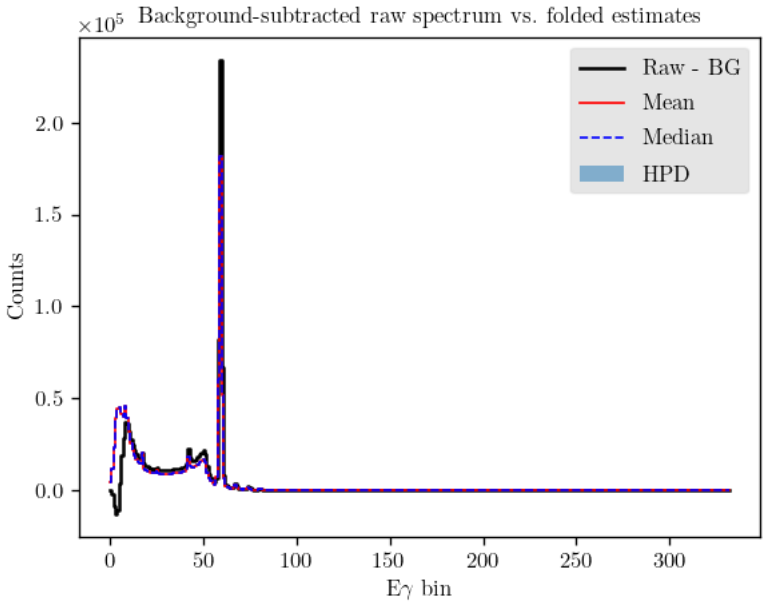


(a)

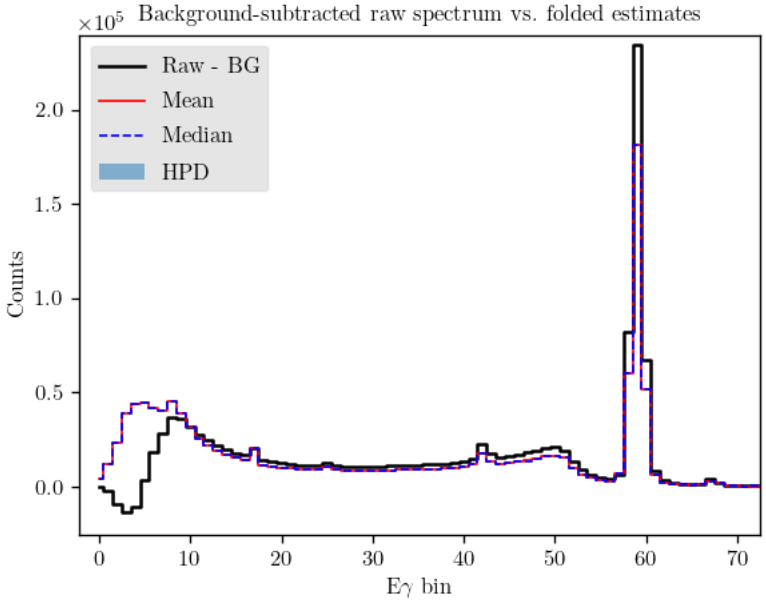


(b)

Figure 6.4: The aggregated estimates for the posterior distributions in each bin, representing an estimated unfolded spectrum, for all bins (a) and zoomed between bin 50 and 70 (b). We see that almost all counts have been redistributed to a single peak around bin 59 for our expectation. This result is then refolded and compared with the raw spectrum (figures 6.5 and 6.6 and table 6.1). Lastly, we compare with Valsdóttir's results in figure 6.7.

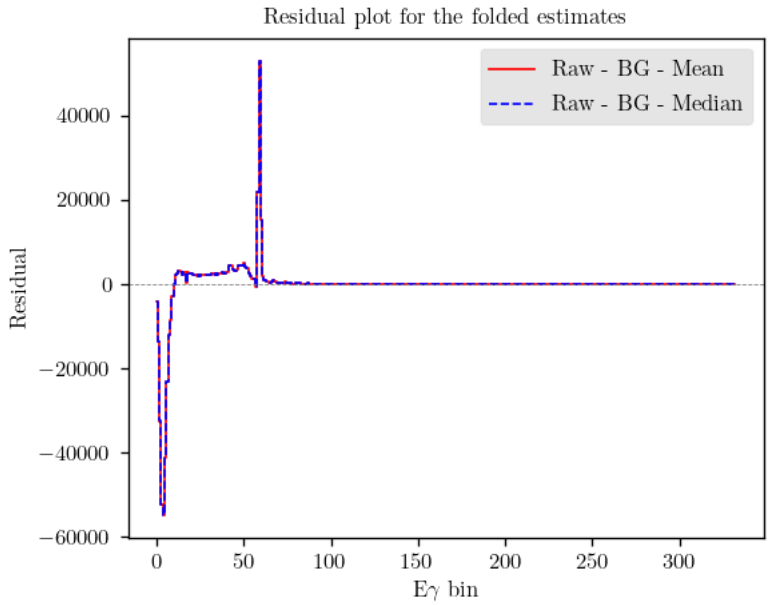


(a)

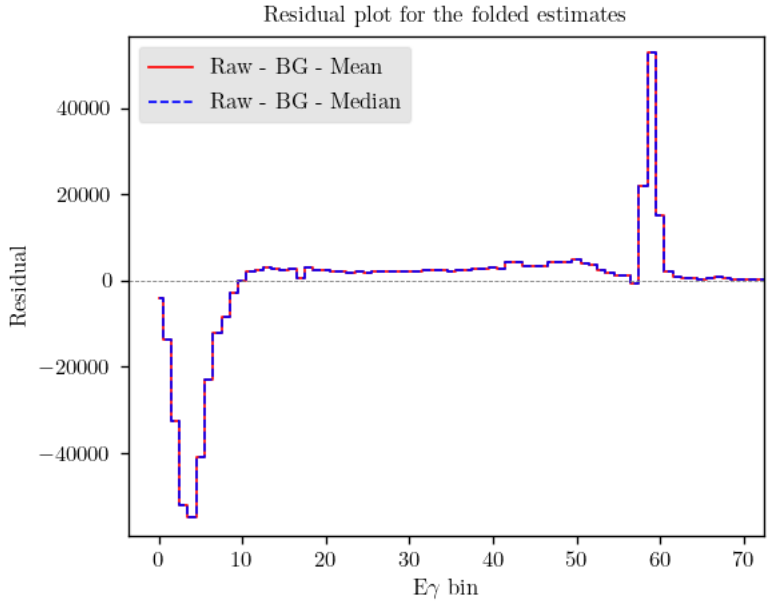


(b)

Figure 6.5: Refolding of the unfolded results compared with the background-subtracted raw data. The first plot (a) shows the result of folding the truth-sample estimates (HPD, point estimates) shown in figure 6.4. The second plot (b) shows the same result for the first 70 bins. The background subtraction is performed to match the presentation of Valsdóttir's results [4]. When unfolding the new spectra (^{146}Nd) below, we instead add the background to the estimates and compare directly with the raw data, see figure 6.16.

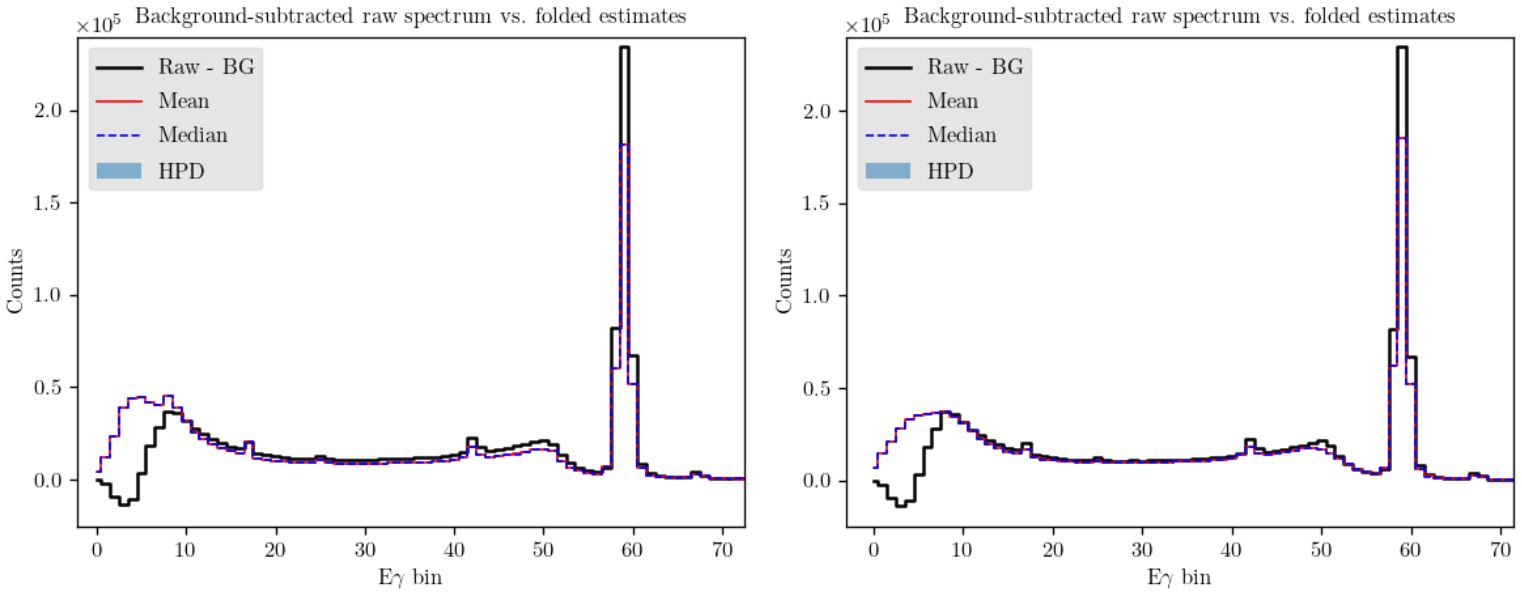


(a)



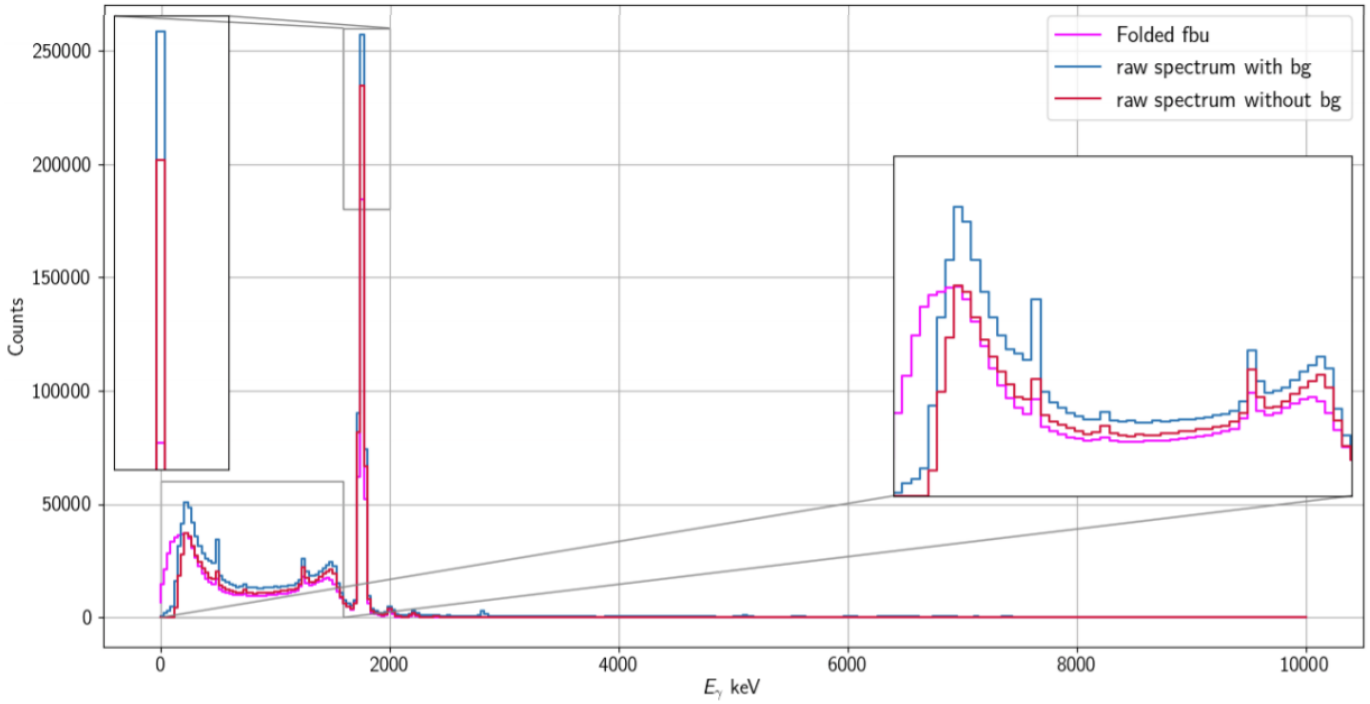
(b)

Figure 6.6: Residual plots showing the difference between the observed raw spectrum and the point estimates from FBU (figure 6.5a), first for all bins (a), then zoomed to the first 70 bins (b).



(a)

(b)



(c)

Figure 6.7: Zoomed version of the refolded result (figure 6.5a) for the first 70 bins (a). We have also included a corresponding result for the old response matrix from 2017 (b). Lastly, Valsdóttir's result with the old response (c) [4]. Surprisingly, the new response leads to a larger discrepancy for the leftmost area, as well as the tall peak being slightly lower. This may seem to imply that the new response is a worse match to the experimental conditions than the old, when the opposite should be true. The error metrics shown in table 6.1 tell a similar story.

Table 6.1: Error metrics for refolded results (rounded) in figure 6.7. We see again that the old response matrix appears to lead to a better accuracy than the new. As mentioned in figure 6.7, the low-energy area shows the most apparent discrepancy. Both response matrices are known to contain errors for the low-energy bins, and we will investigate what happens when we cut the first 10 bins from the data before unfolding in the next subsection.

	Median		**Mean**	
	New response	Old response	New response	Old response
MAE	1428	1130	1427	1128
R^2**-score**	0.864	0.902	0.864	0.902

6.1.2 The first excited state - Cutting the first 10 bins

Due to the apparent reduction in accuracy from switching to the new response matrix, which should be a better fit to the raw data, we now perform FBU on the data with the 10 first bins cut. This area is as mentioned associated with errors for the response matrices, errors which seem to be larger in the new response. Removing this area should reduce the impact of the erroneous response values on the unfolded spectrum, leading to a better result. The projection of the first excited state with the cut bins is shown in figure 6.8. Now we do the same unfolding process as previously, this time using the cut raw data. The unfolded spectrum is shown in figure 6.9.

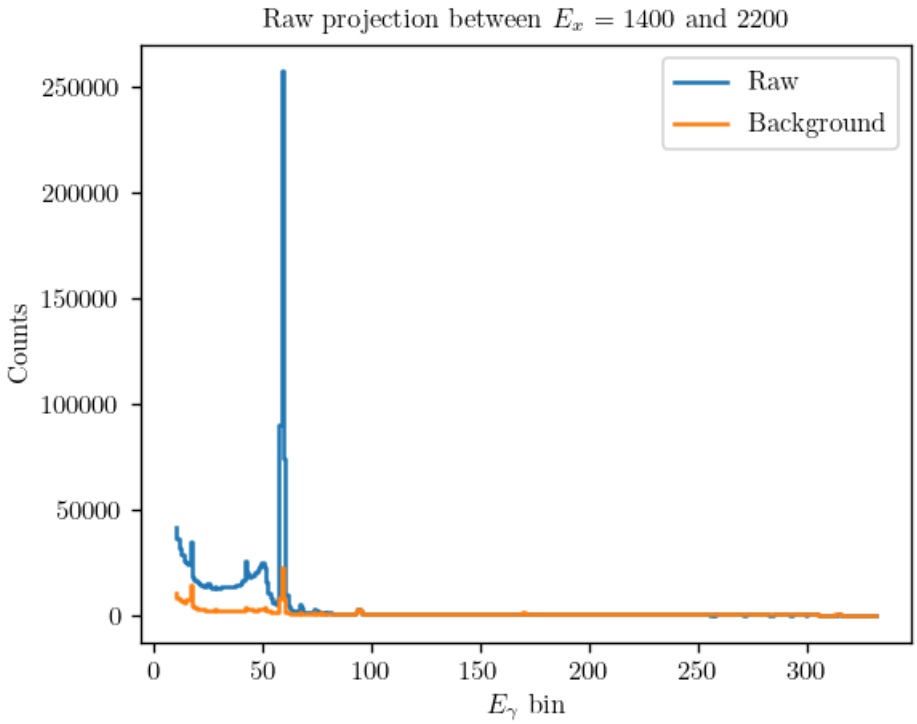
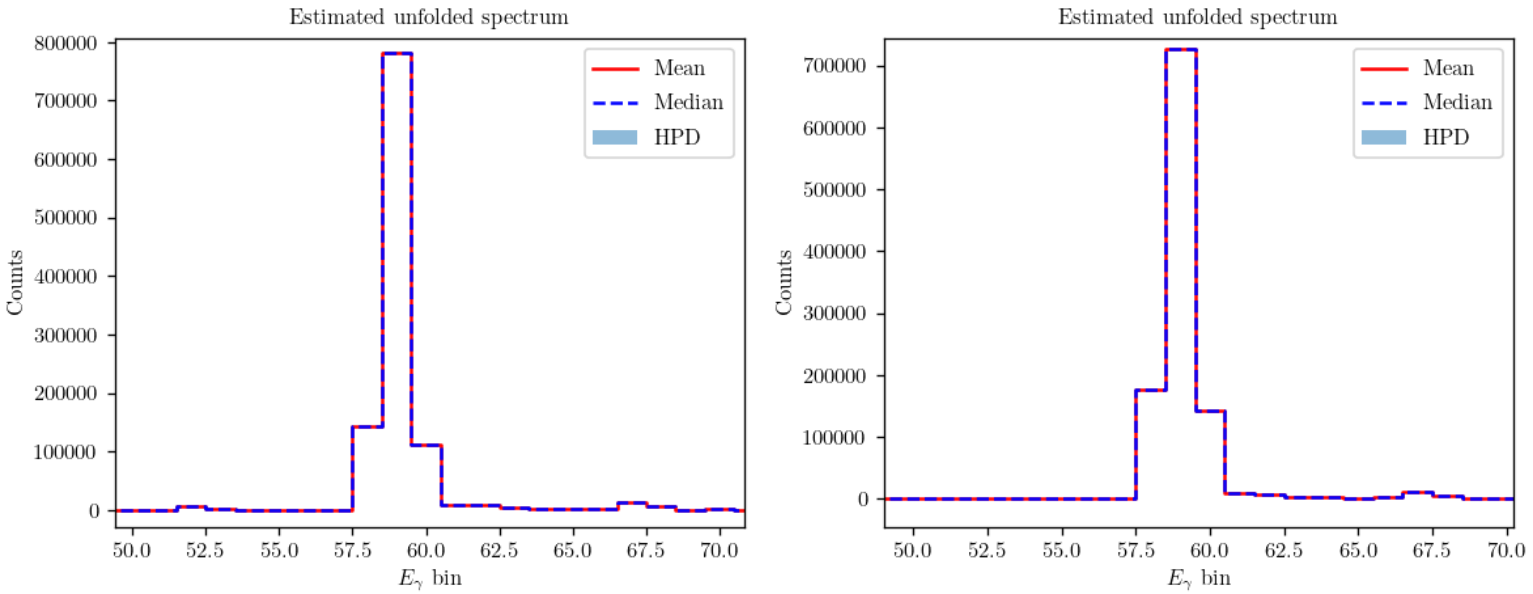
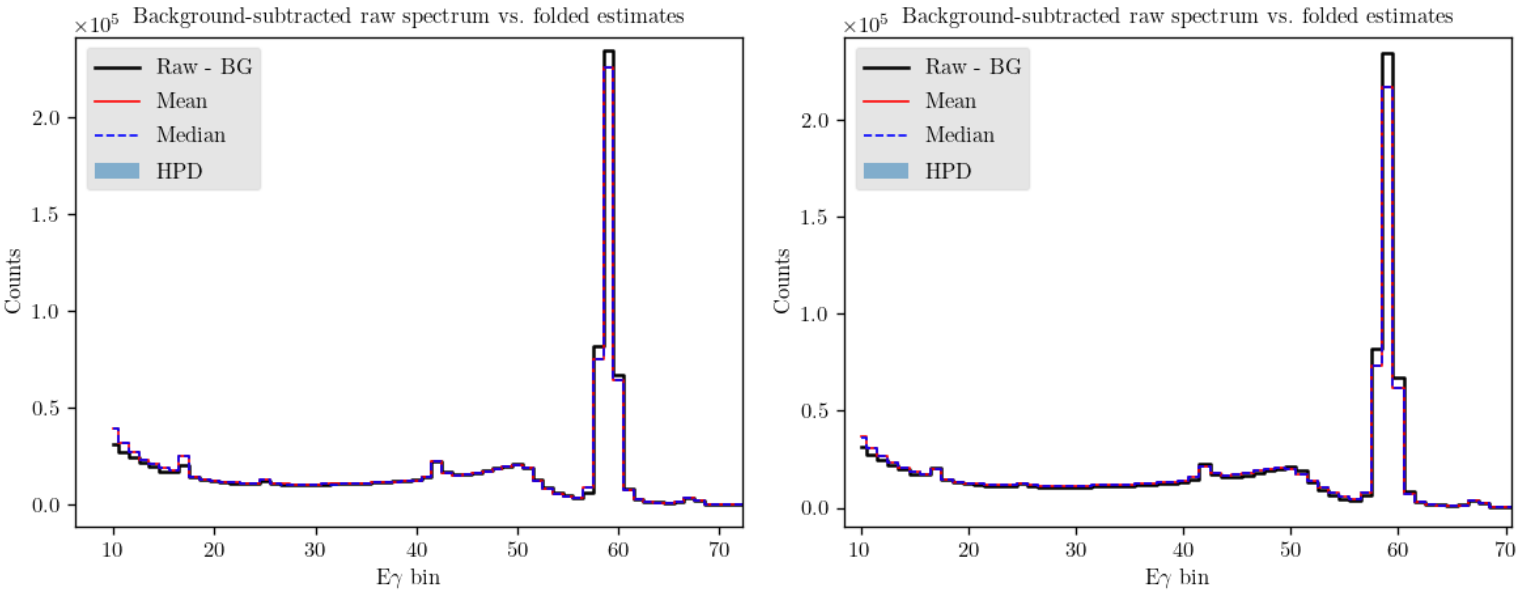


