technische universität dortmund

SFB 876 Providing Information by Resource-Constrained Data Analysis

DSEA Rock-Solid

Regularization and Comparison with other Deconvolution Algorithms

– Master's Thesis –

Submitted in partial fulfillment of the requirements for the degree of Master of Science

by

Mirko Bunse

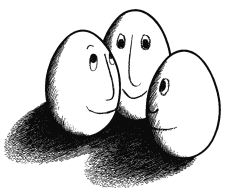
Dortmund, July 2018

Supervisors:

Prof. Dr. Katharina Morik

Dr. Christian Bockermann

Technische Universität Dortmund
Fakultät für Informatik
Lehrstuhl für Künstliche Intelligenz, LS VIII
D-44221 Dortmund



– Abstract –

Deconvolution reconstructs the distribution of a physical quantity from related quantities. The present work surveys popular deconvolution methods with a comprehensive benchmark targeted on Cherenkov astronomy. Meanwhile, a novel unified view on deconvolution is established from the perspective of machine learning. Within this view, the essential building blocks of deconvolution methods are identified, opening the subject to several directions of future work. Particular attention is turned to Dsea, the Dortmund Spectrum Estimation Algorithm, which employs a classifier to reconstruct the sought-after distribution. An improved version of this algorithm, Dsea$^+$, is proposed here. This version is more accurate and it converges faster than the original Dsea, thus matching the state of the art in deconvolution.

Acknowledgements: Let me express my deep gratitude to my friends, family members, colleagues, and supervisors, who supported me with enlightening discussions, instructive feedback, and with their faith in my work.

I also want to thank the FACT collaboration and the MAGIC collaboration for providing me with the data simulated for their telescopes.

This work has been supported by the DFG, Collaborative Research Center SFB 876, project C3 (`http://sfb876.tu-dortmund.de/`).

Contents

Chapter 1

Introduction

Obtaining the distribution of a physical quantity is a frequent objective in experimental physics. A reliable estimate of such distribution can tell us a lot about the related phenomena, allowing us to reconcile our scientific assumptions with the results obtained in an experiment. In cases where the distribution of the relevant quantity cannot be measured directly, it has to be reconstructed from distributions of related but different quantities that are measured, instead. This reconstruction is called *deconvolution*.

Cherenkov astronomy is a deconvolution use case which studies the energy distribution of cosmic γ radiation to reason about the characteristics of celestial objects emitting such radiation. One prominent tool in this field are Imaging Air Cherenkov Telescopes (IACTs), which are placed at high altitudes on the surface of our planet. Since it is not viable to detect high-energy γ particles directly there, IACTs record Cherenkov light emitted by a cascade of secondary particles, instead. Such cascades, called air showers, are induced by γ particles interacting with Earth's atmosphere. Since the γ radiation is not directly measured by the telescopes, deconvolution is applied to reconstruct the energy distribution from the related Cherenkov light recorded by IACTs. Figure 1.1 presents this setup.

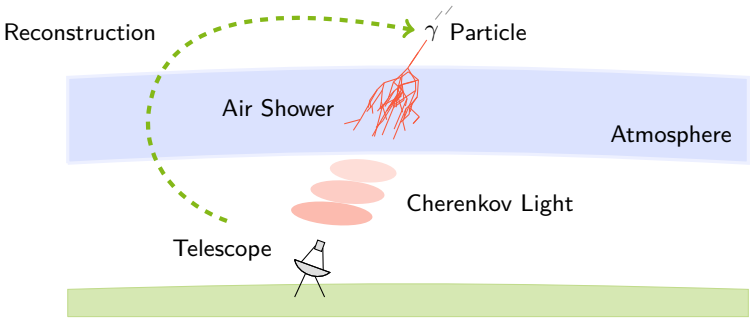


Figure 1.1: A γ particle hitting Earth's atmosphere produces a cascade of secondary particles, the air shower. This shower emits Cherenkov light, which is measured by an IACT. The distribution of γ particles is reconstructed from IACT measurements [1].

1.1 The Deconvolution Problem

The task of deconvolution is to estimate a density function $f : \mathcal{Y} \to \mathbb{R}$ of a physically relevant quantity. Formally, this quantity is a random variable Y with the state space $\mathcal{Y}$. The most prominent quantity in Cherenkov astronomy is the energy of γ particles, of which the Cherenkov light is observed by an IACT. Researchers reason about the characteristics of a monitored γ ray source by interpreting the density function of its particle energies. Estimating any density f can be complicated by the following deficiencies, which may be inherent to the respective experimental setup [2]:

Transformation When Y cannot be measured directly, one has to measure a related but different quantity X instead. Only with sufficient knowledge about the relation between X and Y, f can be reconstructed from the density $g : \mathcal{X} \to \mathbb{R}$ of the measured quantity. In case of IACTs, the features of the observed light cone, e.g. its shape and size, are measurable quantities that are related to the particle energy Y.

Finite resolution The measurement accuracy may be limited, resulting in measurements that do not perfectly resemble the actual values of the measured quantity X. For example, when an IACT observes an air shower only partially, it may fail to capture the exact shape and size of the corresponding light cone.

Background noise Additional events may be recorded which do not originate from the respective process under study. For example, not every observed air shower is induced by a γ particle emitted by the monitored γ ray source.

Limited acceptance The detector may fail to recognize some of the events that occur in the process under study. For example, IACT observations are dropped when it is not sufficiently certain that they are induced by a γ particle.

A reliable estimate of the relevant density f is only obtained, if these deficiencies are rectified, i.e. if f is appropriately reconstructed from the measured density g. In Cherenkov astronomy, this boils down to reconstructing the density of γ particle energies from IACT measurements. The procedure of reconstructing f with respect to the experimental deficiencies is called *deconvolution*, being also known as "unfolding" or "unsmearing". The name of this procedure is motivated by g being modeled as a convolution of f with a detector response function $R : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$.

$$g(x) \;=\; \int_{\mathcal{Y}} R(x \,|\, y) \cdot f(y) \; \mathrm{d}y \tag{1.1}$$

The detector response R provides the link between X and Y, representing the background knowledge about their relation. Specifically, $R(x \,|\, y)$ represents the conditional probability of measuring some $x \in \mathcal{X}$ when the actual value of the relevant quantity is $y \in \mathcal{Y}$. To obtain f, this model of g has to be inverted—it has to be "deconvolved". This is done by fitting f to the model of g, when g and R are given.

Readers familiar with deconvolution may have already noticed that the nomenclature in Equation 1.1 is slightly different from the notation that is conventional in particle physics. Specifically, the roles of the letters x and y are switched here for compliance with the notation that is common in machine learning. Here, y refers to the value of the target variable and x is the observed value. This notation also clarifies the relation between the deconvolution problem and classification. The difference between these two tasks is merely that deconvolution aims at estimating the density of Y while classification estimates each $y \in \mathcal{Y}$ individually. A mapping between the nomenclatures motivated by physics and machine learning is given in Section A.1.

1.2 Deconvolution with DSEA

The present thesis turns particular attention to the Dortmund Spectrum Estimation Algorithm (DSEA) [3, 4], which translates the deconvolution problem into a multinomial classification task. The labels of this task are half-open ranges $\mathcal{Y}_i = [\, t_i,\, t_{i+1}[\subseteq \mathcal{Y}$ with the range boundaries $t_i \in \mathcal{Y}$. Each of the $\mathcal{Y}_i$ corresponds to one of the I components in a discrete estimate $\hat{\mathbf{f}} = (\hat{\mathbf{f}}_1, \dots \hat{\mathbf{f}}_I)$ of the continuous target density f. DSEA trains a machine learning model $\mathcal{M}$ which predicts the range $\mathcal{Y}_i$ that each observation $\mathbf{x} \in \mathcal{X}$ belongs to. The model is then used to obtain confidence values $c_{\mathcal{M}}(\, i \,|\, \mathbf{x}_n)$ for the labels, given each of the N individual observations. Equation 1.2 averages these confidence values, thus obtaining $\hat{\mathbf{f}}$ as a density of confidences over the labels. This density is reported to be a suitable discrete estimate of f [3, 4]. It is updated multiple times by iteratively adapting the weights of the training examples that induced $\mathcal{M}$.

$$\hat{\mathbf{f}}_i \;=\; \frac{1}{N} \sum_{n=1}^{N} c_{\mathcal{M}}(\, i \,|\, \mathbf{x}_n) \qquad \forall \;\; 1 \leq i \leq I \tag{1.2}$$

Figure 1.2 presents the density of a toy data set, as it is estimated by DSEA after three iterations. The toy data consists of the relevant quantity Y and ten additional attributes that are used to predict that quantity. We see that there is good agreement between this result and the true (discrete) density $\mathbf{f}$. Figure 1.3 presents the distance between the true $\mathbf{f}$ from Figure 1.2 and $\hat{\mathbf{f}}^{(k)}$, which is the estimated density in iteration k. From the distance increasing with k, it is apparent that DSEA diverges from the true $\mathbf{f}$ after having found an appropriate estimate like $\hat{\mathbf{f}}^{(3)}$. This undesirable property of the algorithm is also observed on other data sets.

Besides DSEA, there are several other deconvolution algorithms. Two of the most popular approaches are the Regularized Unfolding [5, 2], which maximizes the likelihood of the estimated density $\hat{\mathbf{f}}$, and the Iterative Bayesian Unfolding [6, 7], which iteratively applies Bayes' rule starting from an initial guess $\hat{\mathbf{f}}^{(0)}$. Unfortunately, it is not clear which of these methods is most suitable for IACTs.

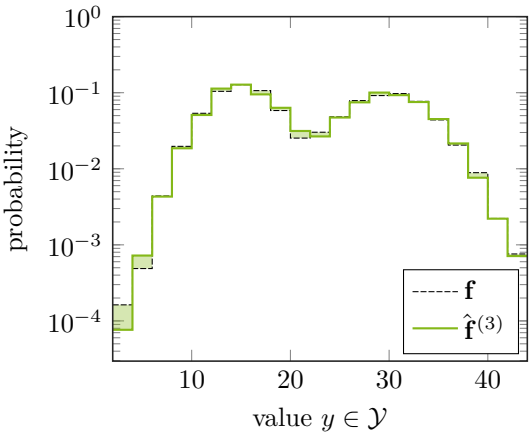


Figure 1.2: The deconvolution result $\hat{\mathbf{f}}^{(3)}$ is obtained after the third iteration of DSEA. It accurately resembles the true density $\mathbf{f}$. The toy data set used to obtain this result is described in Subsection 4.2.1. It is employed throughout this thesis.

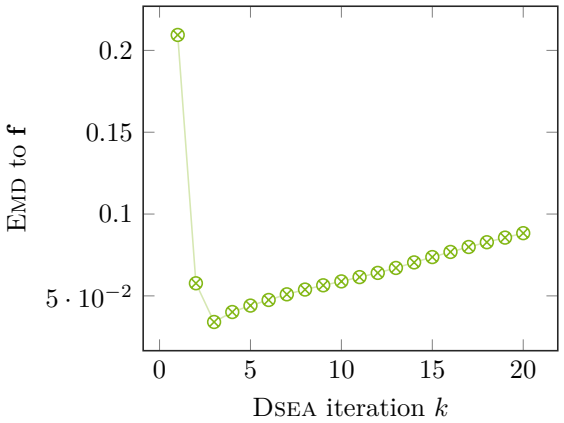


Figure 1.3: The distance of the DSEA estimate $\hat{\mathbf{f}}^{(k)}$ from the true density $\mathbf{f}$ is presented over the iteration number k. Here, high values indicate a low accuracy of the k-th estimate. The algorithm diverges from the optimal solution $\mathbf{f}$ after achieving the suitable estimate $\hat{\mathbf{f}}^{(3)}$ already presented in Figure 1.2. The distance metric employed here is the Earth Mover's Distance (EMD), which is described in Section 4.1. This metric is used throughout this thesis.

1.3 Goals and Outline

Since DSEA diverges from the true density after a few iterations, it has to be stopped in time. Usually, the practitioner makes an educated guess about the right number of iterations by trying the algorithm on simulated data first, before applying DSEA to the actual observations. A stopping criterion proposed for DSEA is rarely used in practice because it is parametrized based on simulated data in a similar fashion. If DSEA did not diverge that much, it would be less critical to find a suitable stopping criterion. Stopping the divergence relates to the concept of regularization, which limits the freedom of an algorithm to achieve more robust results. In short, the first goal of the thesis is to:

G1 Stop DSEA from diverging from the true density.

It is still unclear, which of the existing deconvolution methods is best suited for Cherenkov astronomy. Moreover, a unified notation of these methods is still missing, making it hard to identify the building blocks that make an algorithm suitable for this use case. The second goal therefore is to:

G2 Find out, which algorithm is best suited for deconvolution in IACTs.

The remainder of this thesis is structured as follows: Chapter 2 describes DSEA and two other popular deconvolution algorithms, namely the Regularized Unfolding and the Iterative Bayesian Unfolding. These algorithms are presented in a unified view that is given from the perspective of machine learning. Improvements to DSEA are proposed in Chapter 3 and a comprehensive evaluation of the algorithms is presented in Chapter 4. This evaluation is specifically targeted at the use case of Cherenkov astronomy, for which conclusions are drawn in Chapter 5. Figure 1.4 presents this outline, loosely mapping the chapters to the goals mentioned above.

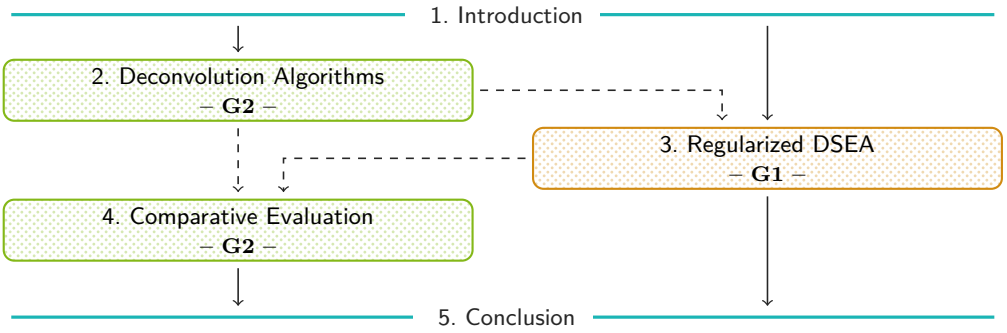


Figure 1.4: The outline of the present thesis addresses the two goals. Solid arrows indicate that one chapter strongly depends on the other. Dashed arrows express that there is some dependency between chapters, but that the one chapter can be understood independently from the other.

Chapter 2

Deconvolution Algorithms

Literature proposes a multitude of algorithms for deconvolution. The novel algorithmic framework Dsea is unique among these algorithms because it translates the deconvolution problem into a multinomial classification task, thus opening deconvolution to the field of machine learning and the rapid advances being made in that field. Section 2.1 presents the original version of Dsea, which is improved in the next chapter.

Classical deconvolution algorithms follow another approach than Dsea, solving a discrete reformulation of the continuous deconvolution problem instead of applying classification. This discrete deconvolution problem is introduced in Section 2.2, providing the basis for the Iterative Bayesian Unfolding and the Regularized Unfolding. These popular representatives of classical algorithms are presented in sections 2.3 and 2.4. Their results are compared to those of Dsea later in this thesis. Section 2.5 gives a rough outlook on some additional deconvolution algorithms, which are not further investigated here.

2.1 The Dortmund Spectrum Estimation Algorithm DSEA

Dsea reconstructs the density f of the target quantity Y from predictions which a classifier makes about this quantity. The following subsections present how these predictions are made and how they are used to reconstruct f. Dsea improves its estimate of f by iterating the reconstruction, updating the density of the training set that is used to obtain the classifier. The update is realized by weighting the individual training examples.

Dsea has some outstanding advantages over the classical deconvolution algorithms. Besides relating deconvolution to classification, it enables researchers to assess the contribution of each individual observation to the deconvolution result. For example, this feature of the method makes deconvolution in a sliding window over a time frame possible. In Cherenkov astronomy, such time-dependent deconvolution is indispensable for astronomers to reason about γ ray sources that change their emission over time.

2.1.1 Deconvolution as a Classification Task

Classification is the task of mapping observations $\mathbf{x} \in \mathcal{X}$ to a finite set of labels. Such mapping is inferred from the training set $\mathcal{D}_{\text{train}} = \{ (\mathbf{x}_n, y_n) \in \mathcal{X} \times \mathcal{Y} \ : \ 1 \leq n \leq N' \}$, in which the target value $y \in \mathcal{Y}$ is given for N' observations. It is then applied to the N unlabeled observations in $\mathcal{D}_{\text{obs}} = \{ \mathbf{x}_n \in \mathcal{X} \ : \ 1 \leq n \leq N \}$, for which a label is required. In order to translate the deconvolution problem into a classification task, a set of suitable labels has to be defined from the continuous state space $\mathcal{Y}$ of the target quantity.

A set of labels is generally obtained by a discretization function $d_f : \mathcal{Y} \to \{1, 2, \ldots, I\}$, which partitions the pre-image of the continuous target density $f : \mathcal{Y} \to \mathbb{R}$ into a set of I distinct sub-spaces $\mathcal{Y}_i \subseteq \mathcal{Y}$. These sub-spaces are then used as the labels of the classification task, so that a classification algorithm can learn to predict the sub-space an individual observation $\mathbf{x} \in \mathcal{D}_{\text{obs}}$ belongs to. For particle energy densities, $\mathcal{Y}$ is one-dimensional and its discretization reduces to a quantization of the target values. The function d_f is thus defined by $I+1$ strictly ordered interval boundaries $t_i \in \mathcal{Y}$, as indicated by Figure 2.1. It assigns each event $(\mathbf{x}, y) \in \mathcal{D}_{\text{train}}$ to the interval $\mathcal{Y}_i$ that $y \in \mathcal{Y}$ is in, so that a classification algorithm can learn to predict the interval that an individual observation belongs to.

$$\mathcal{Y}_i \ = \ [\, t_i, \, t_{i+1} [\qquad \forall \ 1 \leq i \leq I \tag{2.1}$$

In order to solve the deconvolution problem with a set of predictions given by a classifier, the target density f has to be reproduced from these predictions. Equation 2.2 presents the concept of such reproduction, estimating a discrete variant $\mathbf{f} = (\mathbf{f}_1, \ldots \mathbf{f}_I)$ of the continuous density f by the probability of each label $\mathcal{Y}_i$. Here, $Y \equiv i$ is short for $d_f(Y) = i$. The computation of $\mathbf{f}$ within this concept is given in the following subsection.

$$\mathbf{f}_i \ = \ \mathbb{P}\,(\, Y \equiv i \,) \ = \ \int_{\mathcal{Y}_i} f(y) \ \mathrm{d}\, y \qquad \forall \ 1 \leq i \leq I \tag{2.2}$$

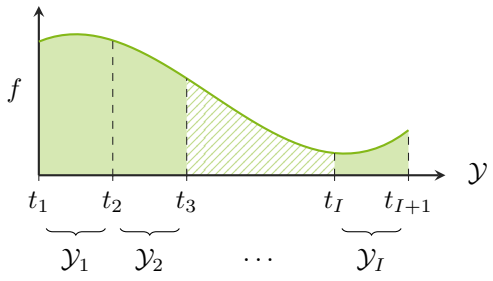


Figure 2.1: The continuous pre-image $\mathcal{Y}$ of the target density f is split into I distinct intervals $\mathcal{Y}_i \subseteq \mathcal{Y}$, which are used as the labels of the classification task. The density f is estimated from the density of these labels, which corresponds to a histogram of the target quantity Y.

All experiments in this thesis are conducted with equidistant interval boundaries $t_i = t_1 + \Delta \cdot (i-1) \;\; \forall \; i \geq 2$, so that each interval $\mathcal{Y}_i$ has the same width $\Delta \in \mathbb{R}$. These boundaries are favored here because a density $\mathbf{f}$ over the resulting labels directly forms an equidistant histogram of Y. Therefore, and since the classical deconvolution methods also estimate equidistant histograms, equidistant interval boundaries are well suited for comparing DSEA with these other methods.

2.1.2 Reproducing the Target Density from Confidence Values

Equation 2.3 computes an estimate of the probability $\mathbf{f}_i = \mathbb{P}(Y \equiv i)$. DSEA estimates the conditional probabilities $\hat{\mathbb{P}}(Y \equiv i \mid X = \mathbf{x})$ in this equation with confidence values $c_{\mathcal{M}}(i \mid \mathbf{x}_n)$ obtained from a trained classifier $\mathcal{M}$. Such value represents the confidence of $\mathcal{M}$ that the observation $\mathbf{x} \in \mathcal{D}_{\text{obs}}$ belongs to the label $\mathcal{Y}_i$. Moreover, $\hat{\mathbb{P}}(X = \mathbf{x}) = \frac{1}{N}$ $\forall\, \mathbf{x} \in \mathcal{D}_{\text{obs}}$ is imposed because each observation in $\mathcal{D}_{\text{obs}}$ occurs exactly once. Equation 2.4 applies these considerations to reproduce an estimated density $\hat{\mathbf{f}}$ from confidence values obtained for the observed data set $\mathcal{D}_{\text{obs}}$.

$$\hat{\mathbb{P}}(Y \equiv i) \;=\; \sum_{\mathbf{x} \in \mathcal{X}} \hat{\mathbb{P}}(Y \equiv i \mid X = \mathbf{x}) \cdot \hat{\mathbb{P}}(X = \mathbf{x}) \qquad \forall \; 1 \leq i \leq I \tag{2.3}$$

$$\hat{\mathbf{f}}_i \;=\; \frac{1}{N} \sum_{n=1}^{N} c_{\mathcal{M}}(i \mid \mathbf{x}_n) \tag{2.4}$$

Note that $\mathbf{f}$ could also be estimated with the relative frequencies $\hat{\mathbb{P}}(Y \equiv i) = \frac{\hat{N}_i}{N}$ of predictions, $\hat{N}_i$ being the number of observed events for which $\mathcal{Y}_i$ is the predicted label. However, this approach does not account for the uncertainty of the classifier, which is only expressed through the confidence values. Ignoring this uncertainty produces deconvolution results with inferior quality, compared to the reproduction from Equation 2.4 [3].

2.1.3 Classification Algorithms

The classification algorithm used to obtain the confidence values is basically arbitrary. However, the predictive quality of the classifier—and particularly the appropriateness of its confidence values—plays a crucial role in the quality of the estimated density.

Two exemplary classification algorithms are presented here, Naive Bayes and the Random Forest. The Naive Bayes classifier has a clear probabilistic meaning and it relates DSEA to the Iterative Bayesian Unfolding presented in Section 2.3. In fact, Subsection 2.3.4 proves that DSEA and the Iterative Bayesian Unfolding are equivalent under certain conditions. In Cherenkov astronomy, the Random Forest is more popular than the Naive Bayes classifier. Moreover, preliminary studies revealed that it also predicts the energy of γ particles more accurately than the Naive Bayes.

Naive Bayes

The Naive Bayes classifier is based on Bayes' theorem and the simplifying ("naive") assumption that all features are conditionally independent from each other [8]. Bayes' theorem is applied to model the posterior probabilities of the labels, given a particular observation. These probabilities are used as the confidence values, as presented in Equation 2.5. The Naive Bayes classifier predicts the label to which the highest confidence value is assigned.

$$\begin{aligned} c_{\mathcal{M}}(\, i \,|\, \mathbf{x}_n) &= \hat{\mathbb{P}}\,(\, Y \equiv i \mid X = \mathbf{x}_n) \\ &= \frac{\hat{\mathbb{P}}\,(\, X = \mathbf{x}_n \,|\, Y \equiv i\,) \cdot \hat{\mathbb{P}}\,(\, Y \equiv i\,)}{\sum_{i'=1}^{I} \hat{\mathbb{P}}\,(\, X = \mathbf{x}_n \,|\, Y \equiv i'\,) \cdot \hat{\mathbb{P}}\,(\, Y \equiv i'\,)} \end{aligned} \tag{2.5}$$

The prior probability $\hat{\mathbb{P}}\,(\, Y \equiv i\,)$ in Equation 2.5 is estimated by the relative frequency $\frac{N_i'}{N'}$ in the training set $\mathcal{D}_{\text{train}}$, where N_i' is the number of training examples with the label $\mathcal{Y}_i$. The conditional probability $\hat{\mathbb{P}}\,(\, X = \mathbf{x}_n \,|\, Y \equiv i\,)$ in this equation is where the naive independence assumption is applied. This assumption states that Equation 2.6 holds for multi-dimensional observations $\mathbf{x} \in \mathcal{X}$ consisting of D features. The conditional probability is thus given by the product of independent feature-wise probabilities.

$$\hat{\mathbb{P}}\,(\, X = \mathbf{x}_n \,|\, Y \equiv i\,) = \prod_{d=1}^{D} \hat{\mathbb{P}}\,(\, X_d = \mathbf{x}_{n;\,d} \,|\, Y \equiv i\,) \tag{2.6}$$

For a nominal feature with a finite set of discrete values, such conditional feature-wise probability is estimated by the relative frequencies in the training set, similar to estimating $\hat{\mathbb{P}}\,(\, Y \equiv i\,)$. In case of a numerical feature, the conditional probability has to be estimated by some continuous probability density, instead. A common simplification is to assume the Normal distribution $\mathcal{N} : \mathbb{R} \to \mathbb{R}$ presented in Equation 2.7. This distribution is characterized by its mean $\mu \in \mathbb{R}$ and its standard deviation $\sigma \in \mathbb{R}$, which are easily estimated for each combination of a label and a feature.

$$\mathcal{N}(\, x\,;\, \mu,\, \sigma\,) = \frac{1}{\sqrt{2\pi}\,\sigma}\, e^{-\frac{(x-\mu)^2}{2\sigma^2}} \tag{2.7}$$

Numerous improvements of the Naive Bayes classifier have been proposed, ranging from pre-processing steps which mitigate the independence assumption to extensions of the model that allow to represent other probability distributions. However, the basic version presented above is used here due to its simplicity. All experiments employ the ScikitLearn implementation further documented online[1].

[1] `http://scikit-learn.org/stable/modules/generated/sklearn.naive_bayes.GaussianNB.html`

Random Forest

The Random Forest is an ensemble method which aggregates the predictions from multiple randomized decision tree classifiers [9]. It is a popular tool in Cherenkov astronomy, being used for classification [10, 11] and for regression tasks [12, 13], in which it delivers outstanding predictive accuracies. The experiments conducted here use the implementation from ScikitLearn, which is described online[2] and in the following.