Figure 6.8: The projected raw spectrum for the first excited state of ^{28}Si shown in figure 6.3, with the 10 first bins removed. This is done to avoid the negative impact of response matrix errors for the low-energy area.



(a) (b)

Figure 6.9: The unfolded spectrum for the first excited state of ^{28}Si, with the first 10 bins cut, in the area between bin 50 and 70. Figure (a) shows the result using the new response, while figure (b) shows the corresponding result for the old response. The peaks are very similar to their uncut versions, but lower than in figure 6.4, due to the cut leaving fewer available counts to redistribute. Even though these structures both look similar, the peak in (a) is taller by roughly 58 000 counts (approximately 782 000 (a) vs 725 000 counts (b)). The new response has thus contributed to a greater redistribution of counts into the peak. These results are then refolded with their corresponding response matrices and compared with the raw spectrum in figure 6.10



(a) (b)

Figure 6.10: Refolding of the unfolded results from figure 6.9 compared with the background-subtracted raw data. The first plot (a) shows the result of refolding the results using the new response matrix. The second plot (b) shows the corresponding results for the old response. We observe a significant improvement for both plots compared to figure 6.5, with the peak and values in other bins being more closely approximated. We see that, especially for the peak, the new response has allowed FBU to get closer to the raw spectrum by roughly 8000 counts (approx. 225 000 vs 217 000, the correct raw value being approx. 235 000). The corresponding error metrics are shown in table 6.2.

Table 6.2: Error metrics for refolded results (rounded) in figure 6.10. We see that the scores have significantly improved from table 6.1 by cutting the first bins of the raw data. The biggest improvement is seen in the results using the new response matrix, further implying that the low-energy area errors in the new response are larger than in the old. We see that the new response now in fact scores better for both metrics, signifying that it, as mentioned, is a better match for the ^{28}Si data (at least for all bins above bin 10). Even though we now observe a consistent improvement from using the new response, there may yet be other factors, like the now removed low-energy bins, limiting further improvement. One such factor may be the rebinning of the raw data and background, mentioned in figure 6.1. E.g. if the new response has improvements that only come into play at higher resolutions, rebinning may reduce the impact.

	Median		**Mean**	
	New response	Old response	New response	Old response
MAE	210	272	209	272
R^2**-score**	0.997	0.995	0.997	0.995

6.2 The ^{146}Nd spectrum

Now we perform FBU on a brand new dataset from the ^{146}Nd(p,p'γ) reaction and compare the results with the folding iteration method in OMpy. This time, we use the old response matrix from 2017, see figure 6.13, as it should most closely match the experimental conditions for ^{146}Nd. The raw data and background is shown in figures 6.11 and 6.12. No rebinning has been performed this time, as computation time was deemed reasonable (usually no more than 2 hours).

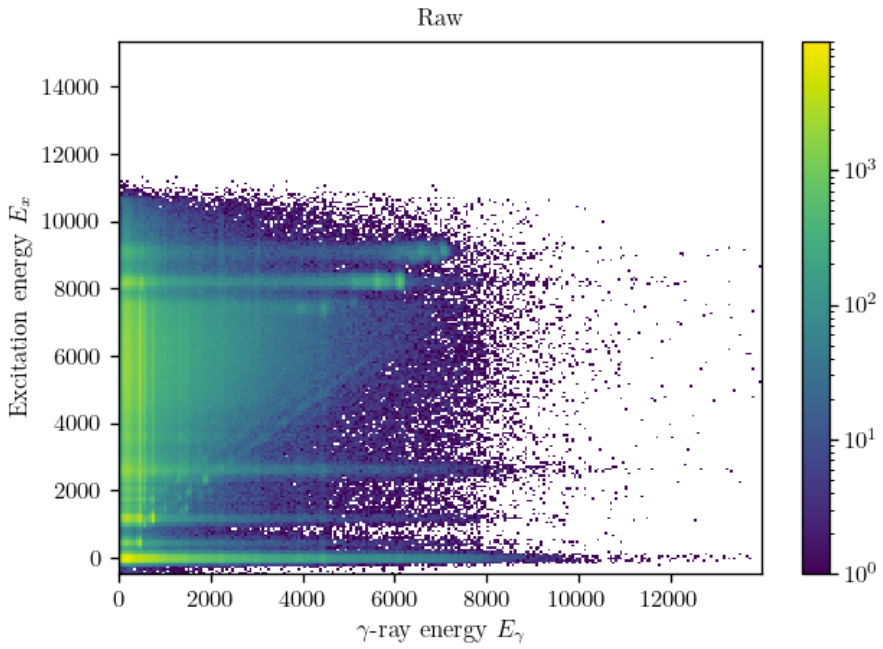


Figure 6.11: Complete raw spectrum for ^{146}Nd, showing the observed γ-ray spectra for each excitation energy E_x, received from Ann-Cecilie Larsen.

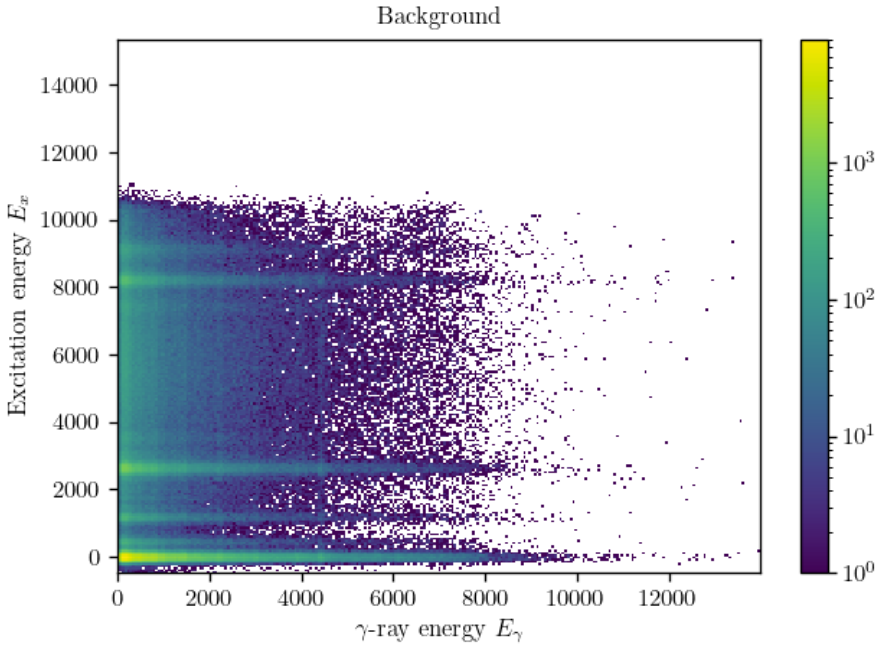


Figure 6.12: The background present in the raw data in figure 6.11, received from Ann-Cecilie Larsen.

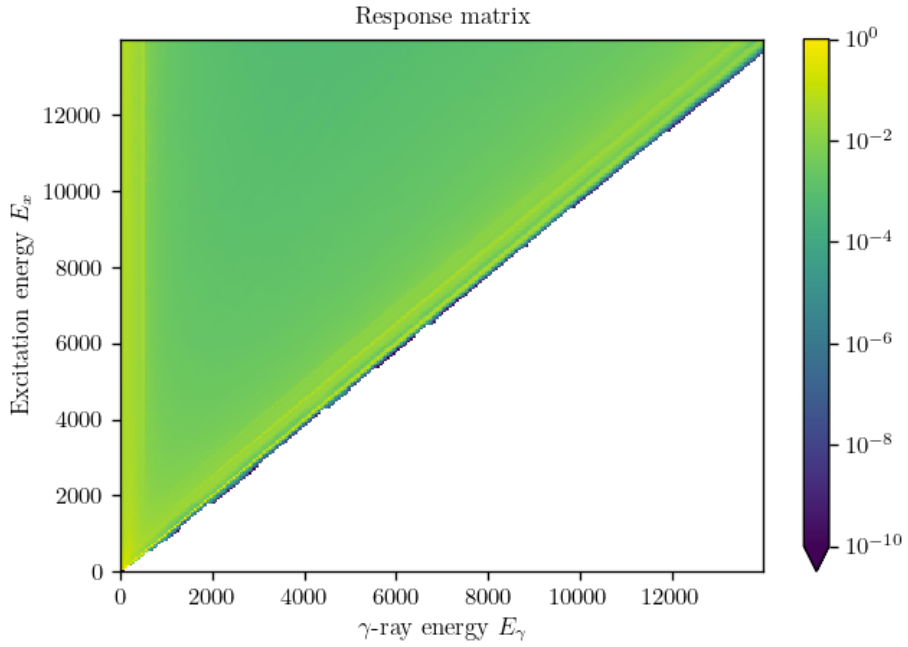


Figure 6.13: The complete 2017 response matrix from the OMpy library [9][10].

6.2.1 The first excited state

We begin with the first excited state of ^{146}Nd, projecting all counts between $E_x = 300$ keV and 600 keV. We do know that the response matrix contains some errors for the first few E_γ-bins, but we start with unfolding the complete uncut projection before comparing with a version with the first 3 bins cut. The raw projection is shown in figure 6.14, and the following unfolded spectrum is shown in figure 6.15.

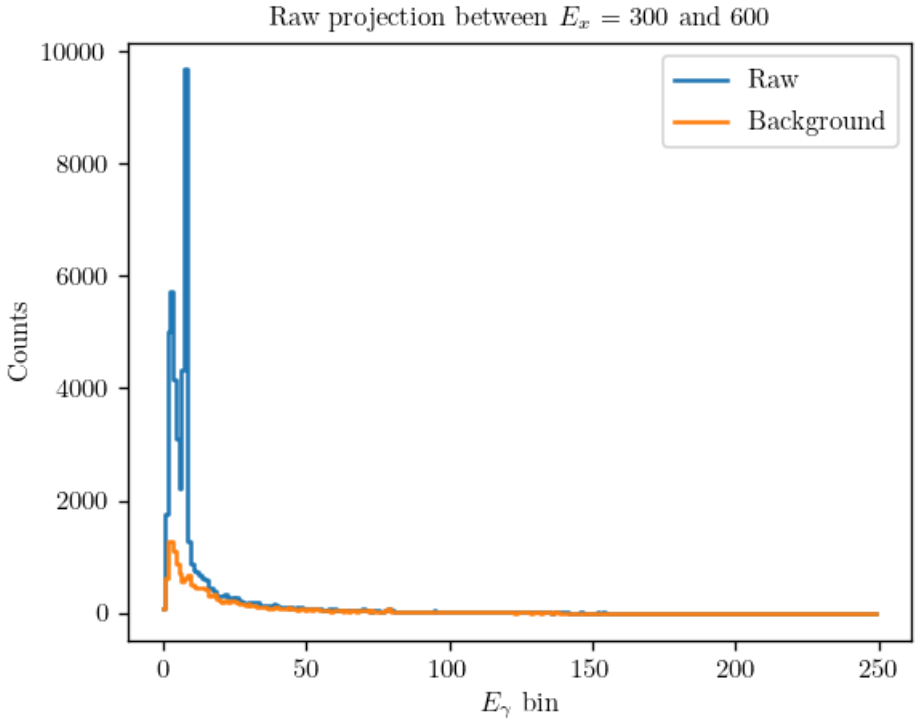
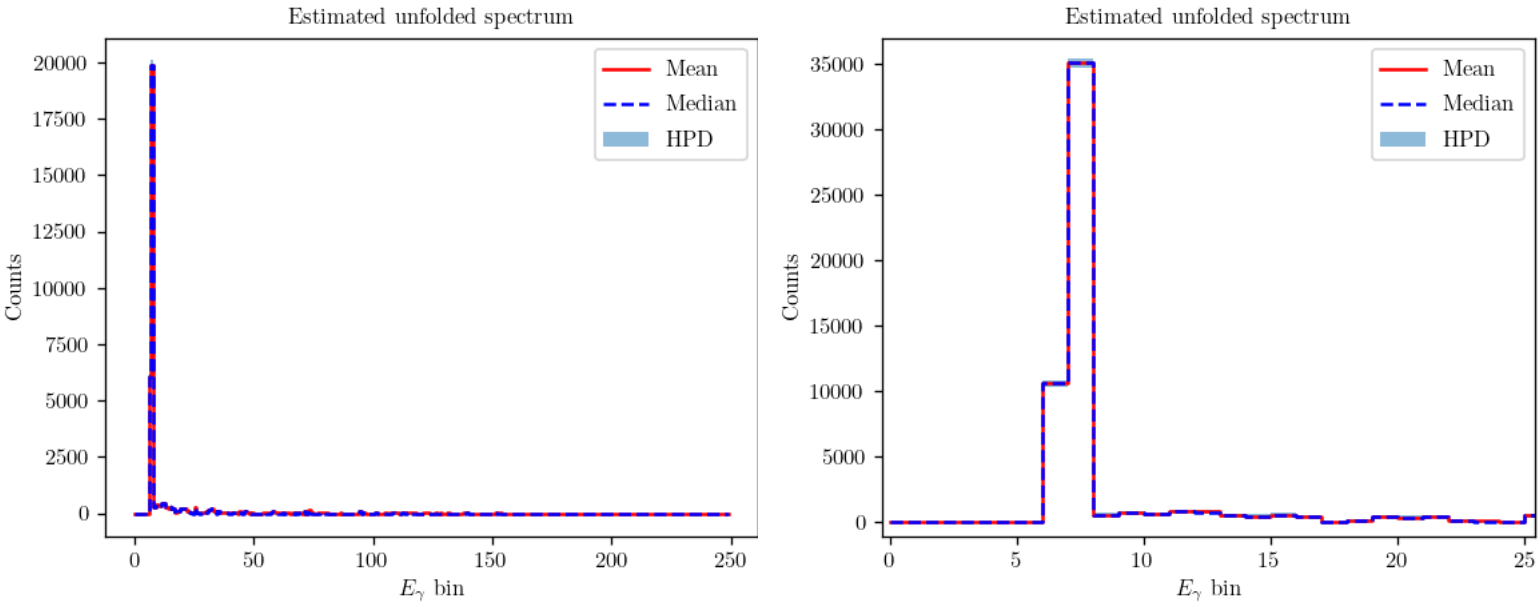


Figure 6.14: Projection of the raw spectrum and background for the first excited state, i.e. between $E_x = 300$ keV and 600 keV, see figure 6.11, showing observed counts in each energy bin.



(a) (b)

Figure 6.15: The aggregated estimates for the posterior distributions in each bin, representing an estimated unfolded spectrum, for all bins (a) and zoomed to the first 25 bins (b). We observe a single peak by the low energy bins, showing that the double peak in the observed spectrum, figure 6.14, has almost been completely combined in our truth estimate. Further analysis is performed on this spectrum and shown in figures 6.16 and 6.17. The results from OMpy and subsequent comparisons are shown in figures 6.18, 6.19 and in table 6.3.