There are multiple methods randomizing the individual trees in the forest [9]. Here, the classical *bagging* approach [14] is employed, where each classifier $\mathcal{T} \in \mathcal{M}$ in the ensemble $\mathcal{M}$ is trained on a bootstrap sample of the training data. Each such sample is obtained by randomly drawing training examples with replacement. The predictions of the trained classifiers are then aggregated by voting. Namely, the aggregated prediction of the ensemble $\mathcal{M}$ is the label which is predicted by most of the ensemble members. Equation 2.8 presents the confidence of a bagging ensemble $\mathcal{M}$ for the label $\mathcal{Y}_i$, which is the relative number of votes $\mathcal{T}(\mathbf{x})$ for that label.

$$c_{\mathcal{M}}(\, i \,|\, \mathbf{x}) \;=\; \frac{\Big|\big\{\, \mathcal{T} \,\in\, \mathcal{M} \;:\; \mathcal{T}(\mathbf{x}) = i \,\big\}\Big|}{|\mathcal{M}|} \tag{2.8}$$

The individual decision trees in a Random Forest create an increasingly fine partition of the feature space with an increasing label purity in each fragment of that partition. For example, CART trees [15] measure the impurity of each such fragment with the *Gini Diversity Index* $\mathcal{G}$ defined in Equation 2.9. This index represents the probability of a random event with label $\mathcal{Y}_i$ being incorrectly labeled with a different $\mathcal{Y}_{i'}$ randomly assigned according to the label density $\mathbf{f}$ inside the respective fragment. Evaluating $\mathcal{G}$ for each possible split in the feature space allows to choose that partition which consists of maximally pure fragments. Usually, $\mathcal{G}$ is only evaluated for a random subset of possible splits in order to increase the randomness of the ensemble.

$$\mathcal{G}(\mathbf{f}) \;=\; \sum_{i=1}^{I} \mathbf{f}_i \cdot \sum_{i' \neq i} \mathbf{f}_{i'} \;=\; 1 - \sum_{i=1}^{I} \mathbf{f}_i^2 \tag{2.9}$$

2.1.4 The Iterative Procedure

The confidence values returned for observations in $\mathcal{D}_{\text{obs}}$ are determined by the density of labels in $\mathcal{D}_{\text{train}}$. For instance, the confidence of a Naive Bayes classifier explicitly incorporates an estimate $\hat{\mathbb{P}}\,(\, Y \,\equiv\, i \,)$ of that prior density. However, practitioners of deconvolution generally want the estimated target density $\hat{\mathbf{f}}$ to be independent from the prior density because the ultimate goal of deconvolution is to infer $\hat{\mathbf{f}}$ from observations, not from prior assumptions.

[2] http://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html

In order to mitigate the influence of the prior density inherently present in $\mathcal{D}_{\text{train}}$, DSEA iterates the reproduction of f by updating the training set density with the latest estimate of the density in $\mathcal{D}_{\text{obs}}$. Such update is performed by weighting each of the N' events in $\mathcal{D}_{\text{train}}$ with a corresponding weight scalar $w_n \in \mathbb{R}$. These weights are accounted for in the training of $\mathcal{M}$. Thus, they determine the confidence values subsequently obtained for the observations in $\mathcal{D}_{\text{obs}}$. Even though the iterative procedure improves DSEA's estimates in the first few iterations, it also decreases the quality of the deconvolution result after a suitable estimate has been found. This divergence from the optimal solution is already presented in Section 1.2, motivating the improvements proposed later in this thesis.

Algorithm 2.1 presents DSEA in its current state. Given are a training set $\mathcal{D}_{\text{train}}$ to infer the classifier from and a set of unlabeled observations $\mathcal{D}_{\text{obs}}$, for which the target density f is estimated with the discrete density $\hat{\mathbf{f}}$ returned by DSEA. Moreover, the total number of iterations K is pre-specified, usually being chosen from test deconvolutions on simulated data. Originally, a convergence criterion based on the χ^2 distance between iterations has been proposed [4], but this criterion is rarely used in practice. The prior $\hat{\mathbf{f}}^{(0)}$ of the target density can also be specified by the practitioner. By default, it weights the training set uniformly, retaining the inherent density of $\mathcal{D}_{\text{train}}$. Line 2 of Algorithm 2.1 performs the weight update of the current iteration. In line 4, the current estimate of the target density is reproduced from confidence values.

Input:

Observed data set $\mathcal{D}_{\text{obs}} = \{\, \mathbf{x}_n \in \mathcal{X} \;:\; 1 \leq n \leq N \,\}$

Training data set $\mathcal{D}_{\text{train}} = \{\, (\mathbf{x}_n,\, y_n) \in \mathcal{X} \times \{1, 2, \ldots, I\} \;:\; 1 \leq n \leq N' \,\}$

Number of iterations K

Prior density $\hat{\mathbf{f}}^{(0)}$ (default: $\hat{\mathbf{f}}_i^{(0)} = \frac{1}{I} \;\; \forall\; 1 \leq i \leq I$)

Output:

Estimated target density $\hat{\mathbf{f}} = \hat{\mathbf{f}}^{(K)}$

1: **for** $k \in \{1, 2, \ldots, K\}$ **do**

2: $\quad \forall\; 1 \leq n \leq N' : \;\; w_n^{(k)} \leftarrow \hat{\mathbf{f}}_{i(n)}^{(k-1)}$

3: $\quad$ Infer $\mathcal{M}$ from $\mathcal{D}_{\text{train}}$ weighted by $w_n^{(k)}$

4: $\quad \forall\; 1 \leq i \leq I : \quad \hat{\mathbf{f}}_i^{(k)} \leftarrow \frac{1}{N} \sum_{n=1}^{N} c_{\mathcal{M}}(\, i \,|\, \mathbf{x}_n)$

Algorithm 2.1: The Dortmund Spectrum Estimation Algorithm DSEA [4]. Here, the convergence check based on the χ^2 distance between iterations is omitted due to its rare application in practice. Instead, the total number K of iterations is pre-specified. The classifier $\mathcal{M}$ is inferred by a machine learning algorithm plugged into the method.

2.2 The Classical Discrete Deconvolution Problem

Classical methods like the Iterative Bayesian Unfolding and the Regularized Unfolding solve a discrete variant of the continuous deconvolution problem from Equation 1.1, instead of applying classification. This discrete variant presented in Equation 2.10 is a linear system of equations where f, g, and R from the continuous problem are replaced by their discrete versions. The relation between the discrete and continuous case is clarified by Equation 2.11, which equivalently presents the discrete problem for the individual components of the observed density $\mathbf{g} = (\mathbf{g}_1, \dots \mathbf{g}_J) \in \mathbb{R}^J$. Classical deconvolution methods differ from DSEA in that they estimate $\mathbf{f} = (\mathbf{f}_1, \dots \mathbf{f}_I) \in \mathbb{R}^I$ by solving Equation 2.10 instead of reproducing $\mathbf{f}$ from predictions.

$$\mathbf{g} = \mathbf{R}\,\mathbf{f} \tag{2.10}$$

$$\Leftrightarrow \qquad \mathbf{g}_j = \sum_{i=1}^{I} \mathbf{R}_{ij}\,\mathbf{f}_i \qquad \forall\ 1 \leq j \leq J \tag{2.11}$$

The vector $\mathbf{f}$ usually represents a histogram of the continuous target density f, being defined by $I + 1$ equidistant interval boundaries $t_i \in \mathcal{Y}$. Such histogram is introduced already in Subsection 2.1.1, but it should be noted that the intervals $\mathcal{Y}_i \subseteq \mathcal{Y}$ of that histogram are not used as labels here. Namely, classical deconvolution methods do not predict the intervals of individual observations. The Regularized Unfolding also supports other discretization schemes for f, of which the most prominent one presented in Section A.2 is based on B-spline coefficients. However, the most recent publication on the method abandoned such other schemes in favor of equidistant histograms [2]. The discretization of the detector response function R and the observed density g are presented in the following. Subsequently, the difficulty of solving the discrete deconvolution problem is reflected.

2.2.1 The Detector Response Matrix

The detector response function $R : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$ embodies the practitioner's knowledge about the detector. However, that knowledge does often not suffice to obtain such continuous function analytically. For example, it is not feasible to analytically obtain R for IACTs, which involve atmospheric processes that are highly random. Instead, a discrete version $\mathbf{R} \in \mathbb{R}^{I \times J}$ of R is estimated from the training set $\mathcal{D}_{\text{train}}$.

The structure of $\mathbf{R}$ is presented in Equation 2.12. Each component $\mathbf{R}_{i\,j}$ of that matrix, conceptually defined in Equation 2.13, represents the conditional probability of observing an event in the sub-space $\mathcal{X}_j \subseteq \mathcal{X}$ when the value of the relevant quantity Y is in the sub-space $\mathcal{Y}_i \subseteq \mathcal{Y}$. For IACTs, this could be the probability of observing an air shower of a specific size or orientation when the initial γ particle of that shower is in a particular energy interval. The definition of the observable sub-spaces $\mathcal{X}_j \subseteq \mathcal{X}$ is given in the following subsection.

$$\mathbf{R} \;=\; \begin{pmatrix} \mathbf{R}_{1\,1} & \dots & \mathbf{R}_{I\,1} \\ \vdots & \ddots & \\ \mathbf{R}_{1\,J} & & \mathbf{R}_{I\,J} \end{pmatrix} \tag{2.12}$$

$$\begin{aligned} \mathbf{R}_{i\,j} \;&=\; \mathbb{P}\,(\,X \equiv j \,|\, Y \equiv i\,) \\ &=\; \int_{\mathcal{Y}_i} \int_{\mathcal{X}_j} R(x \,|\, y)\;\mathrm{d}\,x\;\mathrm{d}\,y \qquad \forall\; 1 \leq i \leq I,\;\; 1 \leq j \leq J \end{aligned} \tag{2.13}$$

The matrix $\mathbf{R}$ is estimated from $\mathcal{D}_{\text{train}}$, in which the values $\mathbf{x} \in \mathcal{X}$ and $y \in \mathcal{Y}$ of each individual observation are known. Equation 2.14 computes the components of such estimate, where N_{ij} is the number of events $(\mathbf{x},\, y) \in \mathcal{D}_{\text{train}}$ for which $\mathbf{x} \in \mathcal{X}_j$ and $y \in \mathcal{Y}_i$.

$$\hat{\mathbf{R}}_{i\,j} \;=\; \frac{N_{ij}}{\sum_{j'=1}^{J} N_{ij'}} \qquad \forall\; 1 \leq i \leq I, \quad 1 \leq j \leq J \tag{2.14}$$

2.2.2 Clustered Observable Quantities

A one-dimensional discrete version $\mathbf{g} = (\mathbf{g}_1, \dots \mathbf{g}_J) \in \mathbb{R}^J$ of the observed density function $g : \mathcal{X} \to \mathbb{R}$ is obtained by partitioning the continuous pre-image $\mathcal{X}$ of g into a set of J distinct sub-spaces $\mathcal{X}_j \subseteq \mathcal{X}$. Each $\mathbf{g}_j$ is then defined similar to the $\mathbf{f}_i$ in Equation 2.2, being estimated by its relative frequency $\hat{\mathbf{g}}_j = \frac{N_j}{N}$ in the observed data set $\mathcal{D}_{\text{obs}}$. Here, N_j refers to the number of observed examples in the respective sub-space.

However, the domain $\mathcal{X}$ of g is usually multidimensional—unlike the domain of the target density f discretized in Subsection 2.1.1. For IACTs, $\mathcal{X}$ comprises multiple geometric features of air showers, like their shape and their size. Therefore $\mathcal{X}$ needs to be clustered in order to obtain a finite set of multidimensional sub-spaces. A Clustering $d_g : \mathcal{X} \to \{1, 2, \dots, J\}$ maps each vector-valued observation $\mathbf{x} \in \mathcal{X}$ to the index j of the sub-space which the observation belongs to, effectively discretizing the multi-dimensional feature space in a single dimension. Equation 2.15 and Equation 2.16 define the properties of such clustering, namely distinct clusters and completeness.

$$\mathcal{X}_j \cup \mathcal{X}_{j'} \;=\; \emptyset \qquad \forall\; j \neq j' \tag{2.15}$$

$$\bigcup_{j=1}^{J} \mathcal{X}_j \;=\; \mathcal{X} \tag{2.16}$$

A suitable clustering of $\mathcal{X}$ can reduce the condition number of the detector response matrix $\mathbf{R}$, which indicates the difficulty of the discrete deconvolution problem. In other words, a suitable clustering facilitates the reconstruction of $\mathbf{f}$. This issue is presented in the following subsection, motivating a supervised clustering approach.