As seen in figure 6.16, we have presented two different ways to view the refolded results. The first is done by folding the truth-sample estimates in figure 6.15, and the second is done by calculating new estimates on the folded truth-samples. Mathematically: mean($\boldsymbol{T}_{\text{sampl.}}$) $\times$ $\boldsymbol{R}$ vs. mean($\boldsymbol{T}_{\text{sampl.}} \times \boldsymbol{R}$). The difference between the two is subtle, and the right choice might not be clear. In the first case (6.16a), we have a direct relation between the estimates in truth-space (figure 6.15) and folded space through the response. Folding the estimates might be deceiving however, as they are not direct samples of the truth-posterior, meaning they may in some cases take a value that is not a probable event (i.e. the mean of [0,0,0,10,10,10] being 5, even though a value of 5 has not happened at all). In the second case (6.16b), we instead perform the refolding on the truth-samples themselves, as they are our actual representation of possible true data, thereby emulating a real experiment. New estimates are then calculated on these folded truth-samples. We take the second case to be the right choice, but will present both in the following figures, for comparison. Another thing to note is that we now compare the observed data with

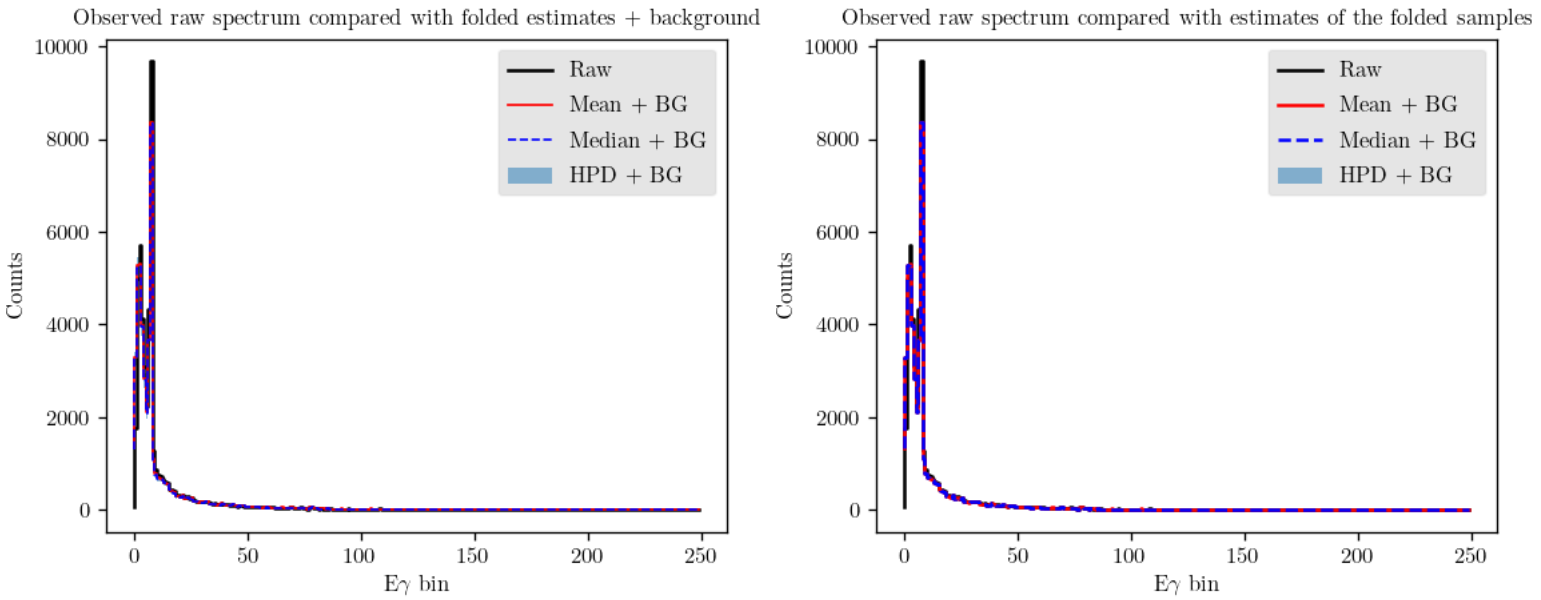
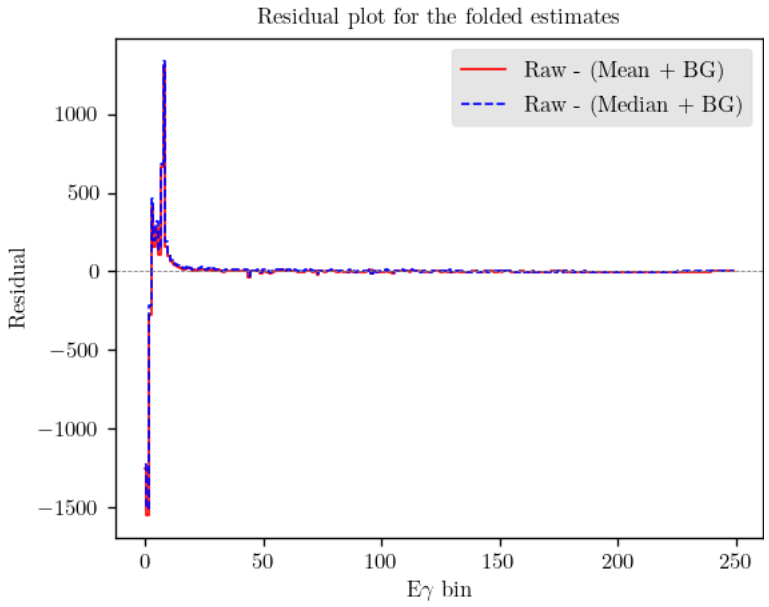
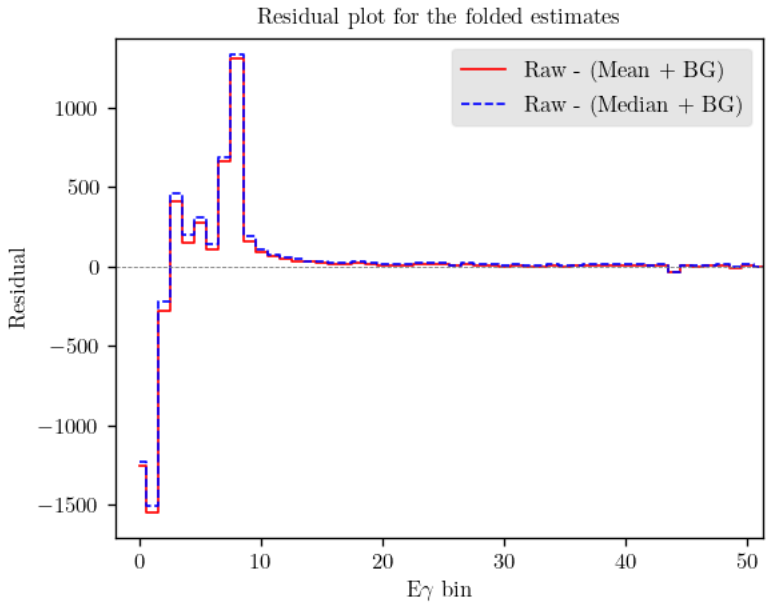


Figure 6.16: Refolding of the unfolded results compared with the raw observed data, in two ways. The first plot (a) shows the result of folding the truth-sample *estimates* (HPD, point estimates) shown in figure 6.15. The second plot (b) has been made by directly folding the truth-samples, and afterwards calculating a new HPD interval and point estimates based on the folded *samples*. The corresponding residual plots are shown in figure 6.17.

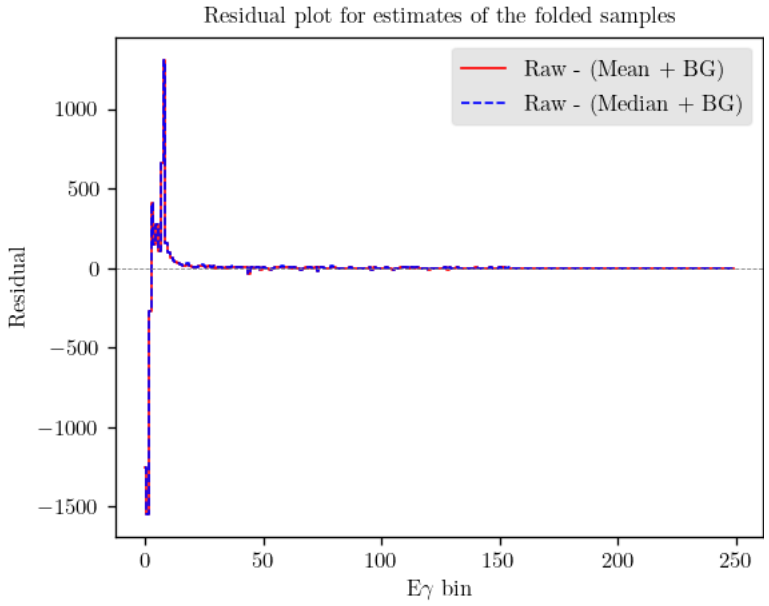
the refolded spectrum with the background *added* (raw vs. refolded + background), as opposed to in figure 6.5. There, the comparison is between a background-*subtracted* observed data and the calculated estimates, essentially moving the background to the other side of the 'equation' (raw - background vs. refolded). Its a matter of preference, but the former avoids falsely showing any negative counts from a subtraction, and is what will be presented in the following figures.



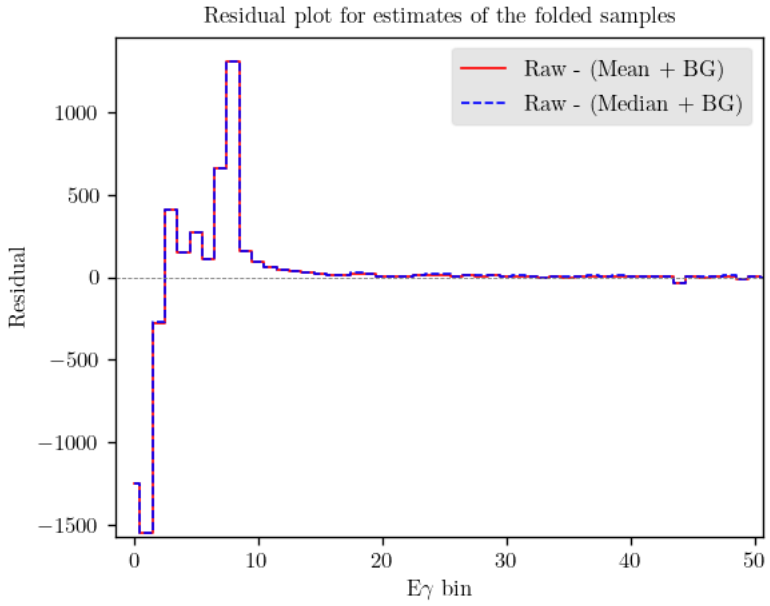
(a)



(b)

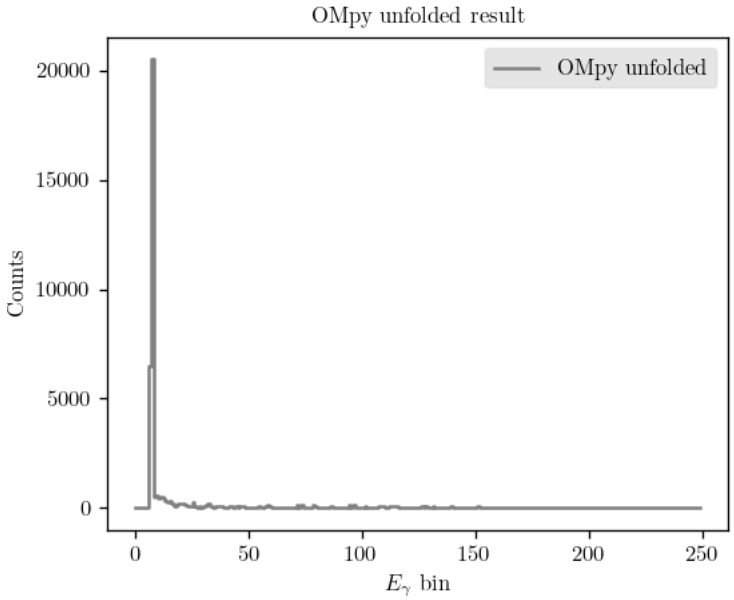


(c)

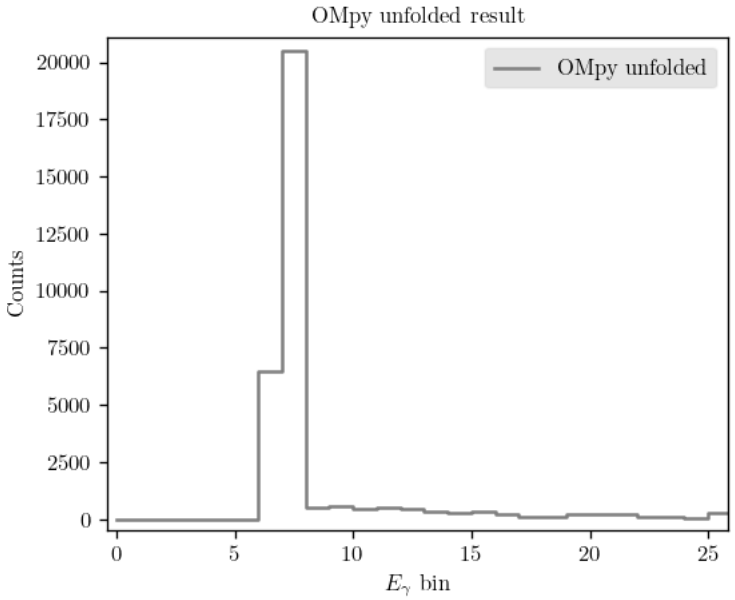


(d)

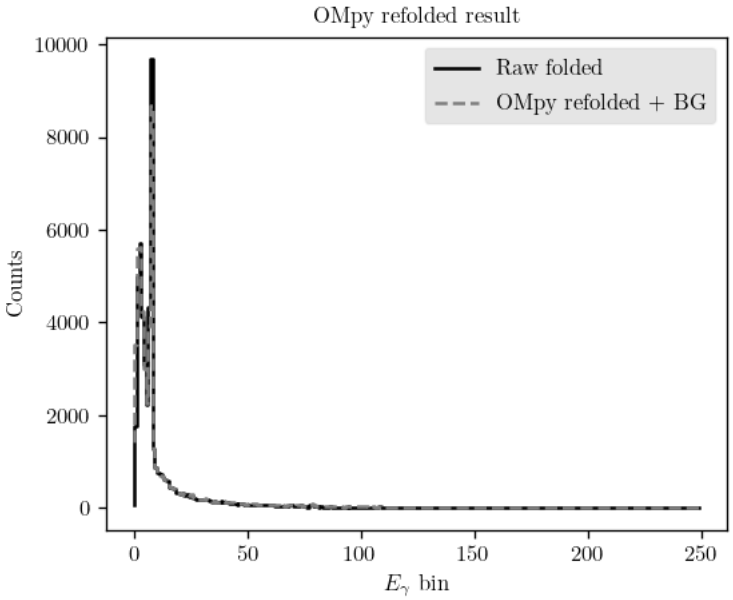
Figure 6.17: Residual plots showing the difference between the observed raw spectrum and the point estimates from FBU, first for all bins, then zoomed to the first 50 bins. Plots (a) and (b) pertain to the folded truth-sample estimates from figure 6.16a. Plots (c) and (d) are connected to the folded-sample estimates from figure 6.16b. These are very similar, but some differences appear in figure 6.19 and table 6.3.



(a)



(b)



(c)

Figure 6.18: Result from unfolding using OMpy for all bins (a) and zoomed to the first 25 bins (b), as well as the subsequent refolded result compared with the observed spectrum (c). These results look similar to those from FBU, and a closer look will be shown in figure 6.19.

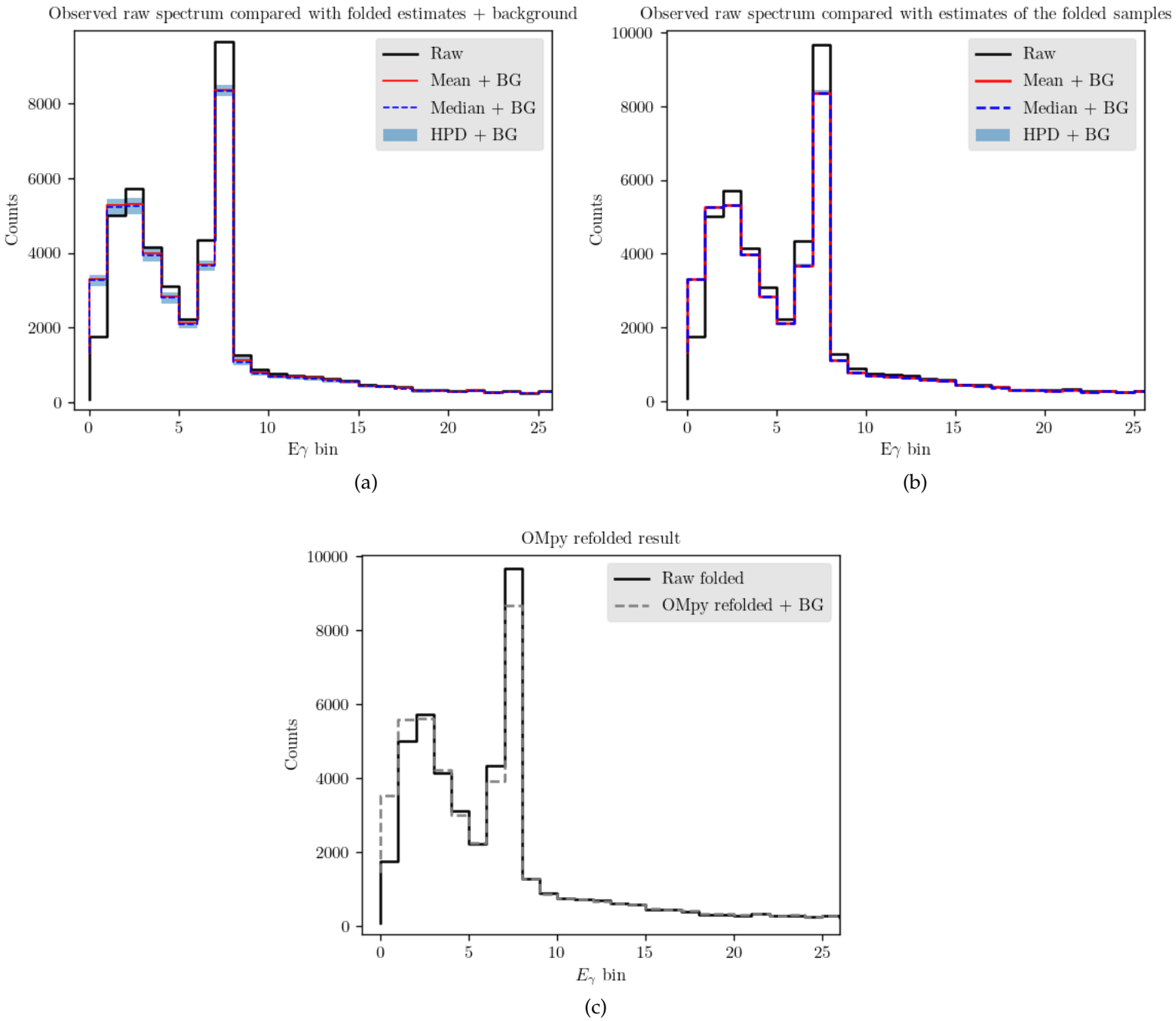


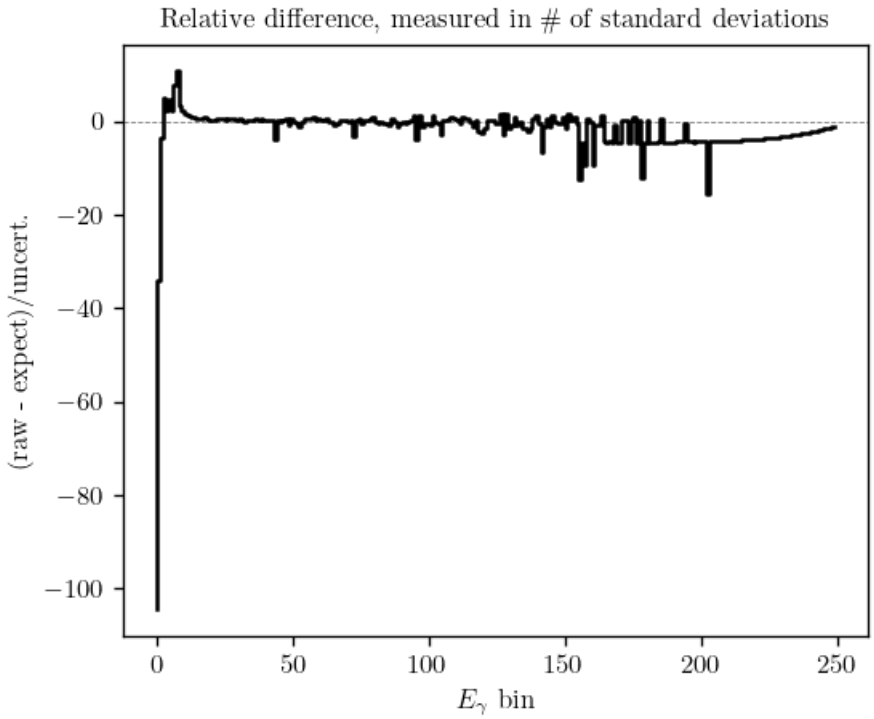
Figure 6.19: Zoomed versions of the refolded results (figures 6.16a 6.16b and 6.18c) comparing FBU with OMpy for the first 25 bins. First, we see one effect of folding the truth-samples estimates (a), the HPD interval has become larger than the interval calculated on the folded samples (b). Overall, all three plots are visually quite similar, with small variations per bin. We see a very similar discrepancy between refolded results and observed data for all three plots at the lower energy bins. This is due to the previously mentioned low-energy errors in the response matrix. With the correct information compensating for these errors, we would be able to adjust the results and observe an even closer match. Alternatively, the first few bins can be cut from the raw spectrum before unfolding, this will be done in subsection 6.2.2. The error metrics for these results are shown in table 6.3.

Table 6.3: Error metrics for refolded results (rounded). We see all estimates score very well with low mean absolute errors, and R^2-scores all above 0.95. As seen in figure 6.19, the largest discrepancies are located at the lowest energy bins, however these are not large enough to have a big impact on the overall scores, since all bins are weighted equally in the metrics. There is an argument to be made about weighting these bins more, as this is where the most interesting structures are located. Doing so would require a manual definition of what is deemed the most interesting area on a case-by-case basis. Since we either way have the residual plots to directly show local errors, we are content to leave the MAE and R^2-score as summarizing error metrics for the entire spectrum. These scores would be even better if we were to include the mentioned additional information about the errors in the response matrix for low energies, see figure 6.19, or if we cut the first few bins of the raw spectrum, which will be done in subsection 6.2.2. The results from both FBU and OMpy are very close, with a slightly higher R^2-score for the FBU estimates, signifying a very slightly overall better match with the observed data. Furthermore, we see a very small, almost negligible improvement in the FBU results when using the estimates calculated on the folded *samples* (figure 6.16b, here abbreviated to Fold. sampl. and what we deemed to be the right choice), over the folded truth-sample *estimates* (6.16a, Fold. est.).

	FBU Median		**FBU Mean**		**OMpy**
	Fold. est.	Fold. sampl.	Fold. est.	Fold. sampl.	
MAE	30.8	28.9	30.8	29.2	26.0
R^2-score	0.968	0.968	0.968	0.968	0.955

Table 6.4: The mean variance for both the truth-samples output from FBU, and the same samples folded with the response. By folding the samples, the subsequent variance is significantly smaller.

	True samples	**Folded samples**
Mean variance	966.9	92.4



(a)

Figure 6.20: Relative uncertainty plot, showing in the number of standard deviations, the level of relative uncertainty for each bin in the spectrum. We see that the very first bins have a significant discrepancy, pointing to systematic errors in the model. This is likely due to the mentioned response matrix errors.

6.2.2 The first excited state - Cutting the first 3 bins

Now we cut the first 3 bins of the spectrum in order to mitigate the discrepancies from the response matrix errors. As opposed to the first excited state of ^{28}Si, the peak is much closer to the first bins, meaning we cannot cut as many without removing the peak entirely. The largest discrepancies are seen in the first 3 bins, so such a cut should have a positive impact. The cut projected raw spectrum and background is shown in figure 6.21. Then we do the same unfolding process as previously, this time using the cut raw data. The unfolded spectrum is shown in figure 6.22.

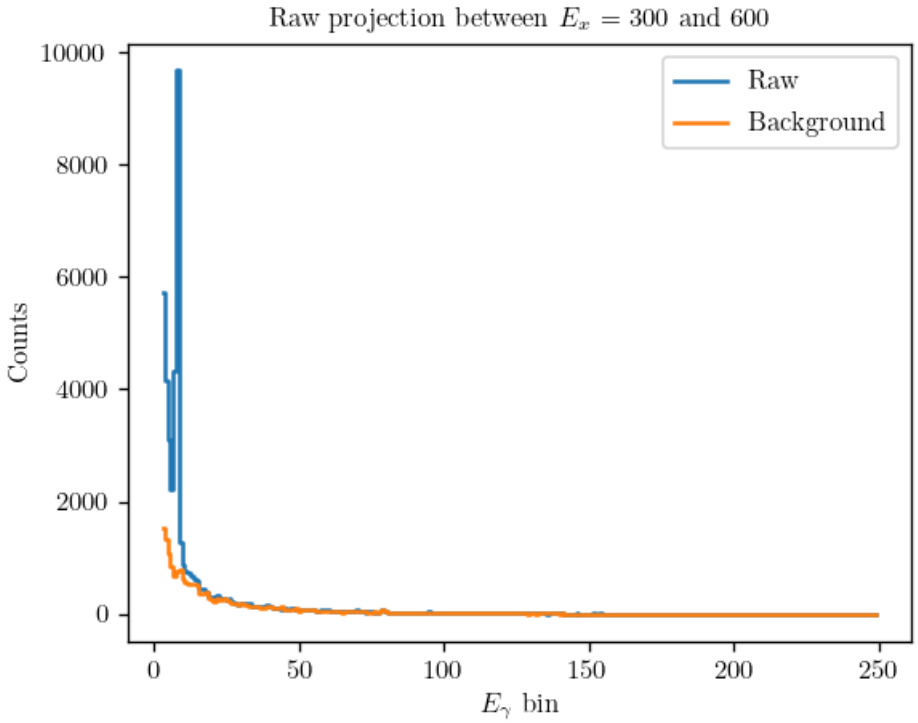
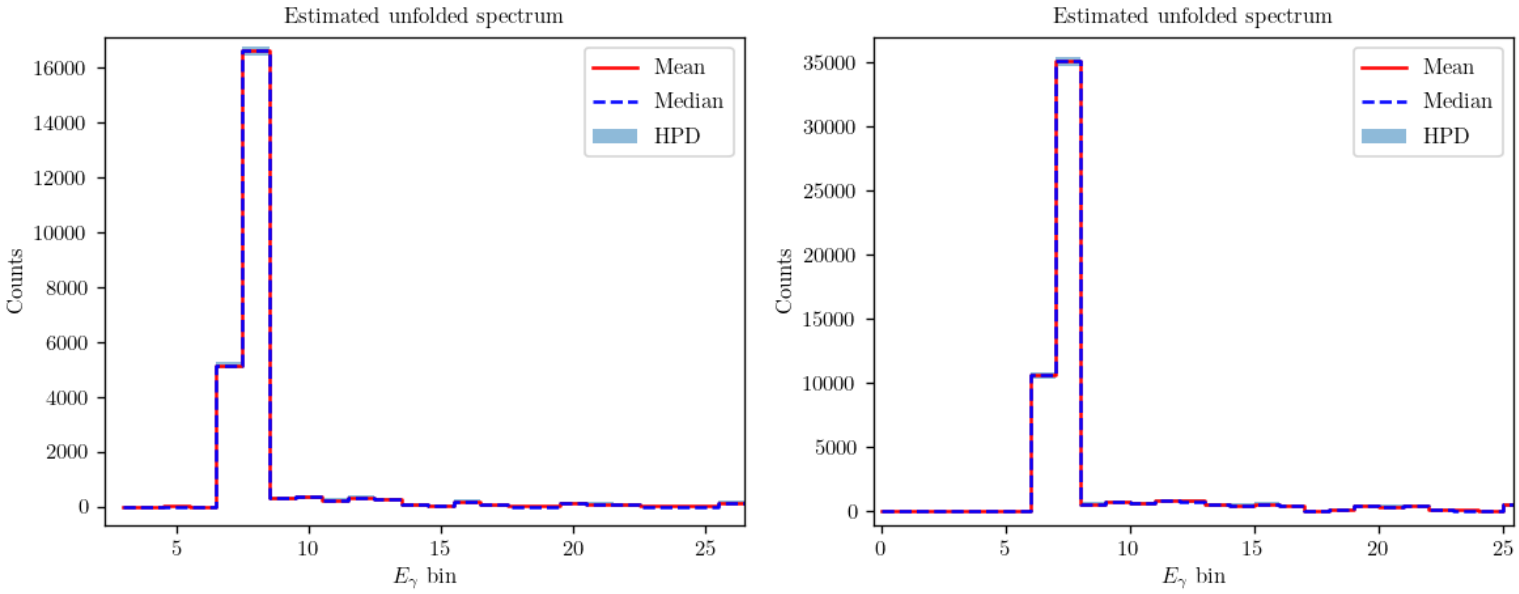


Figure 6.21: The projected raw spectrum for the first energy state shown in figure 6.14, with the 3 first bins removed. This is done to avoid the negative impact of response matrix errors for the low-energy area.



(a) (b)

Figure 6.22: The unfolded spectrum for the first excited state of ^{146}Nd zoomed to the first 25 bins. The left plot (a) shows the result using the cut raw spectrum, while the right (b) shows the result using the uncut spectrum, see figure 6.15. The peaks are very similar, but lower in (a), due to the cut leaving fewer available counts to redistribute.

The following refolded result in figure 6.23 shows only the estimates calculated on the folded truth-samples in figure 6.16b. We also focus on the zoomed plots to compare between the results for the cut and uncut spectra.

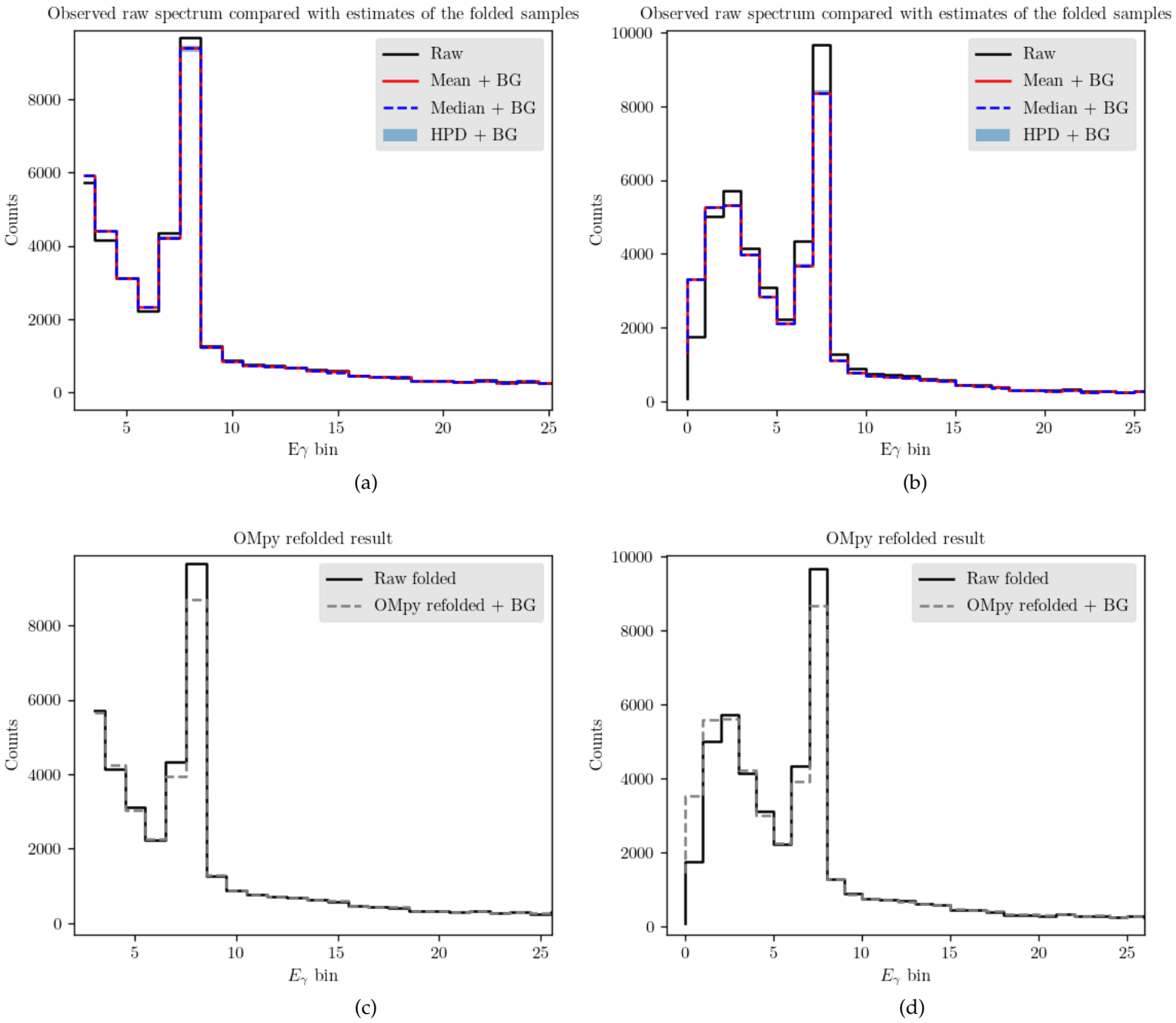


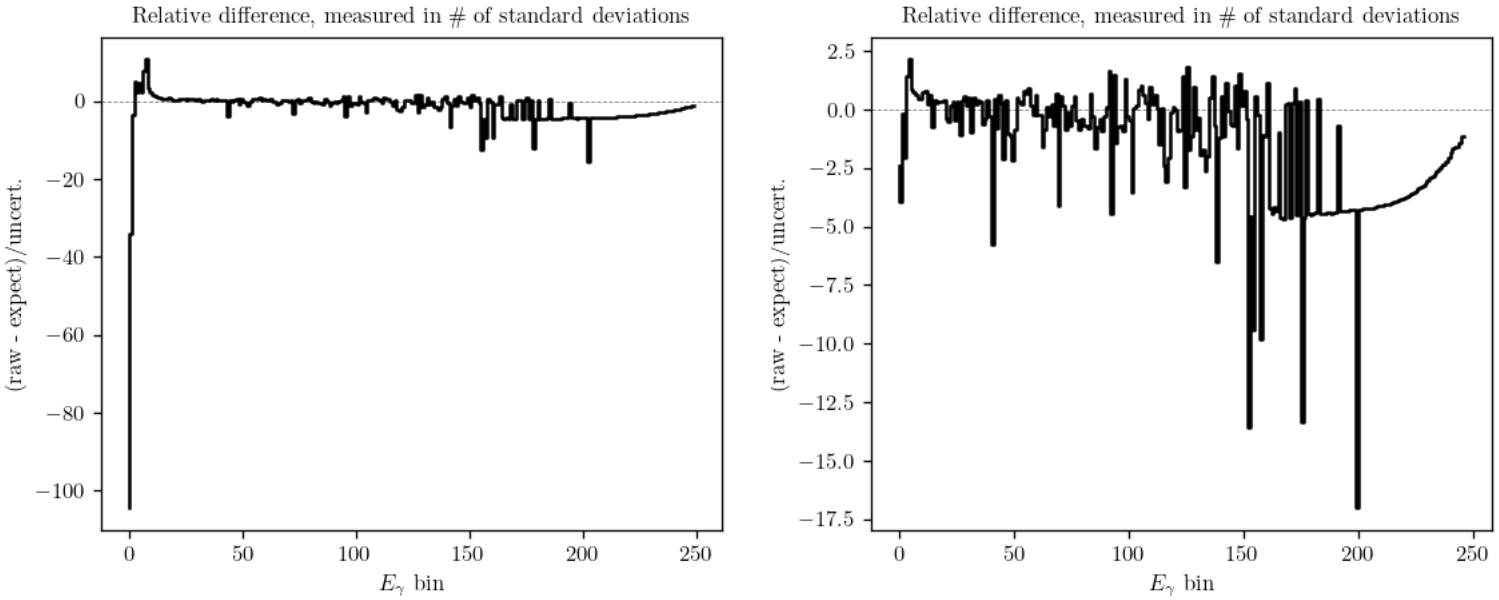
Figure 6.23: Zoomed versions of the refolded results comparing FBU with OMpy for the first 25 bins. The top plots show the FBU results using the cut (a) and uncut (b) raw spectra, respectively. The bottom plots, (c) and (d), show the corresponding OMpy results. We see that cutting the raw spectrum has a very positive effect on the unfolding performance, with a much closer match between raw and refolded. Furthermore, FBU sees a greater improvement than OMpy, with FBU now showing a clear upper hand with regards to accuracy. The error metrics are shown in table 6.5.

Table 6.5: Error metrics for refolded results (rounded). The columns named 'Uncut' are the results from table 6.3, while columns named 'Cut' contain the metrics for the refolded results in figure 6.23. We see all estimates have improved substantially for the cut raw spectrum, with R^2-scores all above 0.99. The error metrics also confirm that FBU has a better accuracy than OMpy for the cut spectrum.

	FBU Median		**FBU Mean**		**OMpy**	
	Cut	Uncut	Cut	Uncut	Cut	Uncut
MAE	8.3	28.9	8.1	29.2	12.5	26.0
R^2-score	0.999	0.968	0.999	0.968	0.993	0.955

Table 6.6: The mean variances for cut and uncut spectra. These are calculated on both the truth-samples output from FBU, and the same samples folded with the response. We see that cutting the spectrum reduces the variance for the truth-samples while not really impacting the variance for the folded samples.

	True samples		**Folded samples**	
	Cut	Uncut	Cut	Uncut
Mean variance	561.8	966.9	95.3	92.4



(a) (b)

Figure 6.24: Relative uncertainty plot, showing in the number of standard deviations, the level of relative uncertainty for each bin in the spectrum. The first plot (a) corresponds to the uncut raw spectrum (figure 6.20), with the second (b) to the cut spectrum. We see that the cut has significantly reduced the systematic discrepancies, while still showing some impact around bins 150-200, maybe due to further errors in the response.

6.2.3 High excitation energies (6.0-6.2 MeV)

We now investigate an area with more complex structures, i.e. a projection of the counts between $E_x = 6000$ keV and 6200 keV of the raw matrix in figure 6.11. As cutting the first 3 bins proved to greatly improve the previous results, we do the same for this spectrum.