2.2.3 The Difficulty of Classical Deconvolution

The first problem faced in solving the discrete deconvolution problem from Equation 2.10 is that the detector response matrix $\mathbf{R}$ can usually not be inverted, what would provide the most straightforward estimate $\hat{\mathbf{f}}_{\text{unfeasible}} = \mathbf{R}^{-1}\mathbf{g}$ of the target density. This issue is circumvented by the naive estimator presented in Equation 2.17, replacing $\mathbf{R}^{-1}$ with the Moore-Penrose pseudo-inverse $\mathbf{R}^{\dagger}$ defined in Equation 2.18. The advantage of $\mathbf{R}^{\dagger}$ over $\mathbf{R}^{-1}$ is that it always exists—even for rectangular matrices. It is based on the singular value decomposition of $\mathbf{R} = \mathbf{U}\,\mathbf{S}\,\mathbf{V}^T$, where $\mathbf{S} = \text{diag}(\sigma_1, \dots \sigma_I) \in \mathbb{R}^{I \times J}$ is a diagonal matrix consisting of the singular values of $\mathbf{R}$ in decreasing order. Let r be the index of the last non-zero singular value in $\mathbf{S}$. The naive estimator corresponds to a minimum-norm least-squares solution of the discrete deconvolution problem [16]. It is naive because it exhibits a huge variance, as we will see.

$$\hat{\mathbf{f}}_{\text{naive}} = \mathbf{R}^{\dagger}\,\mathbf{g} \tag{2.17}$$

$$\mathbf{R}^{\dagger} = \mathbf{V}\,\mathbf{S}^{\dagger}\,\mathbf{U}^T \quad \text{where } \mathbf{S}^{\dagger} = \text{diag}\left(\sigma_1^{-1}, \dots \sigma_r^{-1}, 0, \dots 0\right) \tag{2.18}$$

The naive estimator effectively transforms the original deconvolution problem to the surrogate problem presented in Equation 2.19. Equation 2.20 then obtains the solution of the original problem from the surrogate solution $\hat{\mathbf{f}}'$. This view on the naive estimator is employed here to trace back the difficulty of deconvolution to the condition of $\mathbf{R}$.

$$\begin{aligned} \mathbf{g}' &= \mathbf{S}\,\mathbf{f}' \\ \text{where} \quad \mathbf{g}' &= \mathbf{U}^T\,\mathbf{g} \quad \text{and} \quad \mathbf{f}' = \mathbf{V}^T\,\mathbf{f} \end{aligned} \tag{2.19}$$

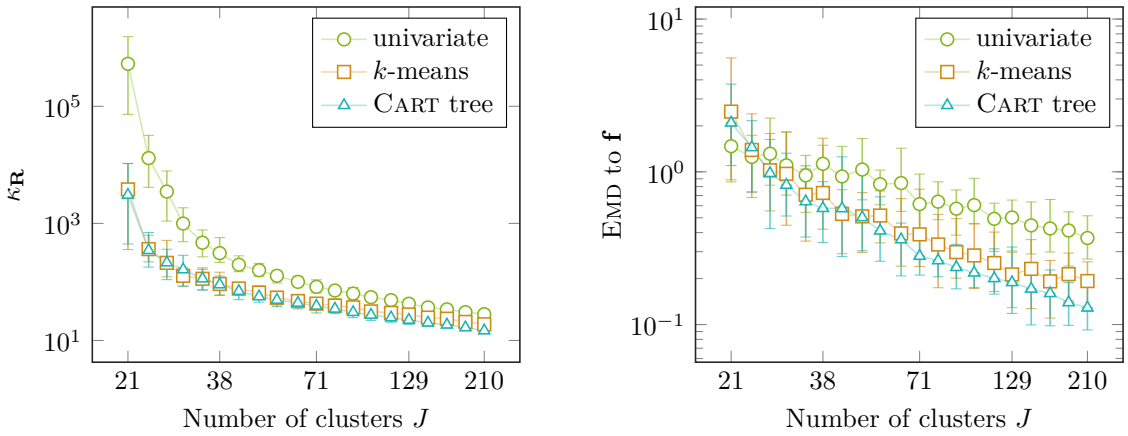
$$\hat{\mathbf{f}}_{\text{naive}} = \mathbf{V}\,\hat{\mathbf{f}}' \quad \text{where} \quad \hat{\mathbf{f}}' = \mathbf{S}^{\dagger}\mathbf{g}' \tag{2.20}$$

Imagine the last singular values of $\mathbf{R}$ to be extraordinarily small compared to the first singular values. The inverse of these last values is accordingly large and their respective components in $\mathbf{g}'$ consequently contribute a lot to $\hat{\mathbf{f}}'$ (see Equation 2.20). Small changes in these components thus result in huge changes in the estimated target density. In other words, the naive estimator has an immense variance, when the singular values of $\mathbf{R}$ vary. The difficulty of deconvolution is thus given by the condition number $\kappa_{\mathbf{R}}$ from Equation 2.21, which expresses the range spanned by the singular values of $\mathbf{R}$. Classical deconvolution methods aim at reducing the variance of their estimates by introducing a small bias, which is an approach referred to as regularization. A high condition number of $\mathbf{R}$ requires a strong regularization, which produces highly biased results.

$$\kappa_{\mathbf{R}} = \frac{\sigma_1}{\sigma_I} \tag{2.21}$$

Figure 2.2a shows that the condition number decreases with an increasing number of clusters used to obtain $\mathbf{g}$ on a toy data set. The naive estimator accordingly produces increasingly good results, which can be seen in Figure 2.2b. A suitable clustering of $\mathcal{X}$ thus facilitates deconvolution in classical methods.

Three different approaches to clustering are applied in this experiment, the first one using only that single feature which is most strongly correlated with the target variable Y. The second approach is k-means, a classical unsupervised method for clustering. The third approach trains a CART decision tree classifier [15] (also see Section 2.1.3) and it interprets the leaves of such tree as clusters. This method, already being proposed for Cherenkov astronomy [17], thus relates to supervised clustering, a problem recently described [18]. Further research down this lane promises a reduction of the inherent complexity of the discrete deconvolution problem. The present thesis focuses on the CART discretization, which produces the best results so far.



(a) The difficulty of the discrete problem is indicated by the condition number $\kappa_{\mathbf{R}}$ of the detector response matrix $\mathbf{R} \in \mathbb{R}^{I \times J}$. This number decreases with an increasing number J of clusters. k-means and the CART discretization produce less difficult problems than a univariate equidistant quantization of the single most correlated feature.

(b) The distance between the true target density $\mathbf{f}$ and the estimate $\hat{\mathbf{f}}$ obtained by the naive estimator gets smaller as more clusters are used to obtain $\mathbf{g} \in \mathbb{R}^J$. This observation is consistent with the decreasing difficulty of the problem, which is already indicated by the left subfigure. Low values of the distance metric EMD from Section 4.1 indicate a high quality of $\hat{\mathbf{f}}$.

Figure 2.2: The difficulty of the discrete deconvolution problem decreases with an increasing number J of clusters used to obtain the discrete observed density $\mathbf{g} \in \mathbb{R}^J$. The values of J lie on a logarithmic scale here.

2.3 Iterative Bayesian Unfolding

The Iterative Bayesian Unfolding (IBU) [6, 7] is a method solving the classical deconvolution problem from Section 2.2. It obtains an estimated target density $\hat{\mathbf{f}}$ by applying Bayes' theorem to the observed density $\mathbf{g}$ and the conditional probabilities embodied by the detector response matrix $\mathbf{R}$. Bayes' theorem is applied multiple times, each time updating the prior density with the respective latest estimate. Repeating this update reduces the influence of the initial prior density, what is desired by practitioners, who usually want to infer $\hat{\mathbf{f}}$ from observations and not from prior assumptions.

Basically, the components of the estimated target density $\hat{\mathbf{f}}$ are given by Equation 2.22, where $\hat{\mathbb{P}}(Y \equiv i \mid X \equiv j)$ is an estimate of the conditional probability of the target value being in the sub-space $\mathcal{Y}_i$ when the measured value is in the sub-space $\mathcal{X}_j$. This estimated conditional probability is obtained from Bayes' theorem, which is presented in Equation 2.23. Note that each probability $\hat{\mathbb{P}}(X \equiv j \mid Y \equiv i)$ of the other conditional direction is already given by the corresponding component $\mathbf{R}_{ij}$ of the detector response matrix. The probabilities $\hat{\mathbb{P}}(Y \equiv i)$ of the intervals in $\mathcal{Y}$ represent a prior assumption about the target density $\mathbf{f}$.

$$\hat{\mathbf{f}}_i = \sum_{j=1}^{J} \hat{\mathbb{P}}(Y \equiv i \mid X \equiv j) \cdot \mathbf{g}_j \tag{2.22}$$

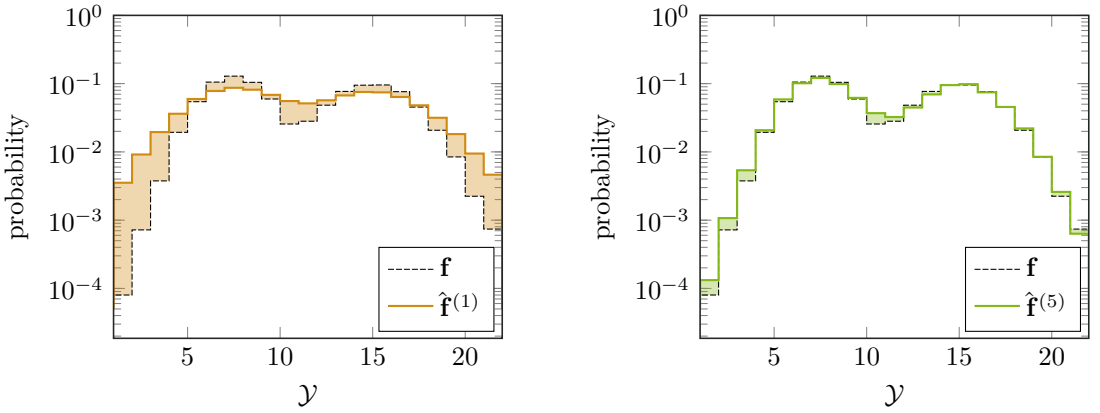
$$\hat{\mathbb{P}}(Y \equiv i \mid X \equiv j) = \frac{\hat{\mathbb{P}}(X \equiv j \mid Y \equiv i) \cdot \hat{\mathbb{P}}(Y \equiv i)}{\sum_{i'=1}^{I} \hat{\mathbb{P}}(X \equiv j \mid Y \equiv i) \cdot \hat{\mathbb{P}}(Y \equiv i)} \tag{2.23}$$

2.3.1 Iterative Reduction of the Prior's Influence

Since the goal of deconvolution is to estimate $\mathbf{f}$ from observations, it is generally assumed that the prior $\hat{\mathbb{P}}(Y \equiv i)$ is not sufficiently accurate. It is thus desirable to reduce the strong influence of $\hat{\mathbb{P}}(Y \equiv i)$ on the deconvolution result. IBU achieves such reduction by repeating the reconstruction, each time replacing the prior with the estimate $\hat{\mathbf{f}}^{(k-1)}$ from the previous iteration. Equation 2.24 summarizes this update, bringing together Equations 2.22 and 2.23. The initial prior $\hat{\mathbf{f}}^{(0)}$ is specified by the practitioner.

$$\hat{\mathbf{f}}_i^{(k)} = \sum_{j=1}^{J} \frac{R_{ij}\, \hat{\mathbf{f}}_i^{(k-1)}}{\sum_{i'=1}^{I} R_{i'j}\, \hat{\mathbf{f}}_{i'}^{(k-1)}} \cdot \mathbf{g}_j \tag{2.24}$$

An example deconvolution with an inappropriate prior is visualized in Figure 2.3. The initial prior $\hat{\mathbf{f}}^{(0)}$ of this deconvolution is uniform, obviously not matching the true target density $\mathbf{f}$ presented in dashed lines. Applying Bayes' theorem a single time maintains $\hat{\mathbf{f}}^{(0)}$ to some degree, yielding the rather flat estimate from Figure 2.3a. Repeating the reconstruction according to Equation 2.24 produces a more accurate estimate of $\mathbf{f}$.



(a) A single iteration of the algorithm retains the uniform prior $\hat{\mathbf{f}}^{(0)}$ to some degree.

(b) The true density $\mathbf{f}$ is more tightly resembled after five iterations.

Figure 2.3: IBU reduces the influence of the initial prior $\hat{\mathbf{f}}^{(0)}$ by applying Bayes' theorem multiple times. In this deconvolution on the toy data from Subsection 4.2.1, a uniform prior was used. This prior does not reflect the double-peak structure of the true density $\mathbf{f}$.

2.3.2 Smoothing and Early Stopping

Publications on IBU report that the deconvolution result is unsatisfactory if the prior is updated too often [6, 7]. This behavior is similar to the divergence observed with DSEA. IBU mitigates this divergence by smoothing the respective latest estimate $\hat{\mathbf{f}}^{(k-1)}$ before using it as a prior in the k-th iteration. Such smoothing is performed by fitting some analytic form to $\hat{\mathbf{f}}^{(k-1)}$. This form is then used as a prior, replacing the actual $\hat{\mathbf{f}}^{(k-1)}$. Second-order polynomials are presented as a suitable default for such analytic form. However, it is noted that other forms may be more appropriate for the specific scenario at hand. It should also be noted that the smoothing is neither performed on the initial prior $\hat{\mathbf{f}}^{(0)}$, nor on the last estimate. Since the operation biases the algorithm towards smooth solutions, it is referred to as a kind of regularization.

Another kind of regularization feasible in IBU is early stopping. However, publications on the IBU recommend to stop only when the difference between two subsequent estimates $\hat{\mathbf{f}}^{(k)}$ and $\hat{\mathbf{f}}^{(k-1)}$ is sufficiently small, i.e. when no regularizing effect results from stopping the iterative procedure. The change between estimates is measured by a χ^2 distance and the algorithm stops when this distance is smaller than some $\epsilon > 0$. Unfortunately, the given reference [6, 7] does not state which particular χ^2 distance to use, even though several of them exist. In our experiments, we use the χ^2_{Sym} distance from Equation 2.25. Subsection C.3.3 provides more information on this distance measure.

$$\chi^2_{\text{Sym}}(\hat{\mathbf{f}}, \mathbf{f}) \;=\; 2 \cdot \sum_{i=1}^{I} \frac{\left(\hat{\mathbf{f}}_i - \mathbf{f}_i\right)^2}{\hat{\mathbf{f}}_i + \mathbf{f}_i} \tag{2.25}$$

2.3.3 The Algorithm

IBU is specified in Algorithm 2.2. It produces a discrete estimate $\hat{\mathbf{f}}$ of the target density, given the observed density $\mathbf{g}$ and the detector response matrix $\mathbf{R}$. An initial prior $\hat{\mathbf{f}}^{(0)}$ can also be given, being uniform by default. The smoothing can be omitted or replaced by another smoothing operation that employs a more appropriate analytic form.