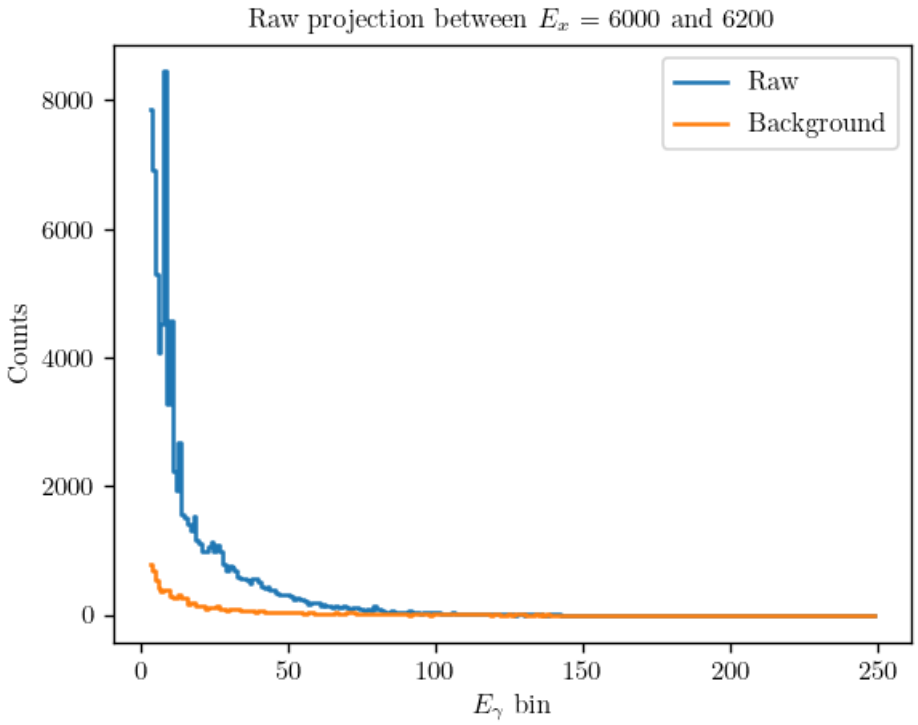
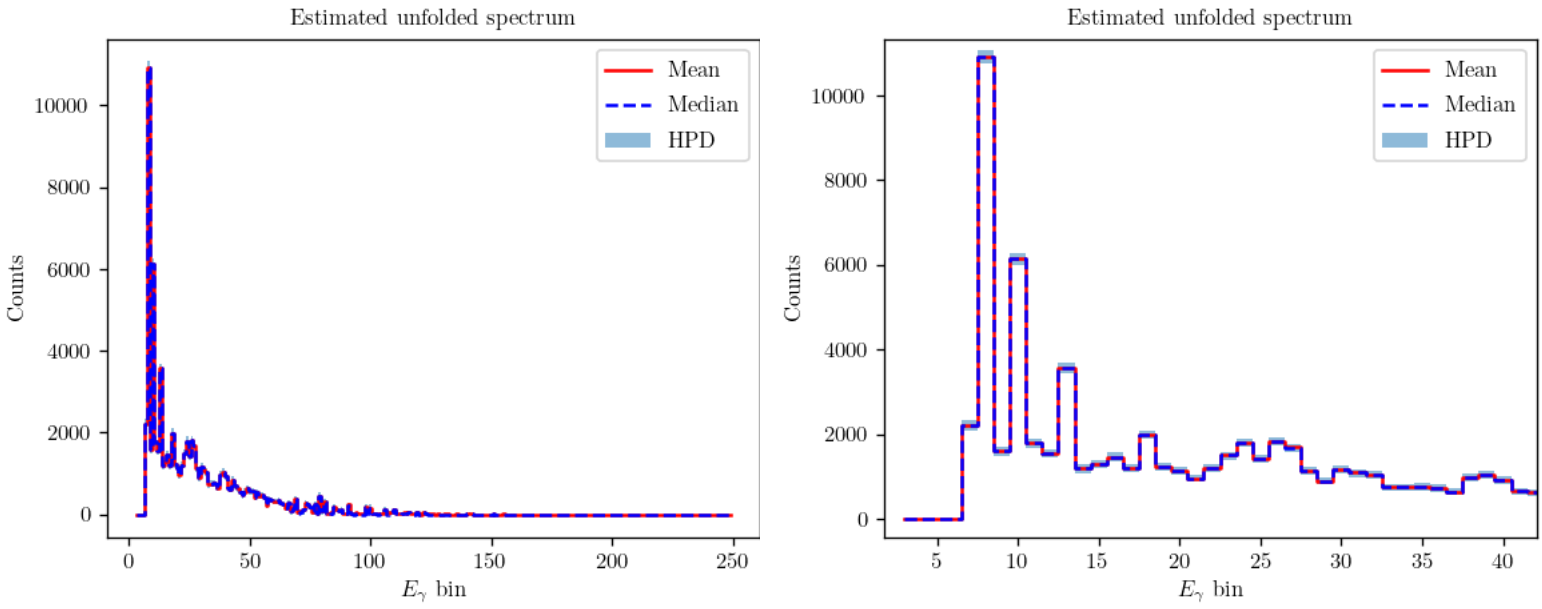
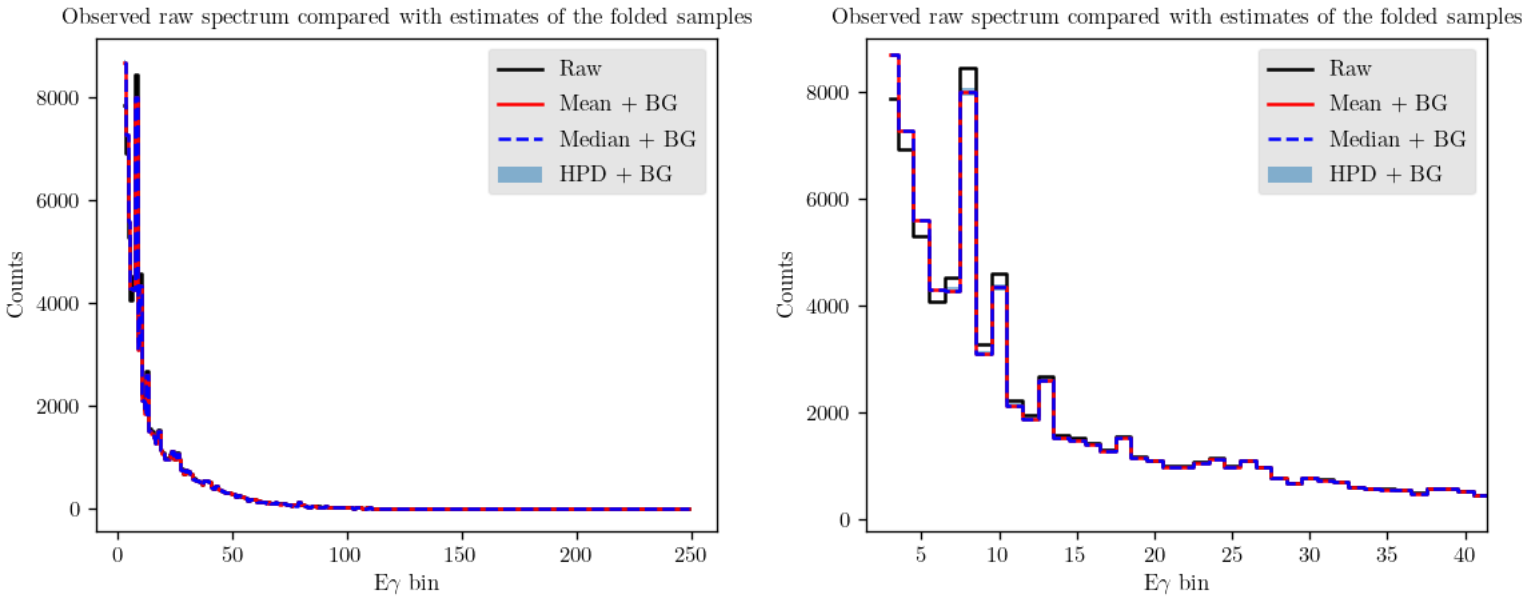


Figure 6.25: Projection of the raw spectrum and background for the high-energy area, i.e. between $E_x = 6000$ keV and 6200 keV, see figure 6.11, showing observed counts in each energy bin, with the first 3 bins cut.



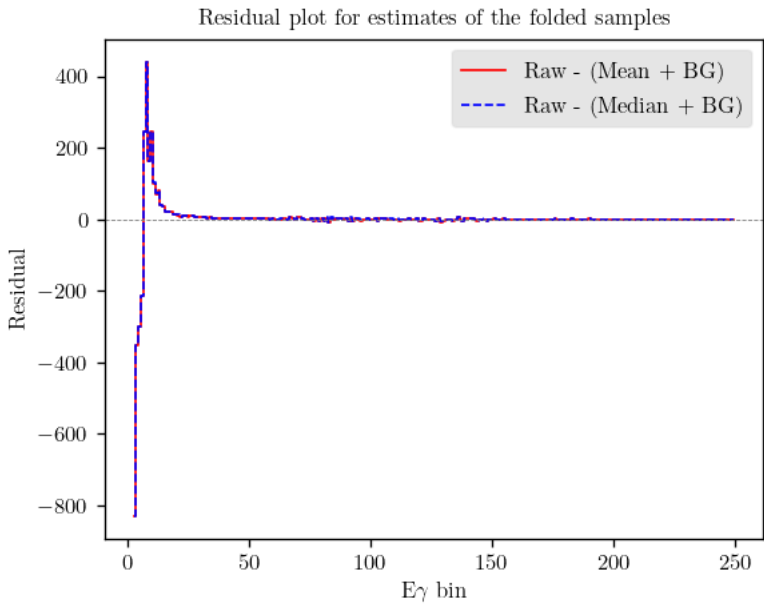
(a) (b)

Figure 6.26: The estimated unfolded spectrum from the raw spectrum in figure 6.25, for all bins (a) and zoomed to the first 40 bins (b). Here we see a more complex composition of peaks than for the first excited state. Further analysis is performed on this spectrum and shown in figures 6.27 and 6.28. The results from OMpy and subsequent comparisons are shown in figures 6.29, 6.30 and in table 6.7.

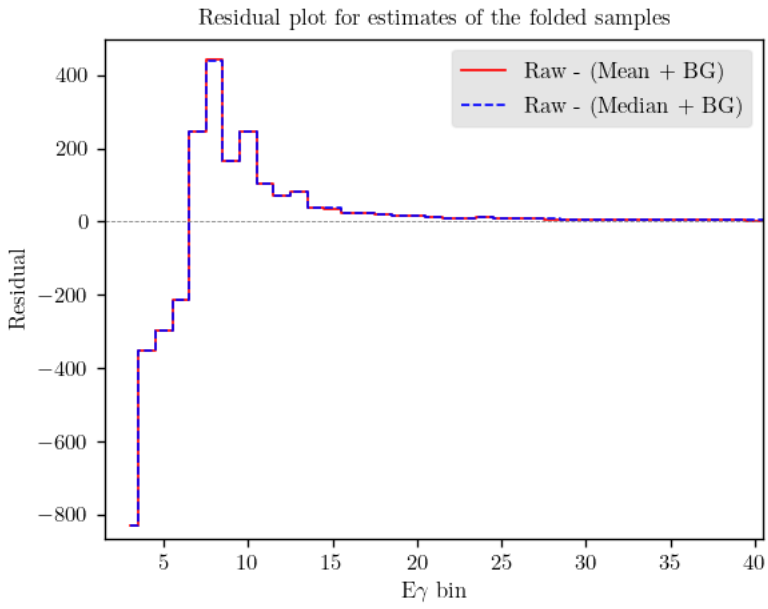


(a) (b)

Figure 6.27: Refolding of the unfolded result in figure 6.26 compared with the raw observed data, for all bins (a) and zoomed to the first 40 bins (b). We see that FBU keeps up with the added complexity and is able to match the spectrum closely.

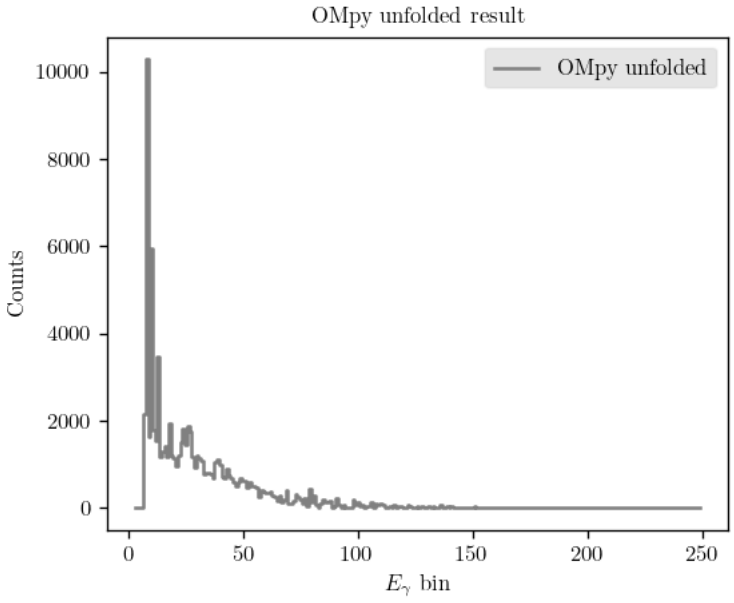


(a)

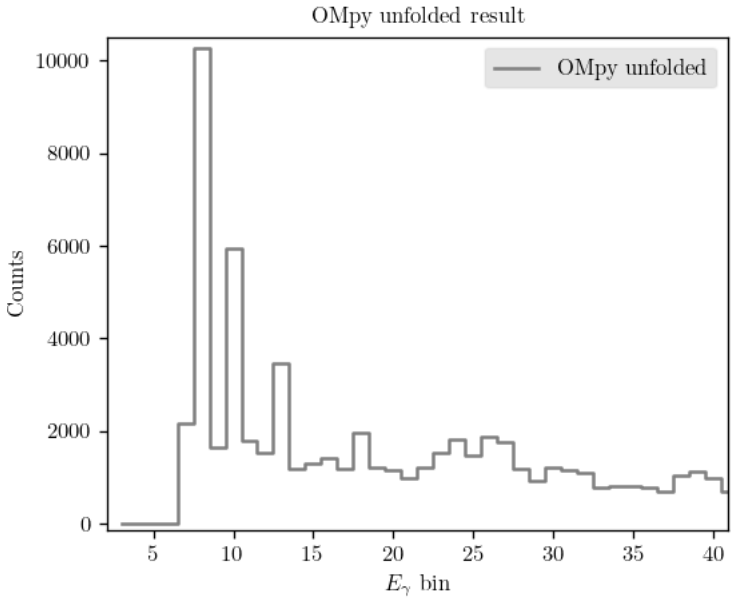


(b)

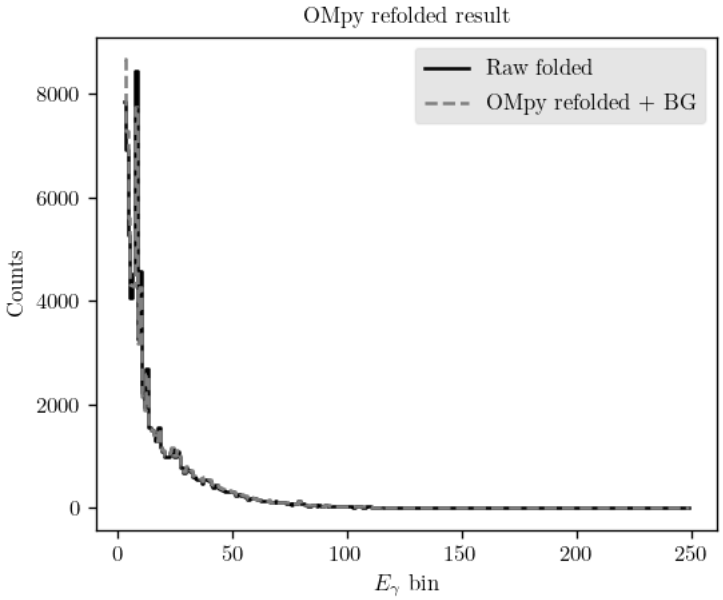
Figure 6.28: Residual plots showing the difference between the observed raw spectrum and the point estimates from FBU, first for all bins (a), then zoomed to the first 40 bins (b). We still observe the largest discrepancies for the first bins. This might be due to a combination of the mentioned response matrix errors and spectrum complexity.



(a)



(b)



(c)

Figure 6.29: Result from unfolding using OMpy for all bins (a) and zoomed to the first 40 bins (b), as well as the subsequent refolded result compared with the observed spectrum (c). These results look similar to those from FBU, and a closer look will be shown in figure 6.30.

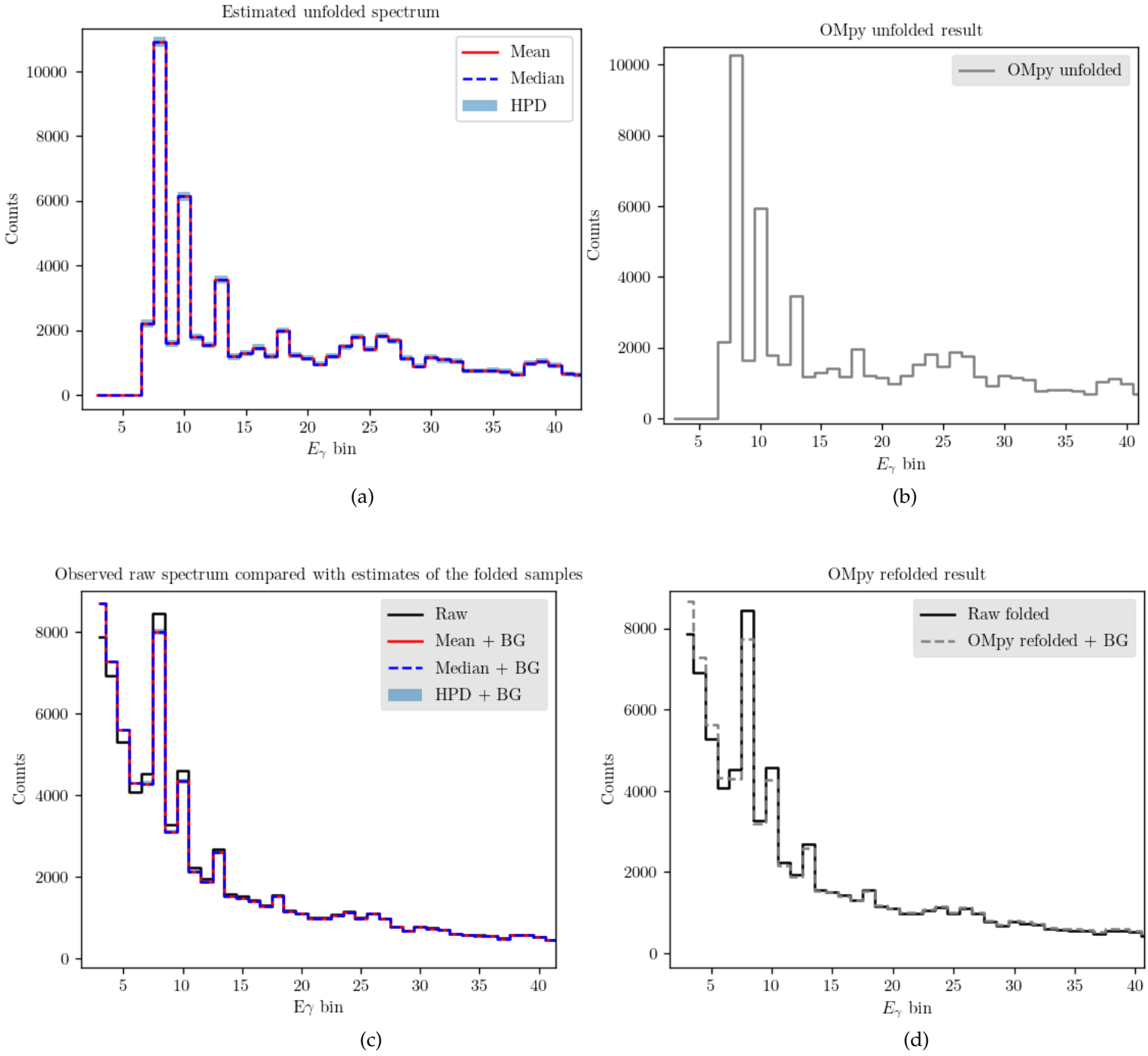


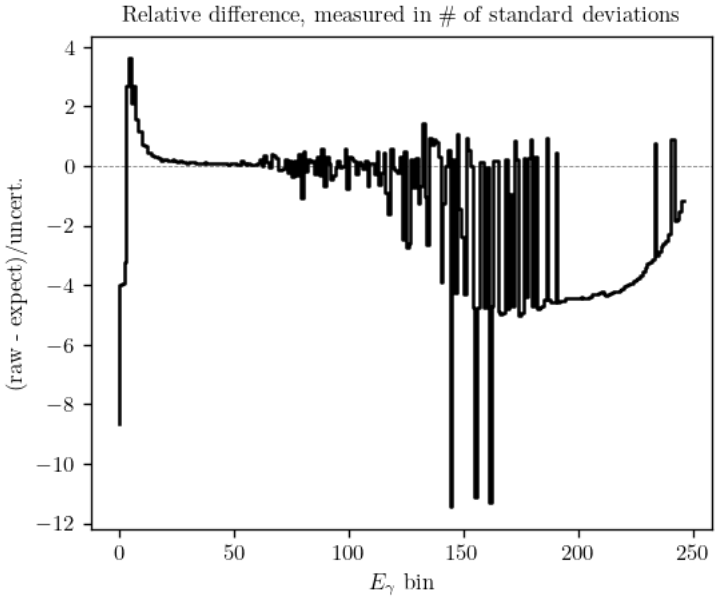
Figure 6.30: Zoomed versions of the unfolded (figures 6.26 and 6.29b) and refolded results (figures 6.27 and 6.29c) comparing FBU (a,c) with OMpy (b,d) for the first 40 bins. We see that FBU has redistributed more counts into the tallest peak through unfolding. We also see that for the refolded results, there is a sizeable discrepancy for the leftmost bin, likely another result of the response matrix imperfections. Still, we observe a slightly better accuracy by FBU, with the spectrum being more closely approximated than by OMpy. The following error metrics are shown in table 6.7.