Input:

Discrete observed density $\mathbf{g} \in \mathbb{R}^J$

Detector response matrix $\mathbf{R} \in \mathbb{R}^{I \times J}$

Discrete prior density $\hat{\mathbf{f}}^{(0)}$ (default: $\hat{\mathbf{f}}_i^{(0)} = \frac{1}{I} \;\; \forall\; 1 \leq i \leq I$)

$\epsilon > 0$, the minimal χ^2_{Sym} distance between subsequent iterations

Output:

Estimated discrete target density $\hat{\mathbf{f}} \in \mathbb{R}^I$

1: $k \leftarrow 0$

2: **repeat**

3: $\quad k \leftarrow k + 1$

4: $\quad$ **if** $k > 1$ **then** $\hat{\mathbf{f}}^{(k-1)} \leftarrow$ low-order polynomial fit of $\hat{\mathbf{f}}^{(k-1)}$ **(optional)**

5: $\quad \forall\, 1 \leq i \leq I: \quad \hat{\mathbf{f}}_i^{(k)} \leftarrow \sum_{j=1}^{J} \frac{\mathbf{R}_{ij}\, \hat{\mathbf{f}}_i^{(k-1)}}{\sum_{i'=1}^{I} \mathbf{R}_{ij}\, \hat{\mathbf{f}}_{i'}^{(k-1)}} \cdot \mathbf{g}_j$

6: **until** $\chi^2_{\mathrm{Sym}}\big(\hat{\mathbf{f}}^{(k)}, \hat{\mathbf{f}}^{(k-1)}\big) \leq \epsilon$

7: **return** $\hat{\mathbf{f}} \leftarrow \hat{\mathbf{f}}^{(k)}$

Algorithm 2.2: The Iterative Bayesian Unfolding [6] without acceptance correction. Here, the probabilistic symmetric χ^2_{Sym} distance from Equation 2.25 defines the stopping criterion. The smoothing in line 4 can be omitted or be replaced by fitting $\hat{\mathbf{f}}^{(k-1)}$ to another analytic form.

2.3.4 Relation between IBU and DSEA

When DSEA employs a Naive Bayes classifier trained on a single nominal attribute, it is equivalent to IBU. Since DSEA is open to other classification algorithms and arbitrary mixtures of multiple nominal and numerical attributes, it is therefore more general than the Iterative Bayesian Unfolding. To see this relation between the two algorithms, let us recap the reconstruction rule of DSEA previously presented in Equations 2.4 and 2.5.

$$\hat{\mathbf{f}}_i^{\mathrm{DSEA}} = \frac{1}{N} \sum_{n=1}^{N} c_{\mathcal{M}}(\, i \,|\, \mathbf{x}_n)$$

$$c_{\mathcal{M}}(\, i \,|\, \mathbf{x}_n) = \frac{\hat{\mathbb{P}}(\, X = \mathbf{x}_n \,|\, Y \equiv i \,) \cdot \hat{\mathbb{P}}(\, Y \equiv i \,)}{\sum_{i'=1}^{I} \hat{\mathbb{P}}(\, X = \mathbf{x}_n \,|\, Y \equiv i' \,) \cdot \hat{\mathbb{P}}(\, Y \equiv i' \,)}$$

Let X consist of a single nominal attribute with J discrete values. Without loss of generality, this means $\mathcal{X} = \{1, 2, \ldots, J\}$. Moreover, let $\mathcal{D}_j = \{\, \mathbf{x} \in \mathcal{D}_{\text{obs}} \mid \mathbf{x} = j \,\}$ be the set of observed examples with the attribute value j. Grouping the N observations $\mathbf{x}_n \in \mathcal{D}_{\text{obs}}$ by their attribute values corresponds to a mere re-ordering of the observations.

$$\begin{aligned}\hat{\mathbf{f}}_i^{\text{DSEA}} &= \frac{1}{N} \sum_{n=1}^{N} \frac{\hat{\mathbb{P}}(X = \mathbf{x}_n \mid Y \equiv i) \cdot \hat{\mathbb{P}}(Y \equiv i)}{\sum_{i'=1}^{I} \hat{\mathbb{P}}(X = \mathbf{x}_n \mid Y \equiv i') \cdot \hat{\mathbb{P}}(Y \equiv i')} \\ &= \frac{1}{N} \sum_{j=1}^{J} \sum_{\mathbf{x} \in \mathcal{D}_j} \frac{\hat{\mathbb{P}}(X = \mathbf{x} \mid Y \equiv i) \cdot \hat{\mathbb{P}}(Y \equiv i)}{\sum_{i'=1}^{I} \hat{\mathbb{P}}(X = \mathbf{x} \mid Y \equiv i') \cdot \hat{\mathbb{P}}(Y \equiv i')}\end{aligned}$$

In this form, all terms in the inner sum are equal to each other because all $\mathbf{x} \in \mathcal{D}_j$ have the same value. Since this value is j, $X = \mathbf{x}$ can be written as $X \equiv j$. Moreover, let N_j be the number of examples in $\mathcal{D}_j$. The nested sum can then be simplified.

$$\hat{\mathbf{f}}_i^{\text{DSEA}} = \frac{1}{N} \sum_{j=1}^{J} N_j \cdot \frac{\hat{\mathbb{P}}(X \equiv j \mid Y \equiv i) \cdot \hat{\mathbb{P}}(Y \equiv i)}{\sum_{i'=1}^{I} \hat{\mathbb{P}}(X \equiv j \mid Y \equiv i') \cdot \hat{\mathbb{P}}(Y \equiv i')}$$

Recall that IBU estimates $\mathbf{g}_j$ from the relative number $\frac{N_j}{N}$. Moreover, each conditional probability $\hat{\mathbb{P}}(X \equiv j \mid Y \equiv i)$ is given by the corresponding component $\mathbf{R}_{ij}$ of the detector response matrix. $\hat{\mathbb{P}}(Y \equiv i)$ is a component of the prior on $\mathbf{f}$ also written as $\hat{\mathbf{f}}_i^{(0)}$. Rewriting the DSEA update rule under these considerations yields the update rule of IBU from Equation 2.24. DSEA is thus equivalent to IBU, if a single nominal feature is used to train a Naive Bayes classifier.

$$\hat{\mathbf{f}}_i^{\text{DSEA}} = \sum_{j=1}^{J} \frac{R_{ij}\, \hat{\mathbf{f}}_i^{(0)}}{\sum_{i'=1}^{I} R_{i'j}\, \hat{\mathbf{f}}_{i'}^{(0)}} \cdot \mathbf{g}_j$$

This equivalence does not hold if multiple features or a single numerical feature are used in DSEA because the probabilities are computed differently, then. Moreover, the algorithms are not equivalent if another classifier than Naive Bayes is employed. In the sense that DSEA is able to use multiple features to train an arbitrary classifier, DSEA is thus more general than the Iterative Bayesian Unfolding. This relation is first given here, not being published before.

2.4 Regularized Unfolding

The Regularized Unfolding (RUN) [5, 2] is another method solving the discrete deconvolution problem from Section 2.2. It basically performs a maximum likelihood fit to estimate the target density, assuming that the absolute number $\bar{\mathbf{g}}_j = N \cdot \mathbf{g}_j$ of events is Poisson-distributed in each observed sub-space $\mathcal{X}_j$. Regularization is employed in order to produce sufficiently stable results in spite of the large variance found in un-regularized solutions. Subsection 2.2.3 already presented this variance as the difficulty of classical deconvolution.

In RUN, the practitioner controls the amount of regularization by the number of degrees of freedom n_{df} of the solution, what results in a crucial difference between the algorithm and the standard maximum likelihood method. In order to achieve a fixed n_{df}, one has to adapt the actual regularization strength τ—a factor in the objective function—in each iteration of the algorithm. Usually, this factor has a fixed value, instead.

2.4.1 Likelihood Maximization

The likelihood $\mathbb{L}$ of a discrete target density $\mathbf{f}$, given the outcome $\mathbf{g}$ of the measurement, is defined as the conditional probability of $\mathbf{g}$ given $\mathbf{f}$. Maximizing $\mathbb{L}$ therefore corresponds to finding the $\mathbf{f}$ under which the observed density $\mathbf{g}$ is most likely. Such estimate $\hat{\mathbf{f}}_{\max \mathbb{L}}$ is presented in Equation 2.26. RUN makes two simplifying assumptions about $\mathbb{L}$ to find such estimate by equivalently minimizing the loss function from Theorem 2.1.

$$\hat{\mathbf{f}}_{\max \mathbb{L}} \;=\; \arg\max_{\mathbf{f}} \; \mathbb{L}(\mathbf{f} \,|\, \mathbf{g}) \qquad \text{where} \qquad \mathbb{L}\,(\mathbf{f} \,|\, \mathbf{g}) \;=\; \mathbb{P}\,(\mathbf{g} \,|\, \mathbf{f}) \tag{2.26}$$

Theorem 2.1 *The minimizer of the loss function $\ell : \mathbb{R}^I \to \mathbb{R}$ maximizes the likelihood $\mathbb{L}$ under the following two assumptions.*

$$\ell\,(\mathbf{f}) \;=\; \sum_j \mathbf{R}_{\bullet j}^T \bar{\mathbf{f}} \;-\; \bar{\mathbf{g}}_j \ln\left(\mathbf{R}_{\bullet j}^T \bar{\mathbf{f}}\,\right)$$

Assumption 1 The components $\mathbf{g}_j$ of the observed density are statistically independent from each other, i.e. $\mathbb{P}(\mathbf{g} \,|\, \mathbf{f}) \;=\; \prod_{j=1}^{J} \mathbb{P}(\mathbf{g}_j |\, \mathbf{f})$.

Assumption 2 The absolute number $\bar{\mathbf{g}}_j = N \cdot \mathbf{g}_j$ of observations is Poisson-distributed in each observed sub-space $\mathcal{X}_j \subseteq \mathcal{X}$. A Poisson distribution $\mathcal{P}_\lambda$ models the probability of observing such absolute number r in a fixed time interval, in which the average rate of observing an event is λ. This rate is respectively given by $\lambda_j = \mathbf{R}_{\bullet j}^T \bar{\mathbf{f}}$, where $\bar{\mathbf{f}} = N \cdot \mathbf{f}$ is the frequency distribution of the target quantity. The following equation formalizes this assumption.

$$\mathbb{P}(\mathbf{g}_j |\, \mathbf{f}) \;=\; \mathcal{P}_{(\mathbf{R}_{\bullet j}^T \bar{\mathbf{f}})}(\bar{\mathbf{g}}_j) \qquad \text{where} \qquad \mathcal{P}_\lambda(r) \;=\; e^{-\lambda}\,\frac{\lambda^r}{r!} \qquad r \in \mathbb{N}_0$$

Proof. ℓ is obtained by plugging the two assumptions into Equation 2.26 while omitting terms that are constant with respect to $\mathbf{f}$. The full proof is given in Section A.3.

A minimizer of ℓ is found with the Newton method from numerical optimization [19], which evaluates a local quadratic model $\hat{\ell}^{(k)}$ of ℓ in each iteration k. The minimizer of each local model is chosen as the next estimate $\hat{\mathbf{f}}^{(k+1)}$, which eventually reaches a (local) minimizer of the actual function ℓ. Equation 2.27 presents such local model where $\boldsymbol{\ell} = \nabla \ell \left(\hat{\mathbf{f}}^{(k)}\right) \in \mathbb{R}^I$ is the gradient and $\mathbf{H} = \nabla^2 \ell \left(\hat{\mathbf{f}}^{(k)}\right) \in \mathbb{R}^{I \times I}$ is the Hessian of ℓ at the latest estimate $\hat{\mathbf{f}}^{(k)}$. RUN works similar to the Newton method, minimizing the regularized loss function presented in the following section.

$$\hat{\ell}^{(k)}(\mathbf{f}) \;=\; \frac{1}{2}\,\mathbf{f}^T\mathbf{H}\,\mathbf{f} \;-\; \mathbf{f}^T(\mathbf{H}\,\hat{\mathbf{f}}^{(k)} - \boldsymbol{\ell}) \tag{2.27}$$

2.4.2 Regularization

To regularize the minimizer of ℓ means to minimize the regularized loss function $\ell_r(\mathbf{f}) = \ell(\mathbf{f}) + r(\mathbf{f})$ instead of ℓ, thus reducing the variance of the estimate by biasing it towards smooth solutions. Equation 2.29 defines the regularization term $r(\mathbf{f})$, being conveniently re-written as a matrix product which models the non-smoothness of $\mathbf{f}$. The regularization strength $\tau \in \mathbb{R}$ is a meta-parameter usually chosen by the practitioner.

$$\hat{\ell}_r^{(k)}(\mathbf{f}) \;=\; \hat{\ell}^{(k)}(\mathbf{f}) \;+\; r(\mathbf{f}) \tag{2.28}$$

$$r(\mathbf{f}) \;=\; \frac{\tau}{2} \cdot \sum_{i=2}^{I-1} (-\,\mathbf{f}_{i-1} \;+\; 2 \cdot \mathbf{f}_i \;-\; \mathbf{f}_{i+1})^2 \;=\; \frac{\tau}{2} \cdot \mathbf{f}^T\mathbf{C}\,\mathbf{f} \tag{2.29}$$

Up to this point, RUN performs a standard maximum likelihood fit with usual regularization. However, it does not fix the value of τ. Instead, the practitioner chooses an effective number of degrees of freedom n_{df} of the solution, which is then forced upon the estimate $\hat{\mathbf{f}}$ by adapting τ in every iteration. The changing value of τ makes the difference between RUN and the standard maximum likelihood method. Theorem 2.2 reveals the relation between τ and n_{df} by re-formulating the local model.