Table 6.7: Error metrics for refolded results (rounded). Compared with the results for the simpler spectrum of the first excited state, table 6.5, we see a slight drop in accuracy. This is expected due to the increased complexity of the spectrum, however the accuracy remains very good with no R^2-scores below 0.99. Again, we see slightly better scores for FBU than OMpy, showing that FBU is a powerful method worthy of consideration.

	FBU Median	**FBU Mean**	**OMpy**
MAE	14.9	14.9	20.4
R^2-score	0.995	0.995	0.994

Table 6.8: The mean variance for both the truth-samples output from FBU, and the same samples folded with the response. As can be expected, the increased complexity of this raw spectrum has lead to an increase in variance for both sets of samples.

	Truth-samples	**Folded samples**
Mean variance	2570.4	244.5



(a)

Figure 6.31: Relative uncertainty plot, showing in the number of standard deviations, the level of relative uncertainty for each bin, corresponding to the cut raw spectrum. From figure 6.24, we saw that the cut removed the by far largest contribution to uncertainties, and the same is expected to be the case here. We see again some oscillations around bins 150-200, maybe due to further errors in the response than what has been removed by the cut.

Part V

Conclusion and future work

6.3 Conclusion

In this thesis we have explored Bayesian statistics on the case of unfolding in nuclear physics, with data sourced from the OSCAR detector array. We have gained a more thorough understanding of the inner workings of the somewhat cryptic PyFBU package and how to interpret its results. Possible changes have been formulated and implemented, which give the user increased flexibility by allowing for the creation of completely custom prior distributions. The requirement to supply prior bounds in keyword arguments, limiting the options to either uniform, truncated normal or triangular distributions, has been sidestepped by including the possibility to use the `Interpolated` class from PyMC3. The user may then completely control the shape and location of the distribution by defining the range of x-values (counts) and the corresponding pdf-values, the result of which is automatically normalized and having prior bounds be determined from the given x-values. Using this, we have successfully implemented and used the log-uniform distribution, currently not a part of PyMC3, for FBU. Some testing with FBU has been done with the uniform distribution, while all FBU results presented in this thesis has been from the log-uniform distribution. This change facilitates an essential part of Bayesian thinking, the freedom to choose and define our prior in accordance with our existing knowledge.

Using Fully Bayesian Unfolding, we have investigated the ^{28}Si and ^{146}Nd γ-ray spectra and produced candidate unfolded spectra in the attempt to reconstruct the actual true energies. For the ^{28}Si spectrum, we have used the new response matrix available in the OMpy library, which is a closer match to the experimental conditions for this spectrum than the old response used by Valsdóttir [4]. Using this and, to our knowledge, otherwise as equal a setup to Valsdóttir as possible, we originally find FBU to produce worse results with the new response. However, after cutting out 10 bins of the lowest energies of the raw spectrum, where the response matrices are known to contain errors, we find the new response to lead to results which are not only much better, but also outperforms the results from using the old response.

The ^{146}Nd spectrum has not been unfolded using FBU before, which allows this work to be a further test of the performance and adaptability of FBU compared to the standard method of unfolding by the nuclear physics group at UiO, the folding iteration method. The first projected spectrum, i.e the first excited state, has been unfolded using both FBU and OMpy (folding iteration method). It is found that the unfolded and subsequent refolded results are very similar, with good error metrics, FBU scoring slightly better. When cutting the first 3 bins of the raw spectrum, we find that the accuracy is much higher for

both methods, leading to R^2-scores all above 0.99, and very low mean absolute errors. FBU profits more of this cut, with a closer match between refolded and raw than OMpy. The second projected spectrum we investigate is one located higher on the E_x-scale, meaning more complex structures with fewer clear peaks and a higher spread of counts. Even with that, we find that FBU again performs very well, achieving a closer match between refolded and raw than OMpy, as well as better error metrics. Both methods have some accuracy loss due to the increased complexity, yet the performance is still very good. This result is a good testament to the accuracy of FBU and its ability to handle more advanced spectra, not to mention the added benefit of direct observation of uncertainty through the posterior distribution and credible intervals.

6.4 Future work

Several avenues are available for further exploration on the subjects included in this thesis. One of these regards the choice of prior distribution. There are, as mentioned, an uncountable amount of possible priors, and the extensions to PyFBU implemented here facilitates the ease of creation and thus testing of different priors adapted to different cases. In the same spirit, investigating other spectra, experimental and synthetic, is of interest to further map out the accuracy and adaptability of FBU.

Another example of future work considers the response matrix. As discussed, the response not being a perfect representation of the actual response of the detector will be misleading to FBU, which assumes the given matrix to be fully correct. An interesting extension would be one that includes uncertainties on different areas of the response matrix. These would be considered extra (nuisance) parameters, increasing the dimensionality of the problem. However, the resulting posteriors would then be marginalized over the extra parameters and take such uncertainties into account.

Computational performance is another area of interest. The sampling process of PyFBU usually needs a longer run time than the folding iteration method in OMpy. Introducing parallelization, GPU-utilization or further modification of the underlying PyMC3/Theano logic may prove to increase the performance of PyFBU. Other probabilistic libraries or sampling algorithms may be worth considering as well.

Bibliography

1. Choudalakis, G. *Fully Bayesian Unfolding* 2012. arXiv: 1201.4612 [physics.data-an].
2. Guttormsen, M., Tveter, T., Bergholt, L., Ingebretsen, F. & Rekstad, J. The unfolding of continuum γ-ray spectra. *Nuclear instruments & methods in physics research. Section A, Accelerators, spectrometers, detectors and associated equipment* **374,** 371–376. ISSN: 0168-9002 (1996).
3. Gerbaudo, D., Helsens, C. & Rubbo, F. *PyFBU* https://github.com/pyFBU/fbu.
4. Valsdóttir, V. M. *Exploring Fully Bayesian Unfolding for γ-ray Spectra* MA thesis (University of Oslo, Oslo, 2020).
5. Midtbø, J. E., Zeiser, F., Lima, E., Ingeberg, V. W. & RK, M. *oslocyclotronlab/ompy: Ompy v1.1.0* version v1.1.0. June 2020. https://doi.org/10.5281/zenodo.3898281.
6. D. S. Sivia, J. S. *Data analysis: A Bayesian Tutorial* 2nd ed. ISBN: 0–19–856831–2 (Oxford University Press, Great Clarendon Street, Oxford OX2 6DP, 2006).
7. Spanò, Francesco. Unfolding in particle physics: a window on solving inverse problems. *EPJ Web of Conferences* **55,** 03002. https://doi.org/10.1051/epjconf/20135503002 (2013).
8. Cowan, G. *A survey of unfolding methods for particle physics* in *Prepared for Conference on Advanced Statistical Techniques in Particle Physics, Durham, England* (2002), 18–22.
9. Zeiser, F. & Tveten, G. M. *oslocyclotronlab/OCL_GEANT4: Geant4 model of OSCAR* version v1.0.3. Aug. 2018. https://doi.org/10.5281/zenodo.1339347.
10. Zeiser, F. *et al. The energy response of the Oslo Scintillator Array OSCAR* 2020. arXiv: 2008.06240 [physics.ins-det].

11. Hoffman, M. D. & Gelman, A. *The No-U-Turn Sampler: Adaptively Setting Path Lengths in Hamiltonian Monte Carlo* 2011. arXiv: 1111.4246 [stat.CO].

12. Salvatier, J., Wiecki, T. V. & Fonnesbeck, C. Probabilistic programming in Python using PyMC3. *PeerJ Computer Science* **2.** Publisher: PeerJ Inc., e55. ISSN: 2376-5992. https://peerj.com/articles/cs-55 (2021) (Apr. 6, 2016).

13. Kucukelbir, A., Ranganath, R., Gelman, A. & Blei, D. M. *Automatic Variational Inference in Stan* 2015. arXiv: 1506.03431 [stat.ML].

14. Jaynes, E. T. *Probability Theory: The Logic of Science* (ed Bretthorst, G. L.) (Cambridge University Press, 2003).

15. Lee, P. M. *Bayesian statistics : an introduction* eng. Chichester, West Sussex ; 2012.

16. Edwards, W., Lindman, H. & Savage, L. J. Bayesian statistical inference for psychological research. *Psychological Review* **70.** Publisher: US: American Psychological Association, 193. ISSN: 1939-1471. http://psycnet.apa.org/fulltext/1964-00040-001.pdf (1964).

17. Cort J. Willmott & Kenji Matsuura. Advantages of the mean absolute error (MAE) over the root mean square error (RMSE) in assessing average model performance. *Climate Research* **30,** 79–82. https://www.int-res.com/abstracts/cr/v30/n1/p79-82 (2005).

18. *The Coefficient of Determination* [Online; accessed 2021-06-25]. Jan. 11, 2021. https://stats.libretexts.org/@go/page/553.

19. Theano Development Team. Theano: A Python framework for fast computation of mathematical expressions. *arXiv e-prints* **abs/1605.02688.** http://arxiv.org/abs/1605.02688 (May 2016).

20. Valsdóttir, V. M. *Exploring Fully Bayesian Unfolding for γ-ray Spectra - GitHub repository* https://github.com/valamaria89/Exploring-Fully-Bayesian-Unfolding-for-gamma-ray-Spectra (2020).

APPENDIX A

1-dimensional likelihood testing

Here follows a set of tests attempting to determine the 1-dimensional contribution of the likelihood by erroneously assuming it to take the form of a 1-dimensional Poisson distribution. This is included to show how attempting to decipher and decompose the outputs to their base terms without the sufficient understanding, can lead to frustration.

A.1 Synthetic data

For easier and more predictable testing, a synthetic data set is used for the raw spectrum. It consists of 3 excited states, a simplified representation of a physical case. For the following tests, we use the lowest state.

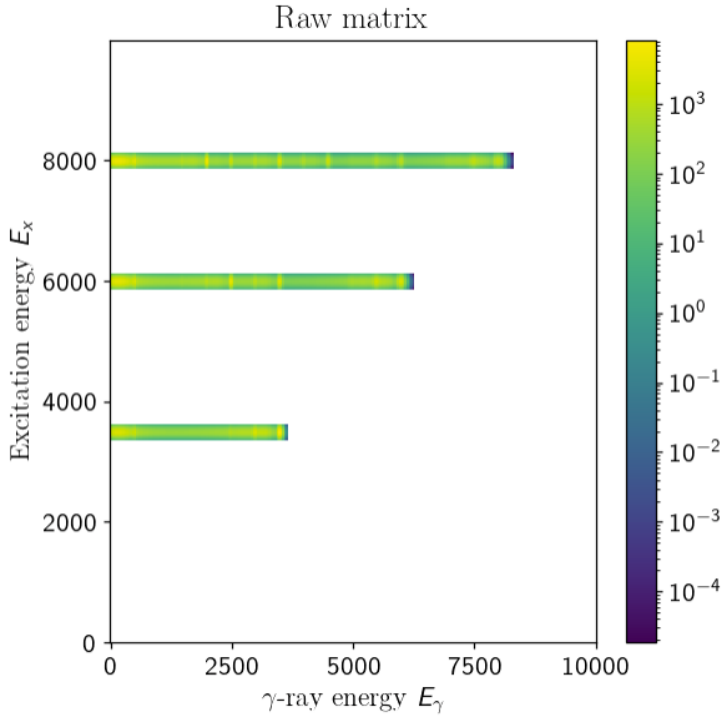


Figure A.1: Synthetic raw $E_\gamma - E_x$ matrix.

A.1.1 Bayesian terms

An interesting aspect to look at is the terms in Bayes' theorem after unfolding has been performed. We should be able to reproduce the shape of the resulting posterior samples by multiplying the prior and likelihood. As the likelihood depends on f_r, rather than the truth-values t, direct comparison between this and the other terms is not easy at first glance. To arrive at comparative foundation, we perform the following modifications to the prior and the posterior to achieve f_r-dependence:

$$P(\boldsymbol{T}|\boldsymbol{D})_{f_r} = \boldsymbol{TR}, \tag{A.1}$$

$$P(\boldsymbol{T})_{f_r} = \boldsymbol{SR}, \tag{A.2}$$

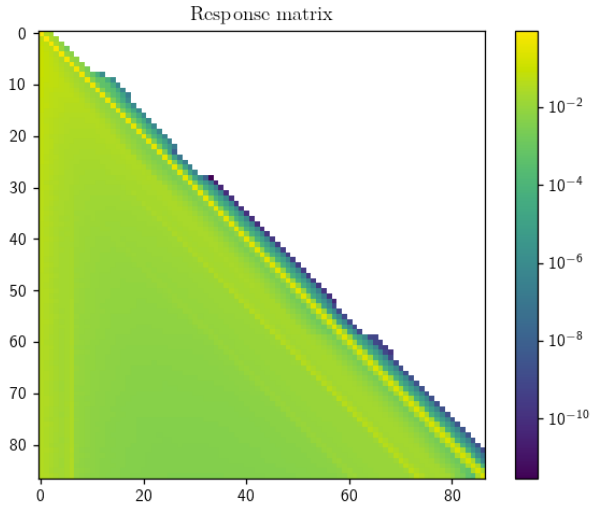
where $\boldsymbol{S}$ is a matrix containing random samples between the upper and lower prior limits, with shape corresponding to the output $\boldsymbol{T}$. There is no easy way of extracting the likelihood from the PyFBU package, nor its corresponding f_r-values. However, as we know that it is a Poisson distribution (**Note:** Here is where the *incorrect* assumption of the likelihood being composed of 1-dimensional independent Poisson distributions, was made. The full, multidimensional Poisson distribution is the correct representation of the likelihood.), we are able to define a range of f_r-values determined by the prior and posterior, and use these as input to equation (3.9).

A.1.2 Response matrices

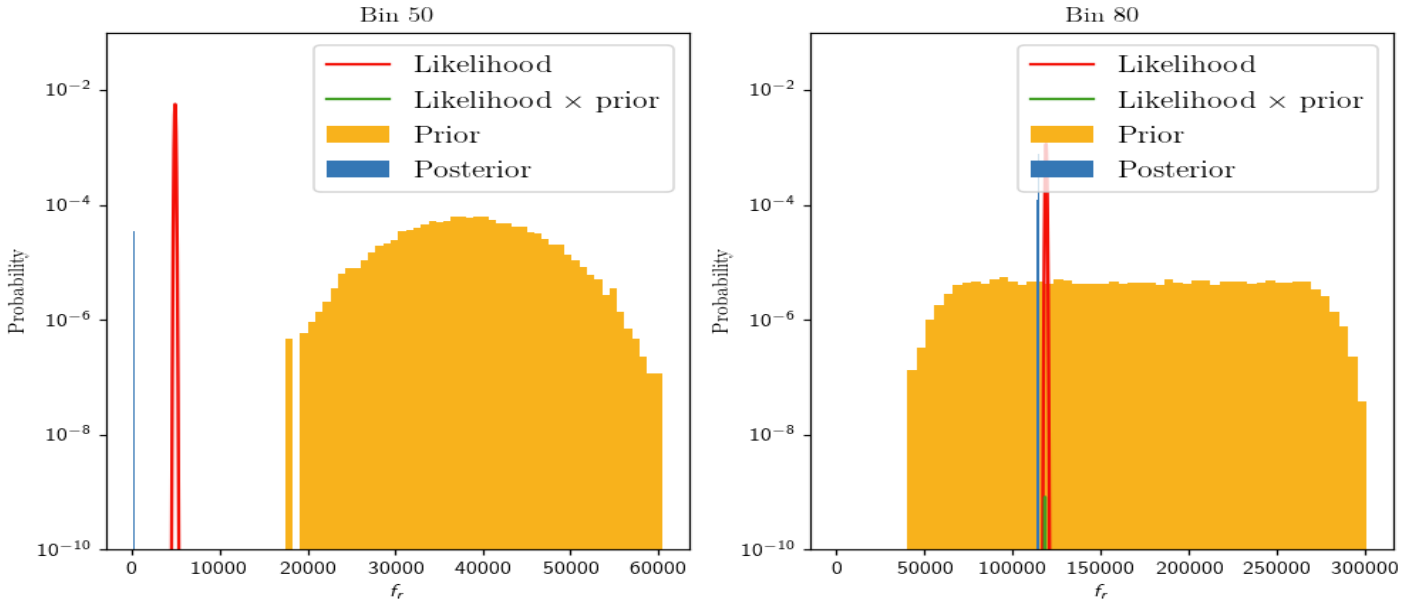
Here, different trial response matrices are tested for examining the impact on the final result, as well as the convergence of the implemented FBU method. As the response matrix is a vital part of the procedure, significant differences are expected when changes are made. In the end, the experimentally determined response should provide the best unfolded spectrum. It is however interesting to see the effect on the resulting Bayesian terms. If our assumption about the likelihood is correct, we should be able to see the posterior shape and location equal that of the product likelihood$\times$prior. Note that below, the response matrices are shown with the origin in the top left corner, when conventionally they are represented with the origin in the bottom left.

A.1.2.1 Normalized response from OCL

The following figures are results from running FBU using the response matrix from OCL, with normalization performed on each row. The bayesian terms do not match up like expected.