Theorem 2.2 *Consider the eigen-decomposition* $\mathbf{H} = \mathbf{U}\mathbf{D}\mathbf{U}^T$ *of the Hessian* $\mathbf{H}$ *from Equation 2.27 and define* $\mathbf{D}^{-\frac{1}{2}} = diag(\frac{1}{\sqrt{\mathbf{D}_{11}}}, \ldots \frac{1}{\sqrt{\mathbf{D}_{II}}})$. *Let the transformed curvature matrix* $\mathbf{C}' = \mathbf{D}^{-\frac{1}{2}}\mathbf{U}^T\mathbf{C}\,\mathbf{U}\,\mathbf{D}^{-\frac{1}{2}}$ *have the eigen-decomposition* $\mathbf{C}' = \mathbf{U}'\mathbf{D}'\mathbf{U}'^T$.

The regularized local model $\hat{\ell}_r^{(k)}$ *from Equation 2.28 can be rewritten in the following form to be interpreted as a surrogate function of* $\mathbf{f}'' = \mathbf{U}'^T\mathbf{D}^{\frac{1}{2}}\mathbf{U}^T\,\mathbf{f}$. *Its actual minimizer is* $\hat{\mathbf{f}} = \mathbf{U}\,\mathbf{D}^{-\frac{1}{2}}\mathbf{U}'\,\hat{\mathbf{f}}''$, *when* $\hat{\mathbf{f}}''$ *is the minimizer of this surrogate form.*

$$\hat{\ell}_r^{(k)}(\mathbf{f}) \;=\; \frac{1}{2}\mathbf{f}''^T(\,I \;+\; \tau \cdot \mathbf{D}'\,)\,\mathbf{f}'' \;-\; \mathbf{f}''^T(\mathbf{U}\,\mathbf{D}^{-\frac{1}{2}}\mathbf{U}')^T(\mathbf{H}\,\mathbf{f}^{(k)} \;-\; \boldsymbol{\ell})$$

Proof. Theorem 2.2 directly follows from the given definitions. The full proof is given in Section A.3.

Theorem 2.3 *The minimizer $\hat{\mathbf{f}}''$ of the reformulated model from Theorem 2.2 is given by the following equation, where $\hat{\tilde{\mathbf{f}}}''$ is the un-regularized minimizer obtained for $\tau = 0$.*

$$\hat{\mathbf{f}}''_i \;=\; \frac{1}{1 \,+\, \tau\, \mathbf{D}'_{ii}}\, \hat{\tilde{\mathbf{f}}}''_i \quad \textit{where} \quad \hat{\tilde{\mathbf{f}}}'' \;=\; \left(\mathbf{U}\, \mathbf{D}^{-\frac{1}{2}} \mathbf{U}'\right)^T \left(\mathbf{H}\, \mathbf{f}^{(k)} \,-\, \boldsymbol{\ell}\right)$$

Proof. The equation is obtained by setting the gradient of the reformulated local model to zero. Its full derivation is given in Appendix A.3.

The sum of the regularizing factors $\frac{1}{1+\tau\, \mathbf{D}'_{ii}}$ from Theorem 2.3 is considered the effective number of degrees of freedom n_{df} of the minimizer $\hat{\mathbf{f}}$, as defined in Equation 2.30. In case of $\tau = 0$, each of the I components of $\hat{\tilde{\mathbf{f}}}''$ fully contributes to the solution and thus $n_{df} = I$. Accordingly, the total contribution of $\hat{\tilde{\mathbf{f}}}''$ to $\hat{\mathbf{f}}$ is reduced whenever τ is greater than zero. Conversely, the value of τ has to be adapted so that the value of n_{df}—which is fixed by the practitioner—is actually attained. Moreover, τ's value has to be changed in every iteration because $\hat{\mathbf{f}}''$ only minimizes the local model. According to the Newton method, this model determines the next estimate of the RUN algorithm. Multiple iterations are required to find a local minimizer of ℓ_r that represents the final estimate of RUN.

$$n_{df} \;=\; \sum_i \frac{1}{1 \,+\, \tau\, \mathbf{D}'_{ii}} \tag{2.30}$$

2.4.3 The Algorithm

Algorithm 2.3 plugs together the concepts presented in the previous two subsections, iteratively minimizing a local model $\hat{\ell}_r^{(k)}$ of the regularized loss function ℓ_r. The regularization strength $\tau^{(k)}$ of each iteration is accordingly chosen based on the reformulation of $\hat{\ell}_r^{(k)}$ from Theorem 2.2. The algorithm returns an estimate $\hat{\mathbf{f}}$ of the discrete target density, its input being the detector response matrix $\mathbf{R}$, the discrete observed density $\mathbf{g}$, and the desired n_{df} chosen by the practitioner. Moreover, a minimal difference $\epsilon > 0$ of losses between iterations is given, which is used to stop the algorithm eventually.

The first estimate $\mathbf{f}^{(1)}$ of RUN is the least-squares solution presented in Equation 2.31, where $\boldsymbol{\ell}_{LSq}$ is the gradient and $\mathbf{H}_{LSq}$ is the Hessian of ℓ_{LSq} at the zero vector. Equation 2.32 presents ℓ_{LSq}, which is the loss function associated to the method of least squares. $\mathbf{f}^{(1)}$ is a suitable starting point for the iterative algorithm, which requires *some* initial estimate. The least-squares solution $\mathbf{f}^{(1)}$ is obtained from every prior at which the gradient and the Hessian of the convex function ℓ_{LSq} are computed. The zero vector is only chosen for convenience.

$$\mathbf{f}^{(1)} \;=\; -\; \mathbf{H}_{LSq}^{-1}\, \boldsymbol{\ell}_{LSq} \tag{2.31}$$

$$\ell_{LSq}(\mathbf{f}) \;=\; \frac{1}{2} \sum_j \left(\, \bar{\mathbf{g}}_j - \mathbf{R}_{\bullet j}^T\, \bar{\mathbf{f}} \,\right)^2 \tag{2.32}$$

Input:

Discrete observed density $\mathbf{g} \in \mathbb{R}^J$

Detector response matrix $\mathbf{R} \in \mathbb{R}^{I \times J}$

$\epsilon > 0$, the minimal difference between losses ℓ_r in subsequent iterations

$n_{df} \leq I$, the desired number of degrees of freedom

Output:

Estimated discrete target density $\hat{\mathbf{f}} \in \mathbb{R}^I$

$k \leftarrow 1$

$\mathbf{f}^{(1)} \leftarrow -\mathbf{H}_{LSq}^{-1} \boldsymbol{\ell}_{LSq}$

repeat

  $k \leftarrow k + 1$

  Decompose $\mathbf{H} = \mathbf{U}\,\mathbf{D}\,\mathbf{U}^T$

  $\mathbf{C}' \leftarrow \mathbf{D}^{-\frac{1}{2}}\mathbf{U}^T\mathbf{C}\mathbf{U}\mathbf{D}^{-\frac{1}{2}}$

  Decompose $\mathbf{C}' = \mathbf{U}'\mathbf{D}'\mathbf{U}'^T$

  Choose $\tau^{(k)}$ satisfying $n_{df} = \sum_i \frac{1}{1+\tau^{(k)}\mathbf{D}'_{ii}}$

  $\hat{\mathbf{f}}'' \leftarrow \left(I + \tau^{(k)} \cdot \mathbf{D}'\right)^{-1} \left(\mathbf{U}\mathbf{D}^{-\frac{1}{2}}\mathbf{U}'\right)^T \left(\mathbf{H}\hat{\mathbf{f}}^{(k-1)} - \boldsymbol{\ell}\right)$

  $\hat{\mathbf{f}}^{(k)} \leftarrow \left(\mathbf{U}\mathbf{D}^{\frac{1}{2}}\mathbf{U}'\right)\hat{\mathbf{f}}''$

until $\left|\ell_r(\hat{\mathbf{f}}^{(k)}) - \ell_r(\hat{\mathbf{f}}^{(k-1)})\right| \leq \epsilon$

return $\hat{\mathbf{f}} \leftarrow \hat{\mathbf{f}}^{(k)}$

Algorithm 2.3: Regularized Unfolding [2]. $\boldsymbol{\ell} = \nabla \ell\left(\hat{\mathbf{f}}^{(k-1)}\right)$ and $\mathbf{H} = \nabla^2 \ell\left(\hat{\mathbf{f}}^{(k-1)}\right)$ represent the gradient and the Hessian of the loss function ℓ from Theorem 2.1 at the latest estimate $\hat{\mathbf{f}}^{(k-1)}$. $\boldsymbol{\ell}_{LSq}$ and $\mathbf{H}_{LSq}$ accordingly denote the gradient and Hessian of the least squares loss at $\mathbf{0} \in \mathbb{R}^I$.

2.4.4 Expansion and Reduction of the Problem

The most recent publication on RUN suggests to perform an additional post-processing step that reduces the dimension of a result $\hat{\mathbf{f}}' \in \mathbb{R}^{I'}$ by averaging the values of neighboring bins [2]. Here, we refer to this step as a *reduction* of the solution with the factor $e \in \mathbb{N}$. Reducing the solution means to obtain the low-dimensional result $\hat{\mathbf{f}} \in \mathbb{R}^I$ from Equation 2.33. The publication suggests to fit $\hat{\mathbf{f}}'$ with RUN, choosing $n_{df} = I < I'$.

$$\hat{\mathbf{f}}_i = \frac{1}{e} \cdot \sum_{i'=i}^{i'+e} \hat{\mathbf{f}}_{i'} \tag{2.33}$$

In order to obtain results with a final dimension equal to that of the other methods, RUN is applied to an *expanded* deconvolution problem with $I' = e \cdot I$ discrete target values. Reducing the result $\hat{\mathbf{f}}'$ of an expanded problem yields the initial dimension I, which is also used in the other methods. Effectively, this expansion/reduction approach smooths the final result $\hat{\mathbf{f}}$ in addition to the regularization already applied while fitting $\hat{\mathbf{f}}'$ with RUN.

2.5 Other Algorithms

In the scope of this work, IBU and RUN are presented as representatives of classical deconvolution methods. They have been selected for this purpose due to their popularity and due to their different motivations. However, more approaches to deconvolution exist, some of which are roughly sketched in the following. These approaches are based on singular value decomposition, on neural networks, and on a learning task formulated with respect to densities instead of individual observations.

2.5.1 Singular Value Decomposition

The approach by Höcker and Kartvelishvili [20] is based on the singular value decomposition (SVD) of the detector response matrix. A similar approach is presented already with the naive estimator from Subsection 2.2.3. However, the SVD approach employs regularization in order to achieve more robust results. The naive estimator does not regularize its estimate. Before applying regularization, SVD finds an un-regularized solution to a surrogate problem. The regularized solution to the actual problem is then derived from this un-regularized surrogate solution. These two steps are quite similar to the RUN algorithm. However, the computations are slightly different because RUN performs an eigen-decomposition instead of finding the singular values.

In addition to these aspects, some pre-processing steps are described for SVD, which are not part of RUN and the naive estimator. Therefore, the SVD approach may produce quite different results, despite its similarities to these methods. Consequently, it is an important direction of future work to study the effects of the pre-processing steps proposed for the SVD deconvolution.

2.5.2 Neural Networks for Conditional Densities

The primary goal of the approach by Feindt [21] is to estimate the probability density of each *individual* observation. In contrast, all other methods considered here estimate the combined target density of an entire set of observations. However, DSEA is based on density estimates for individual observations which are returned by the embedded classifier. One direction of future work is thus to combine the neural network approach with DSEA by embedding the neural net as a classifier.

Another aspect of the neural network approach to deconvolution is that continuous density estimates are returned, instead of histograms. DSEA can presumably be extended to deliver such continuous estimates, adapting this aspect from the neural network approach. However, continuous estimates run the risk of concealing the level of detail with which an estimate is fit to the data.

2.5.3 A Learning Task on Density Functions

The approach by Gagunashvili [22] is based on machine learning, like Dsea. However, it does not embed a classifier to predict the target value of individual observations. Instead, entire target densities are predicted from observed densities. This learning task is thus quite different from the task to which Dsea translates the deconvolution problem. Effectively, it only learns the relation between the observed density and the target density, which is then expressed in a detector response matrix, as usual. In the scope of the present thesis, the detector response matrix is always learned from relative numbers, of which the computation is straight-forward.

Having estimated the detector response, the approach by Gagunashvili obtains a simple least squares solution to the discrete deconvolution problem. This step is similar to the naive estimator from Subsection 2.2.3.

Improving the detector response matrix with machine learning is a relevant direction of research. However, the approach by Gagunashvili requires multiple densities to learn from. It thus requires a high volume of training data, which is costly in IACTs. Another approach in the same direction is already presented here: Discretizing the feature space with a decision tree improves the condition of the detector response matrix with a method from machine learning.

Chapter 3

Regularized DSEA

The original version of DSEA diverges from the true target density after having found a suitable estimate. Since this behavior renders the choice of the iteration number K critical, it is the first goal of the present thesis to stop DSEA from diverging. With a less critical choice of K, it is more likely that the algorithm produces accurate results in practice, where it is applied to previously unseen data for which the optimal K is not known. Section 3.1 tracks the cause of the divergence down to the re-weighting rule of DSEA, which intends to adjust the example weights so that the weighted training set follows the observed density. It is shown that the original re-weighting rule does not implement this intention. This issue is solved by a corrected re-weighting rule, which stops DSEA from diverging. The corrected algorithm DSEA$^+$ converges to a suitable estimate of the true target density.

Going beyond this achievement, the algorithm is accelerated to converge faster and more accurately. For this purpose, the notion of a scalable step between iterations is introduced in Section 3.2. Building up on this concept, Section 3.3 develops an adaptive step size strategy which results in the accelerated method. DSEA$^+$ bundles these aspects, being presented in Section 3.4. Two additional aspects are investigated in Appendix B. However, they do not improve DSEA significantly, and they are therefore excluded from the powerful deconvolution tool presented here.

3.1 Correct Re-weighting of Training Examples

The re-weighting in DSEA intends to adjust the density of the *weighted* training set to the respective latest estimate $\hat{\mathbf{f}}$ of the observed target density. However, the original update rule fails to implement this intention if the training set is not uniformly distributed. This defect results from the actual, *un-weighted* density $\mathbf{f}^t \in \mathbb{R}^I$ of the training set not being accounted for. Instead of weighting each $\mathbf{x}_n \in \mathcal{D}_{\text{train}}$ with the estimated probability $\hat{\mathbf{f}}_{i(n)}$ of its respective bin $i(n) = d_f(y_n)$, each example has to be weighted with the *ratio* between

n	$y_{i(n)}$	w_n	w_n^+
1	$\mathcal{Y}_1$	$\frac{1}{3}$	1
2	$\mathcal{Y}_2$	$\frac{2}{3}$	1
3	$\mathcal{Y}_2$	$\frac{2}{3}$	1

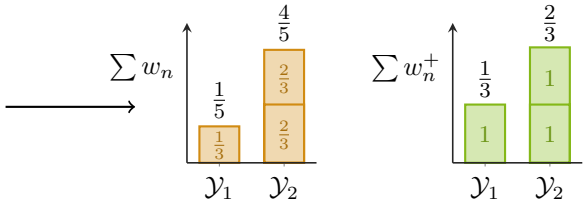


Figure 3.1: The original and the corrected weighting are applied to a trivial training set, of which each example is presented in one row of the left table. The two bar plots present the weighted histograms of this set, clarifying that only the corrected weighting is appropriate.

this estimated probability and the corresponding probability in the un-weighted training set. Such corrected update rule is presented in Equation 3.1. It replaces the second line of the original DSEA stated in Algorithm 2.1.

$$w_n^{(k)+} \;=\; \hat{\mathbf{f}}_{i(n)}^{(k-1)} \;/\; \mathbf{f}_{i(n)}^{t} \qquad \forall\; 1 \leq n \leq N \tag{3.1}$$

Figure 3.1 illustrates the inadequacy of the original weight update with a simple example. The table in this figure presents a trivial training set consisting of three observations in two bins $\mathcal{Y}_1$ and $\mathcal{Y}_2$. Accordingly, the training set density $\mathbf{f}^t = (\frac{1}{3}, \frac{2}{3})$ is not uniform. Now imagine that same density to be estimated for the observed data, i.e. $\hat{\mathbf{f}} = \mathbf{f}^t$. Since both densities already match, all weights should have the same value so that the weighted training set is effectively un-weighted. Otherwise, the weighted training density would not match $\hat{\mathbf{f}}$, which is the unfortunate case with the original update rule of DSEA. Normalizing the bin weights obtained from this rule produces the weighted training set density $(\frac{1}{5}, \frac{4}{5}) \neq \hat{\mathbf{f}}$, which is displayed in the left bar plot in Figure 3.1. Equation 3.1 produces the right bar plot, which correctly resembles the estimate $\hat{\mathbf{f}}$.

Figure 3.2 shows that the corrected update rule stops DSEA from diverging. This effect is particularly noticeable on training sets that are not uniformly distributed.

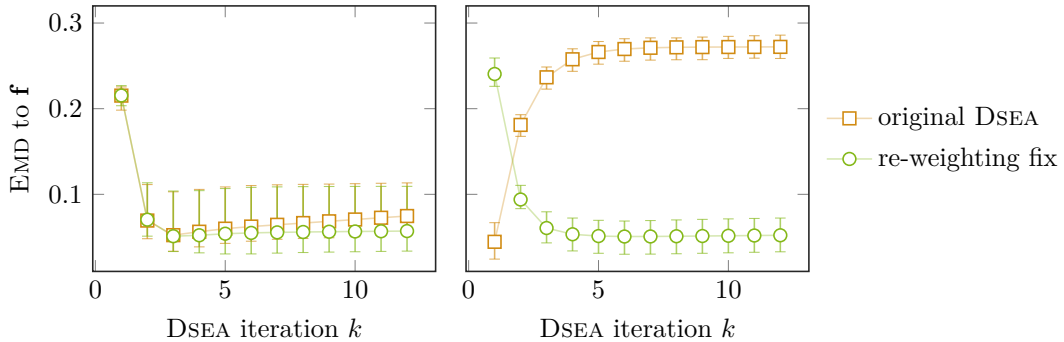


Figure 3.2: The corrected update rule stops DSEA from diverging. This observation is made with two training sets sampled from toy data. The left plot utilizes a uniform training set density, for which the improvement is expectedly small. A non-uniform training set is used on the right, where the original DSEA diverges dramatically and the improvement is significant.

3.2 Scalable Steps

The notion of a scalable step between iterations is based on the following concept: By iteratively updating the estimate $\hat{\mathbf{f}}$, a suitable approximation of the true density $\mathbf{f}$ is searched for in the space of all possible solutions. The difference between two subsequent estimates corresponds to a step $p^{(k)} \in \mathbb{R}^I$ taken through this search space. Equation 3.2 defines this step, which is also referred to as the *search direction* of the k-th iteration. The original DSEA takes the full step in each iteration, moving from one estimate to the other. DSEA$^+$ extends this original algorithm by scaling each $p^{(k)}$ with a corresponding factor $\alpha^{(k)} \geq 0$. This extension is formalized in Equation 3.3, which computes the improved estimate $\hat{\mathbf{f}}^{(k)+}$ from a scaled step. The benefit of this extension is that the practitioner can control the speed of convergence by choosing a suitable step size. Note that the original estimate $\hat{\mathbf{f}}^{(k)}$ is directly obtained by setting $\alpha^{(k)} = 1$.

$$p^{(k)} = \hat{\mathbf{f}}^{(k)} - \hat{\mathbf{f}}^{(k-1)} \tag{3.2}$$

$$\hat{\mathbf{f}}^{(k)+} = \hat{\mathbf{f}}^{(k-1)} + \alpha^{(k)} \cdot p^{(k)} \tag{3.3}$$

This concept of a scalable step requires an appropriate strategy for choosing $\alpha^{(k)}$ in each iteration. Table 3.1 reviews some simple strategies considered here, the last two of which depend on the iteration number k. Each of these strategies is parametrized by a single scalar also presented in the table. Such parameter controls the convergence speed of the algorithm. An adaptive strategy, which evaluates the improvement of the estimate in the given search direction, is presented in the following section.

strategy	parameter	step size
constant factor	$\alpha > 0$	$\alpha^{(k)} = \alpha$
multiplicative decay (slow)	$0 < \eta < 1$	$\alpha^{(k)} = k^{(\eta-1)}$
exponential decay (fast)	$0 < \eta < 1$	$\alpha^{(k)} = \eta^{(k-1)}$

Table 3.1: Some simple strategies determine the step size $\alpha^{(k)}$. η is referred to as the *decay rate*. Each parameter controls the speed of the convergence in DSEA$^+$.

Scalable steps render the choice of the final iteration number K less critical than it is in the original algorithm. A simple constant stepsize $\alpha < 1$ already approaches the optimal solution more slowly than the original DSEA, thus providing a larger range of suitable iterations. The decaying step sizes have the additional benefit of effectively stopping the algorithm gradually by approaching zero over time. Therefore, a maximum number of iterations does not have to be specified with one of these strategies. Instead, the χ^2 stopping criterion already proposed for DSEA [4] becomes feasible.

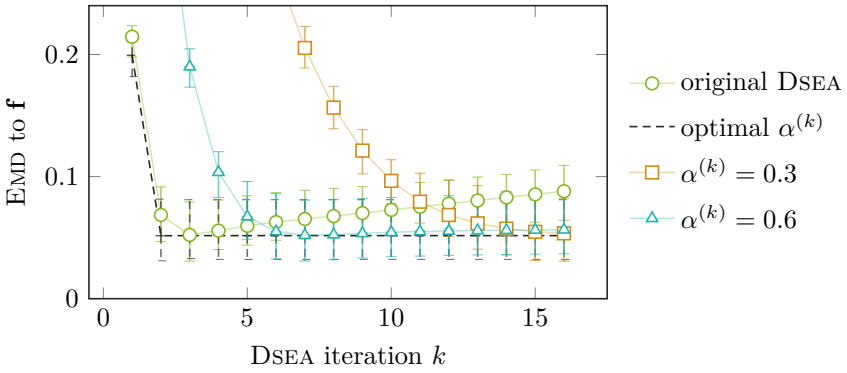


Figure 3.3: Two constant step sizes are compared to the original DSEA with step size one. The constant step size strategy does not stop DSEA from diverging, which is indicated by the Earth Mover's Distance slightly increasing with k. However, the constant step sizes stretch the range of suitable iterations. For reference, the results obtained with an *optimal* step size are given. This optimal size is chosen with full knowledge about the true density $\mathbf{f}$, which is not available in practice. It thus produces the best estimates theoretically obtainable with the stepsize extension.

Figure 3.3 presents an experiment on toy data, in which different constant step sizes are applied. These step sizes do not regularize DSEA, but they stretch the range of suitable iterations to stop at. Compared to the original algorithm, it is therefore less critical to select a total number K of iterations. On the other hand, more iterations of DSEA are performed with these strategies, leading to an increased run time.

The convergence of the two decay strategies is presented in Figure 3.4, where the same toy data set is used. Apparently, the decay rate η is a critical parameter for the fast exponential decay. For the multiplicative strategy, this parameter appears less critical.

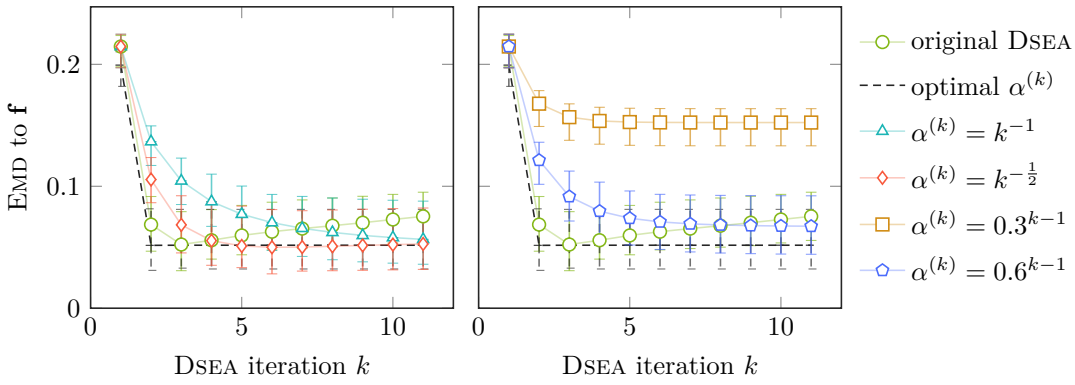


Figure 3.4: The two decay strategies are compared to the original DSEA. Exponential decay (right) converges much faster than the multiplicative decay strategy (left). It even stops too early, if not configured appropriately. The original DSEA and the optimal step size are presented already in Figure 3.3.

3.3 Adapting the Step Size

Configuring the simple step size strategies is already less critical than specifying the iteration number K in the original DSEA. However, making the wrong choice—for example due to differences between the simulated data and the observed data—can lead to deconvolution results that are not optimal. Instead of fixing $\alpha^{(k)}$ to a constant value or decay rate, the following strategy therefore adapts the step size to the previous estimate $\hat{\mathbf{f}}^{(k-1)}$ and to the search direction $p^{(k)}$. This adaptive strategy accelerates the convergence while improving the accuracy of the deconvolution result.

The adaptive strategy minimizes the regularized objective function ℓ_r known from the RUN algorithm (see Section 2.4) in the direction of search. Thus, it maximizes the likelihood of the next estimate in the direction determined by DSEA. In contrast, RUN attempts to maximize the likelihood in the I-dimensional space of all possible solutions, not only in this single direction.

$$\alpha_{\text{RUN}}^{(k)} \;=\; \underset{\alpha \geq 0}{\arg\min}\; \ell_r \left(\hat{\mathbf{f}}^{(k-1)} \;+\; \alpha \cdot p^{(k)} \right) \tag{3.4}$$

Equation 3.4 is parametrized with the regularization strength $\tau \geq 0$ contained in ℓ_r. However, this parameter has almost no impact on the step size. In contrast to RUN, τ is therefore not changed in every iteration. Moreover, $\tau = 0$ is a reasonable default value, which corresponds to no regularization of the adaptive step size. Note that the adaptive strategy is inspired by a usual algorithmic pattern in numerical optimization, where the basic algorithm selects a search direction and a sub-routine called *line search* selects an appropriate step size for this direction [19].