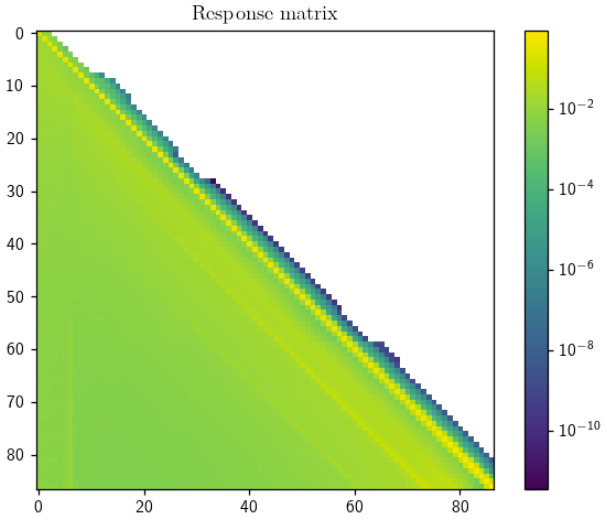
(a) Response matrix



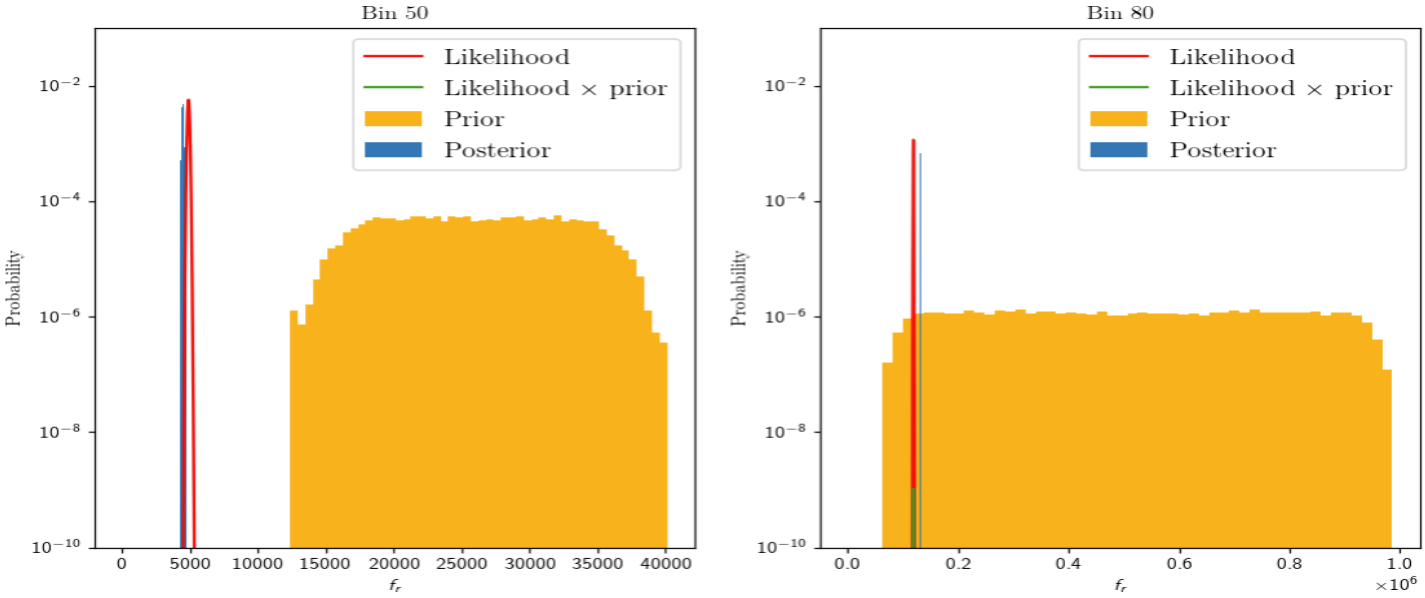
(b) The components of Bayes' theorem after unfolding, for the bins chosen above. Included is also a $L(\boldsymbol{D}|\boldsymbol{T}) \times P(\boldsymbol{T})$-function, which should have the same shape and position as the posterior, only differing by a normalisation constant.

A.1.2.2 Response from OCL with normalized columns

The following figures are results from running FBU using the response matrix from OCL, with normalization performed on each column instead. The bayesian terms do not match up like expected.



(a) Response matrix



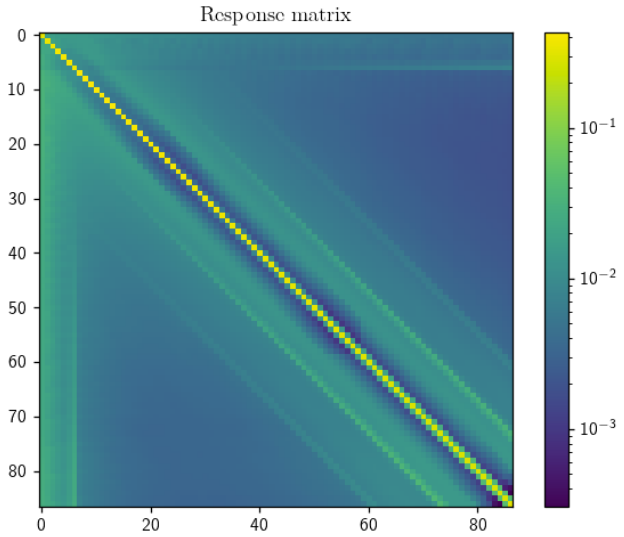
(b) The components of Bayes' theorem after unfolding, for the bins chosen above. Included is also a $L(\boldsymbol{D}|\boldsymbol{T}) \times P(\boldsymbol{T})$-function, which should have the same shape and position as the posterior, only differing by a normalisation constant.

A.1.2.3 Symmetrized response from OCL

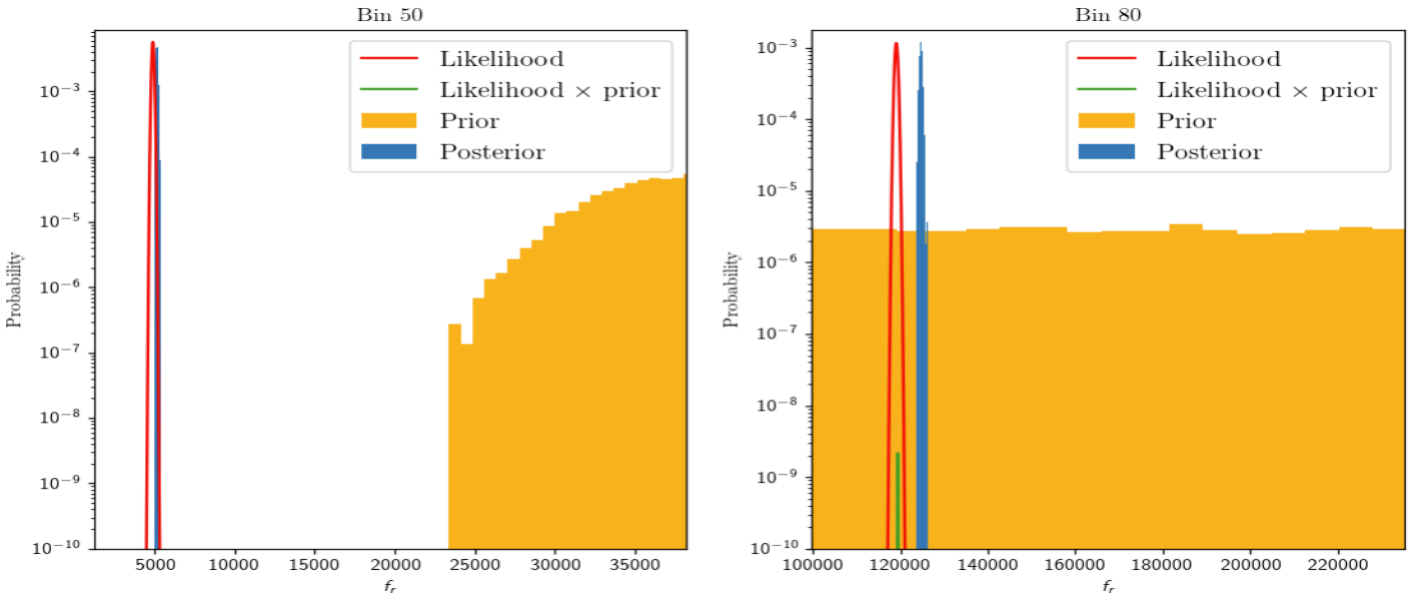
The following is produced using a symmetrized version of the OCL response matrix, that is

$$R = \frac{R_{OCL} + R_{OCL}^T}{2} \tag{A.3}$$

The bayesian terms do not match up like expected.



(a) Response matrix



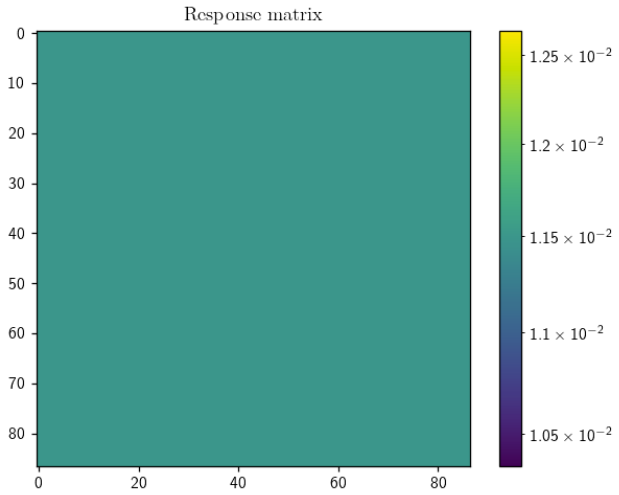
(b) The components of Bayes' theorem after unfolding, for the bins chosen above. Included is also a $L(\boldsymbol{D}|\boldsymbol{T}) \times P(\boldsymbol{T})$-function, which should have the same shape and position as the posterior, only differing by a normalisation constant.

A.1.2.4 Response as a normalized matrix of ones

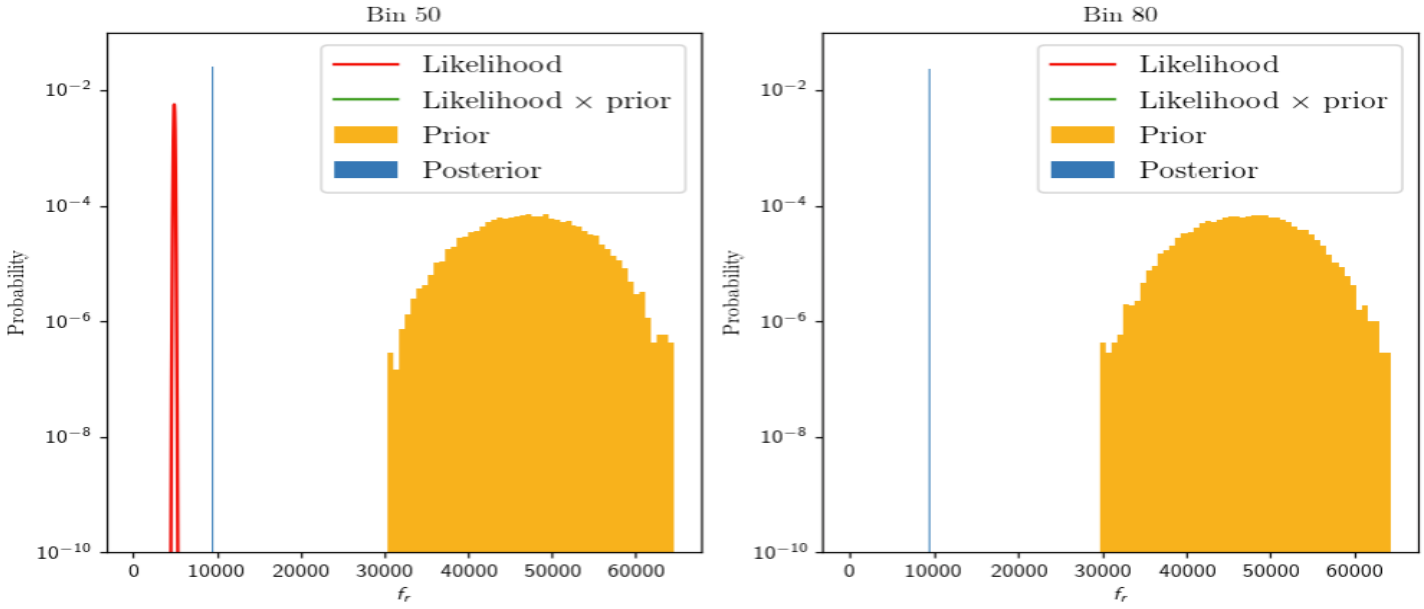
Now, the response is a normalized matrix of ones:

$$R^{N\times N} = \begin{bmatrix} 1/N & 1/N & \dots \\ 1/N & \ddots & \\ \vdots & & \ddots \end{bmatrix} \tag{A.4}$$

The bayesian terms do not match up like expected.



(a) Response matrix

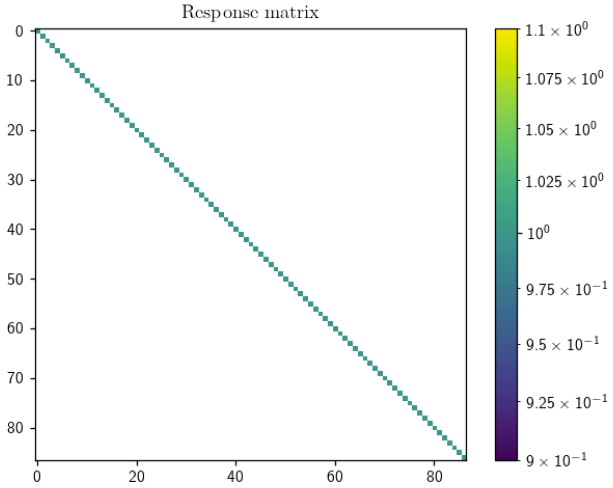


(b) The components of Bayes' theorem after unfolding, for the bins chosen above. Included is also a $L(\boldsymbol{D}|\boldsymbol{T}) \times P(\boldsymbol{T})$-function, which should have the same shape and position as the posterior, only differing by a normalisation constant.

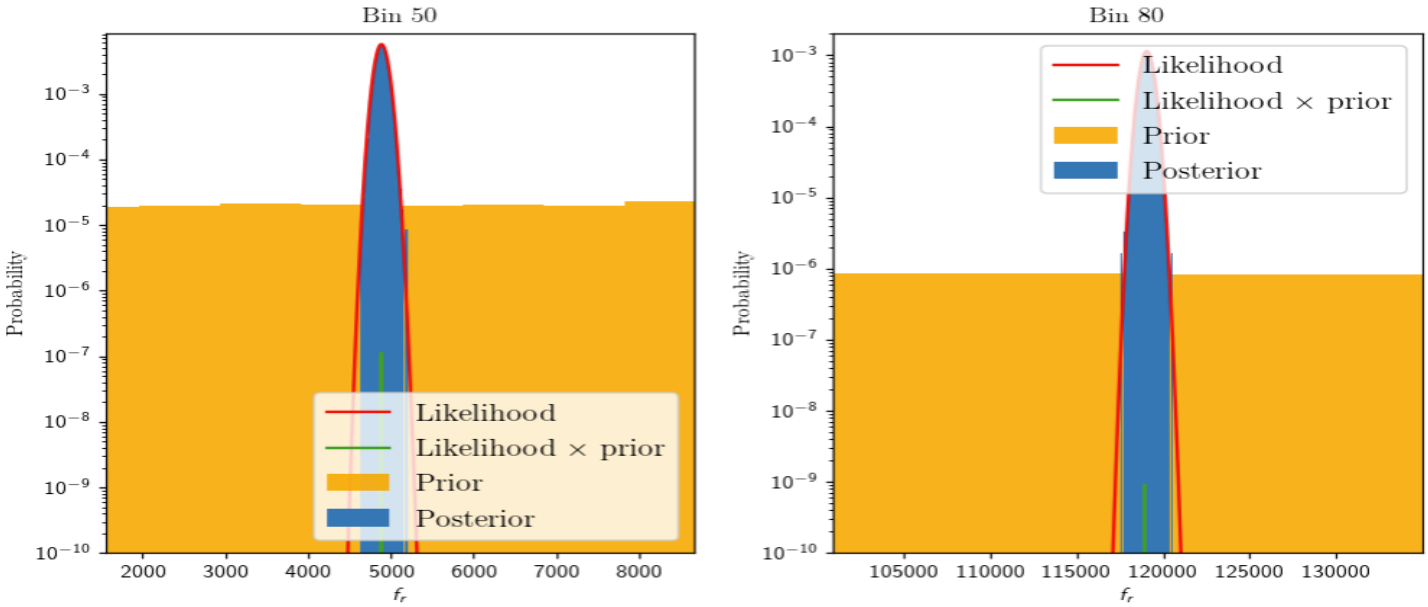
A.1.2.5 Response as the identity matrix

The results below are produced using an identity response matrix.

$$R = \mathbb{I} \tag{A.5}$$



(a) Response matrix



(b) The components of Bayes' theorem after unfolding, for the bins chosen above. Included is also a $L(\boldsymbol{D}|\boldsymbol{T}) \times P(\boldsymbol{T})$-function, which should have the same shape and position as the posterior, only differing by a normalisation constant.

Suddenly the Bayesian terms match up, maybe leading one to think that the response matrix is the culprit for the mismatches earlier. This is incorrect, the match here is due to the total likelihood in this specific case, actually being composed of 1-dimensional in-

dependent Poisson distributions. The identity matrix lead to no cross-bin dependencies after all, in fact it represents a 100% perfect detector, always reproducing the true spectra.

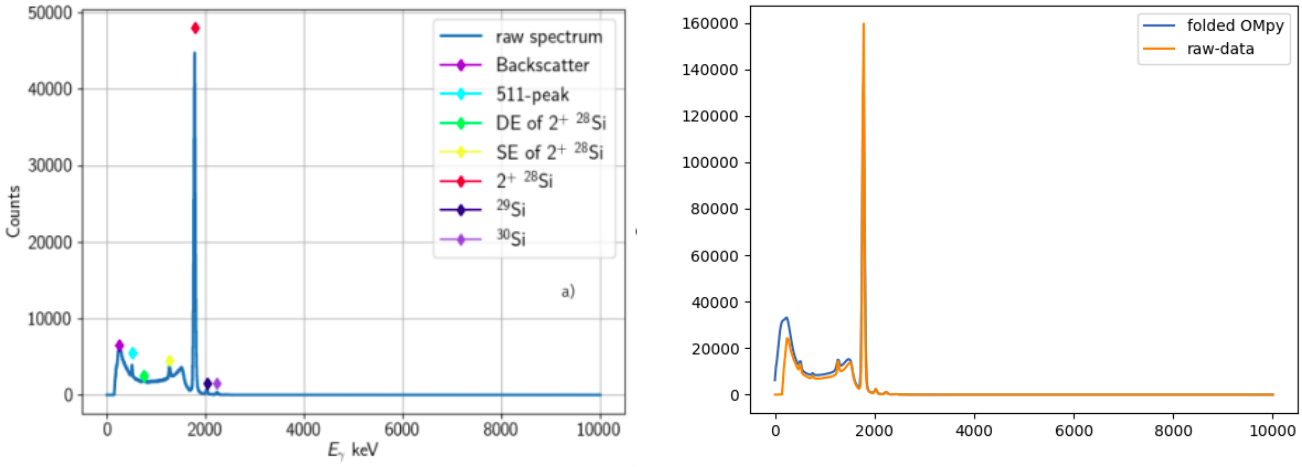
APPENDIX B

Reproduction of results

Under follows an early reproduction of the results achieved by Valsdóttir [4]. This procedure is done to verify the results using the same methods.

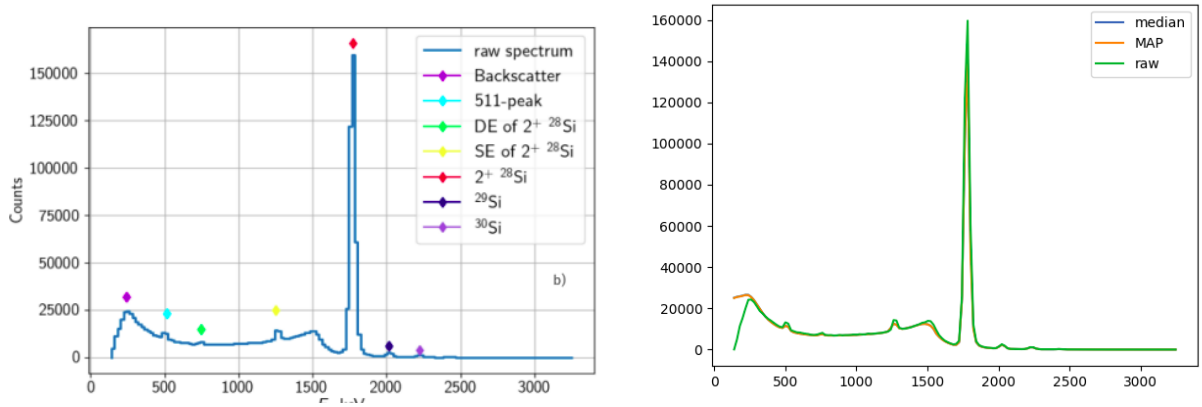
The figures presented as reproductions are outputs of the publicly available Jupyter notebooks on Valsdóttirs GitHub repository [20]. Some discrepancies are seen, pointing to the possibility that the results in Valsdóttirs thesis may stem from newer, locally stored versions of the files that have not been made accessible on the repository. Therefore, the below sections contain only the results for which corresponding output was found to be produced in the mentioned Jupyter notebooks.

B.1 The first excited state of ^{28}Si



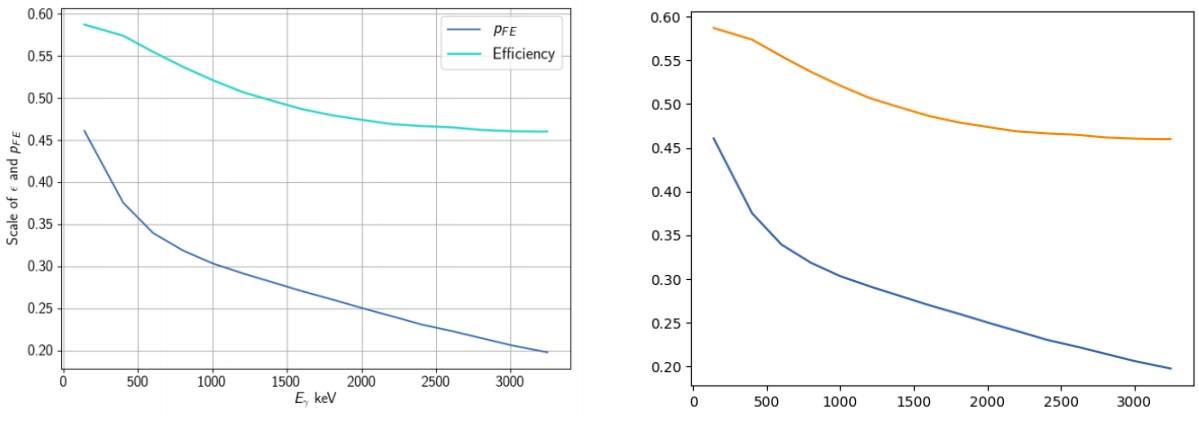
(a) Valsdóttir (b) Reproduction

Figure B.1



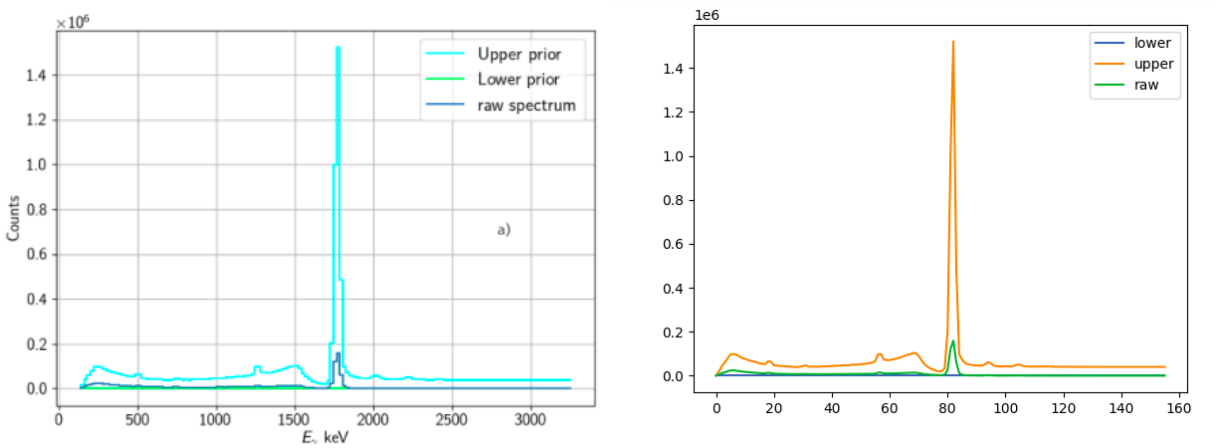
(a) Valsdóttir (b) Reproduction

Figure B.2



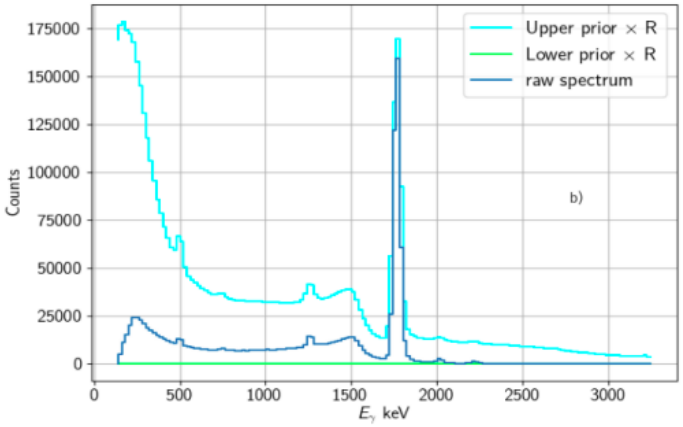
(a) Valsdóttir (b) Reproduction

Figure B.3

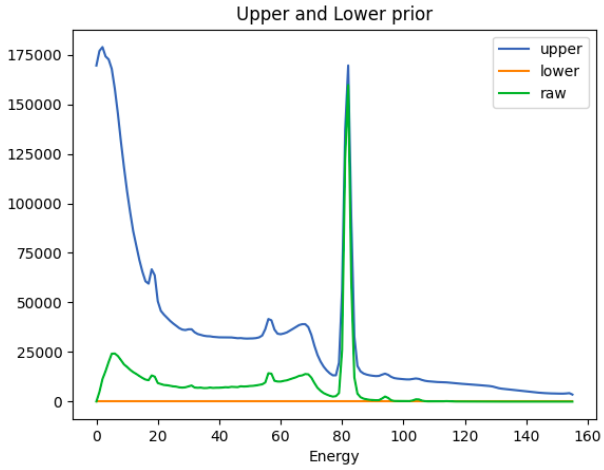


(a) Valsdóttir (b) Reproduction

Figure B.4

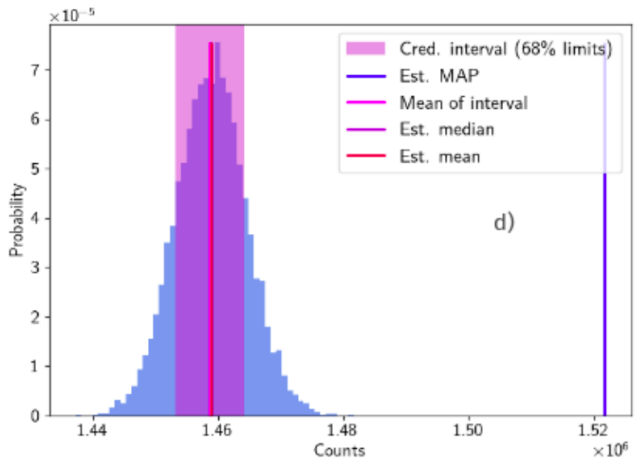


(a) Valsdóttir

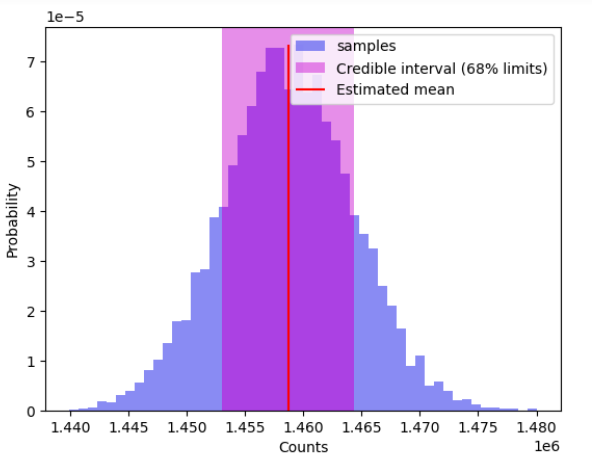


(b) Reproduction

Figure B.5



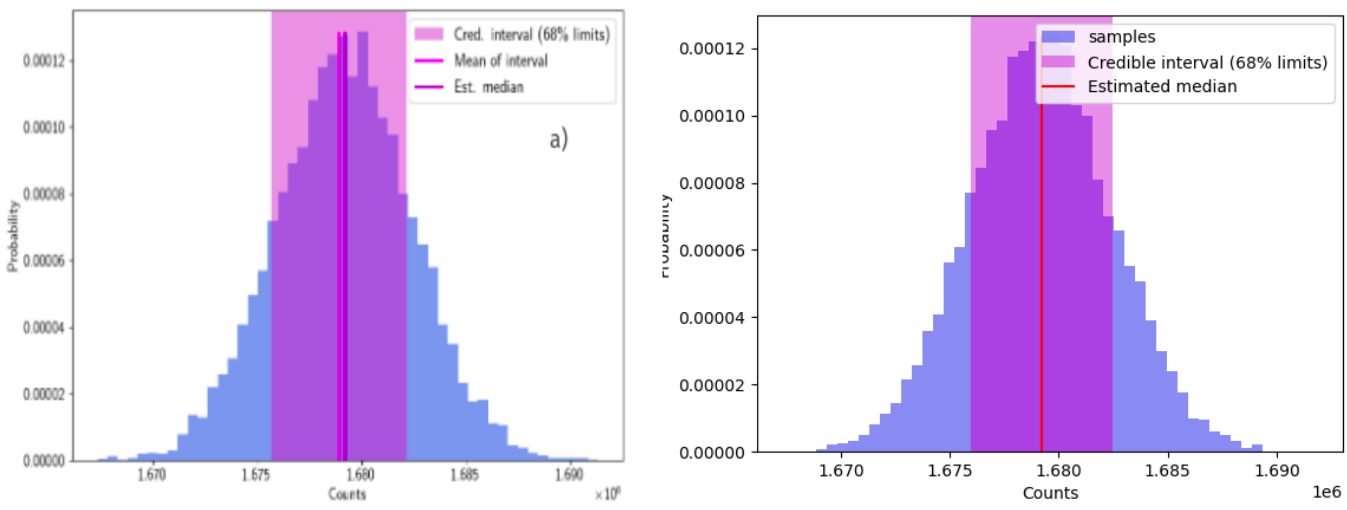
(a) Valsdóttir



(b) Reproduction

Figure B.6

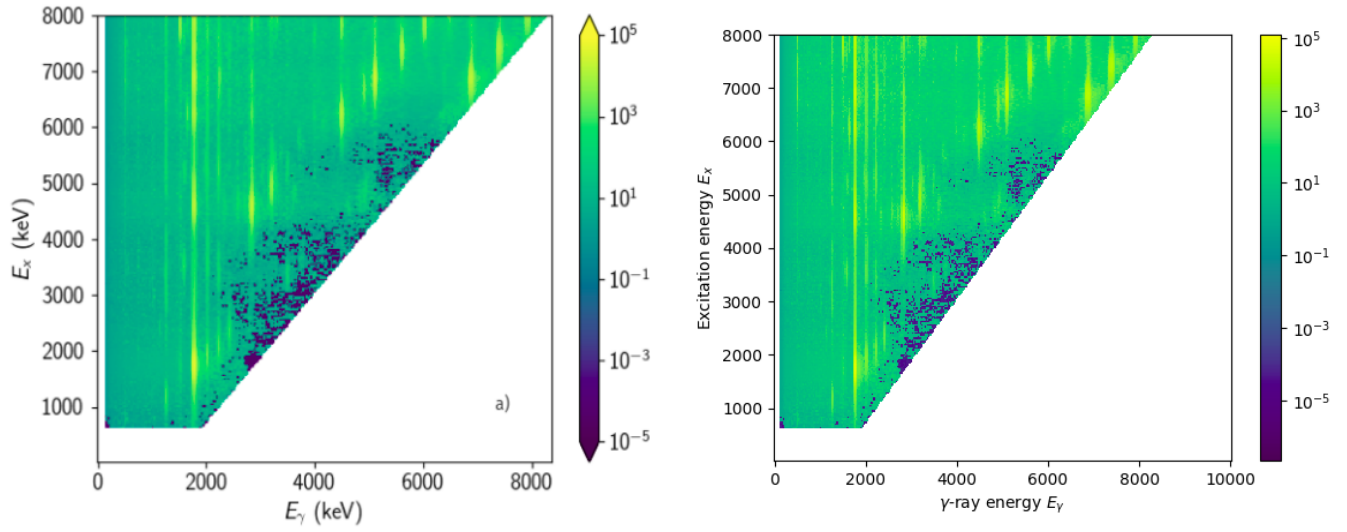
B.2 The first excited state of ^{28}Si including background



(a) Valsdóttir (b) Reproduction

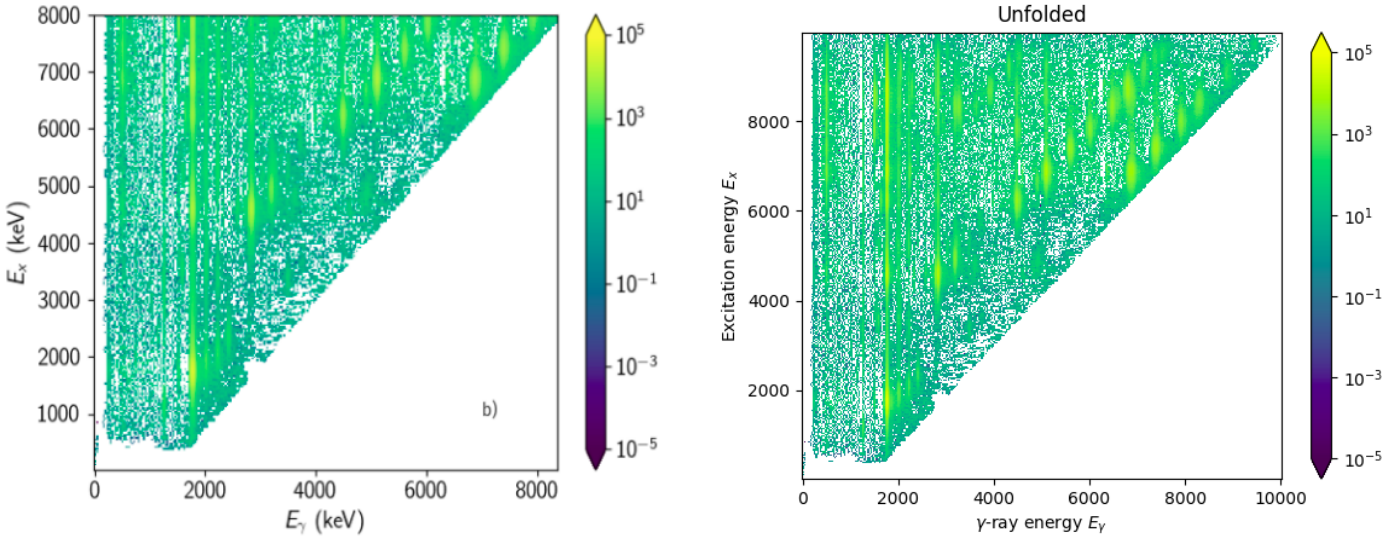
Figure B.7

B.3 All excited states



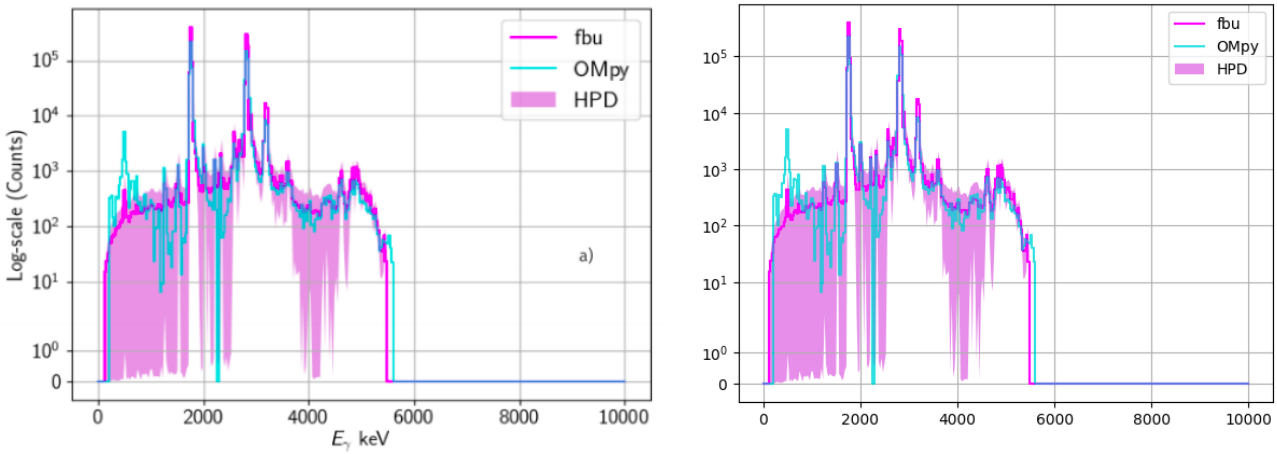
(a) Valsdóttir (b) Reproduction

Figure B.8



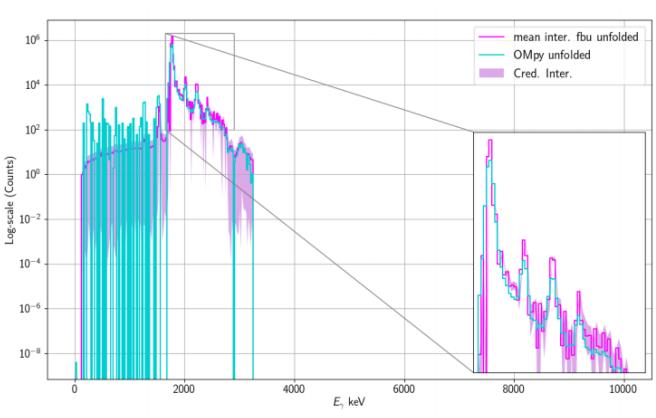
(a) Valsdóttir (b) Reproduction

Figure B.9

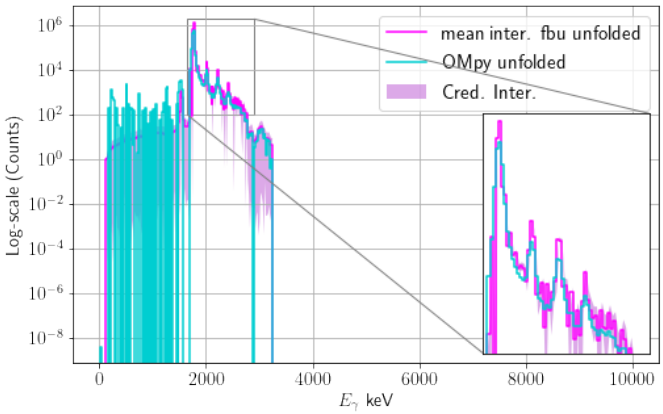


(a) Valsdóttir (b) Reproduction

Figure B.10



(a) Valsdóttir



(b) Reproduction

Figure B.11