Figure 3.5 presents the adaptive strategy on the toy data set. This strategy (almost) attains the the optimal step size for different values of τ. Remarkably, it takes a first

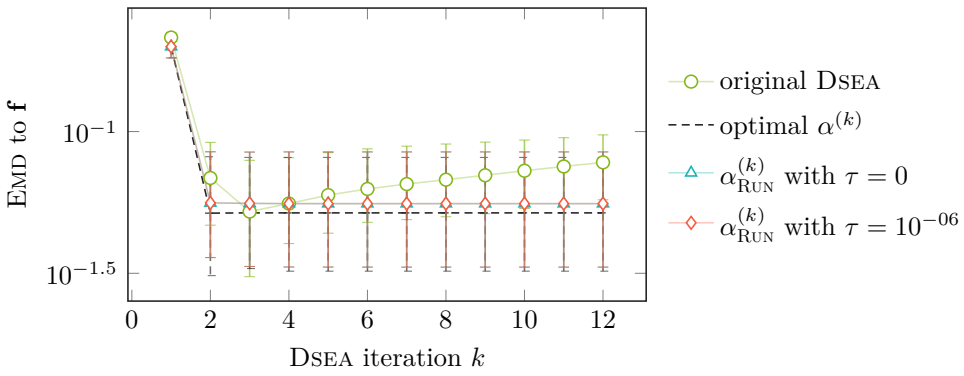


Figure 3.5: The adaptive step size strategy is compared to the original DSEA. Convergence happens quickly and in a remarkable proximity to the true solution $\mathbf{f}$. Note that the y axis scale is logarithmic in this plot.

step that is larger than one, thus approaching the optimal solution faster than the original DSEA. Moreover, the algorithm stops as soon as the adaptive strategy does not expect any further improvement of the estimate, thus saving resources by not wasting iterations.

3.4 The Extended Algorithm

Algorithm 3.1 plugs together the aforementioned aspects, which are the corrected re-weighting of training examples and the step size, which is adaptively chosen by maximizing the likelihood in the direction of search. The convergence is checked with the χ^2_{Sym} distance between subsequent iterations. This distance is small whenever $\alpha^{(k)}_{\text{RUN}}$ is.

Input:

Observed data set $\mathcal{D}_{\text{obs}} = \{\, \mathbf{x}_n \in \mathcal{X} \,:\, 1 \le n \le N \,\}$

Training data set $\mathcal{D}_{\text{train}} = \{\, (\mathbf{x}_n,\, y_n) \in \mathcal{X} \times \{1, 2, \ldots, I\} \,:\, 1 \le n \le N' \,\}$

$\tau \ge 0$, the regularization strength employed in the stepsize adaption (default: 0)

$\epsilon > 0$, the minimal χ^2_{Sym} distance between subsequent iterations (default: 10^{-6})

Prior density $\hat{\mathbf{f}}^{(0)}$ (default: $\hat{\mathbf{f}}^{(0)}_i = \frac{1}{I} \;\; \forall\, 1 \le j \le J$)

Output:

Estimated target density $\hat{\mathbf{f}} \in \mathbb{R}^I$

1: $k \leftarrow 0$

2: **repeat**

3: $\quad k \leftarrow k + 1$

4: $\quad \forall\, 1 \le n \le N' : \;\; w^{(k)+}_n \leftarrow \hat{\mathbf{f}}^{(k-1)}_{i(n)} \;/\; \mathbf{f}^{t}_{i(n)}$

5: $\quad$ Infer $\mathcal{M}$ from $\mathcal{D}_{\text{train}}$ weighted by $w^{(k)+}_n$

6: $\quad \forall\, 1 \le i \le I : \;\; p^{(k)}_i \leftarrow \frac{1}{N} \sum_{n=1}^{N} c_{\mathcal{M}}(\, i \,|\, \mathbf{x}_n) \;-\; \hat{\mathbf{f}}^{(k-1)}_i$

7: $\quad \alpha^{(k)}_{\text{RUN}} \leftarrow \arg\min_{\alpha \ge 0} \;\; \ell_r \left(\, \hat{\mathbf{f}}^{(k-1)} \;+\; \alpha \cdot p^{(k)} \,\right)$

8: $\quad \hat{\mathbf{f}}^{(k)+}_i \leftarrow \hat{\mathbf{f}}^{(k-1)} + \alpha^{(k)}_{\text{RUN}} \cdot p^{(k)}$

9: **until** $\chi^2_{\text{Sym}}\big(\hat{\mathbf{f}}^{(k)},\, \hat{\mathbf{f}}^{(k-1)}\big) \le \epsilon$

10: **return** $\hat{\mathbf{f}} \leftarrow \hat{\mathbf{f}}^{(k)}$

Algorithm 3.1: The improved Dortmund Spectrum Estimation Algorithm DSEA$^+$ corrects the re-weighting of the training examples and adapts the step size $\alpha^{(k)}_{\text{RUN}}$ in each iteration. The algorithm stops when the χ^2_{Sym} distance is below a given threshold ϵ, which is reached when $\alpha^{(k)}_{\text{RUN}}$ is small.

Chapter 4

Comparative Evaluation

The goal of the present chapter is to find the best deconvolution method among the ones presented. This goal is specifically targeted at Cherenkov astronomy, being approached with data sets from two different IACTs. All algorithms are trained with multiple densities of the training set and their configuration is optimized over a comprehensive grid of meta-parameters. All experiments are conducted on multiple random bootstrap samples.

Section 4.1 introduces, how the quality of individual deconvolution results is measured. The data sets utilized in the experiments are presented afterwards, in Section 4.2. Section 4.3 describes the overall experimental setup, which employs bootstrapping and two different training densities to assess the robustness of the deconvolution results. The methods are compared with each other in Section 4.4.

4.1 Assessing the Quality of Deconvolution Results

The accuracy of deconvolution is assessed by measures of *statistical distance*. Such distance is a quantity of dissimilarity between probability density functions. The quality of a deconvolution result $\hat{\mathbf{f}}$ is captured by such quantity being evaluated between $\hat{\mathbf{f}}$ and the respective true density $\mathbf{f}$ of the deconvolved data set. A large distance between $\hat{\mathbf{f}}$ and $\mathbf{f}$ indicates that $\hat{\mathbf{f}}$ is not accurate. There is a plethora of statistical distance measures, many of which have similar semantics [23]. In the scope of this thesis, the Earth Mover's Distance (EMD) [24] is favored, which naturally extends the *ground distance* between individual examples to densities of example sets. The EMD is therefore particularly well suited for the evaluation of deconvolution results. This measure, which is also referred to as Wasserstein metric [25], is presented in the following.

Equation 4.1 defines the distance δ_{EMD} as the minimum cost of transforming one histogram into the other. Such transformation is defined by a set F of flows $f_{ii'}$ between the bins $i, i' \in \{1, 2, \ldots, I\}$ of the histograms $\mathbf{f}$ and $\hat{\mathbf{f}}$. Each flow induces some cost $c_{ii'} \cdot f_{ii'}$

depending on the cost per element $c_{ii'}$ between the bins i and i'. The metric $c_{ii'}$ is referred to as the *ground distance*, giving the dissimilarity of individual examples from these bins.

$$\delta_{\text{EMD}}\left(\hat{\mathbf{f}}, \mathbf{f}\right) = \min_{F} \sum_{f_{ii'} \in F} c_{ii'} \cdot f_{ii'} \tag{4.1}$$

$$\begin{aligned} \text{subject to} \qquad f_{ii'} &\geq 0 \\ \textstyle\sum_{i'=1}^{I} f_{ii'} &= \mathbf{f}_i \\ \textstyle\sum_{i=1}^{I} f_{ii'} &\leq \hat{\mathbf{f}}_{i'} \qquad \forall\ i, i' \in \{1, 2, \ldots, I\} \end{aligned}$$

Finding the minimum in Equation 4.1 requires to optimize the flow F between the two histograms. Such optimization corresponds to a bipartite network flow problem called *transportation problem*. Figure 4.1 illustrates this problem with the structure of the corresponding graph. The optimization is constrained so that applying the flow to a discrete probability density function correctly results in another density function. It should be noted that the original EMD [24] is more general than the δ_{EMD} presented here because it is defined over *signatures* instead of histograms. Signatures are variable-length descriptions of densities, which are more versatile than histograms but not required here. Note that Equation 4.1 omits the normalization factor $\sum_{f_{ii'} \in F} f_{ii'}$ proposed in [24] because it is equal to 1 for any probability density function.

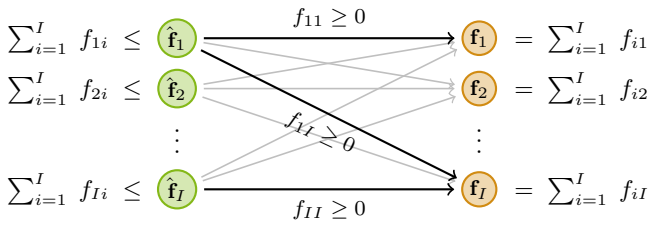


Figure 4.1: Bipartite network of bins in $\mathbf{f}$ and $\hat{\mathbf{f}}$ with flows $f_{ii'} \in F$ in between. Only some of the flows are annotated here for visibility, but flows are present between all pairs of bins. The EMD minimizes the total cost of all flows, which is a sum of individual costs $c_{ii'} \cdot f_{ii'}$. Such minimization is constrained by the equalities and inequalities placed next to the nodes.

Solving the linear optimization problem imposed by Equation 4.1 is computationally expensive. However, limiting the EMD to special cases can enable researchers to develop more efficient algorithms computing this distance. The ordinal Minimum Distance of Pair Assignments (MDPA) [26] is one such algorithm which is limited in the following ways when compared to the general EMD:

- MDPA only applies to univariate histograms
- Both histograms in consideration must have the same number I of bins
- The absolute distance between bin indices is used as the ground distance, namely $c_{ii'} = |i - i'| \quad \forall\ i, i' \in \{1, 2, \ldots, I\}$.

MDPA coincides with the EMD if these properties are met, which holds for all experiments conducted here. Accordingly, Algorithm 4.1 computes the EMD efficiently—it only requires a single pass over all bins to do so. The idea behind this algorithm is that examples in the univariate histograms $\mathbf{f}$ and $\hat{\mathbf{f}}$ can only be shifted leftwards or rightwards. The excess shifted over the i-th bin is captured in the corresponding variable $e^{(i)}$. The sum of absolute shifts is the EMD under the limitations given above [26].

By the way, in the analogy that gave the EMD its name, one density is seen as a pile of dirt and the other one is seen as holes accordingly placed. The Earth Mover's Distance is the minimum amount of work required to fill all holes, which is induced by moving some of the dirt. The further some dirt has to be moved, the more work is required.

Input:

Univariate discrete density functions $\mathbf{f}$ and $\hat{\mathbf{f}}$, each with $I \in \mathbb{N}$ bins.

Output:

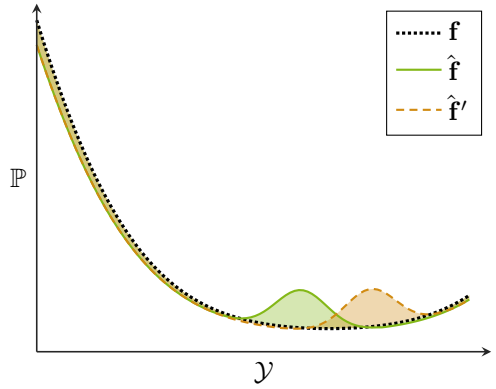
$\delta_{\text{EMD}} \;=\; \sum_{i=1}^{I} \left| \, e^{(i)} \, \right|$

1: $e^{(0)} \;\leftarrow\; 0$
2: **for** $i \;\leftarrow\; \{1, 2, \ldots, I\}$ **do**
3: $\quad e^{(i)} \;\leftarrow\; e^{(i-1)} \;+\; \mathbf{f}_i \;-\; \hat{\mathbf{f}}_i$

Algorithm 4.1: The ordinal Minimum Distance of Pair Assignments (MDPA) [26]. This algorithm computes the Earth Mover's Distance δ_{EMD} for univariate discrete density functions with the same number of bins $I \in \mathbb{N}$ if the ground distance is the absolute distance between bin indices, i.e. $c_{ii'} = |i - i'| \;\; \forall \;\; i, i' \in \{1, 2, \ldots, I\}$.

In addition to the Earth Mover's Distance, multiple other distance measures are evaluated in the experiments to provide results that are comparable with other studies. Any other study, which may use one of these additional measures, but not the EMD, can thus be related to the present thesis. These additional measures are defined in the Section C.3. Plots of their experimental values are not presented here, due to the particular suitability of the EMD for quality assessment in deconvolution. However, all distances are provided for all experiments with the supplementary material of this thesis. Notably, all metrics produce plots with similar interpretations. The following distance measures are considered in addition to the EMD:

- δ_{KL} – the Kullback-Leibler divergence
- δ_{H} – the Hellinger distance
- $\delta_{\chi^2_P}$ – Pearson's χ^2 divergence
- $\delta_{\chi^2_{Sym}}$ – the probabilistic symmetric χ^2 distance



	$\delta(\hat{\mathbf{f}}, \mathbf{f})$		$\delta(\hat{\mathbf{f}}', \mathbf{f})$
δ_{EMD}	0.708	$<$	0.943
δ_{KL}	0.03	$\approx$	0.029
δ_{H}	0.081	$\approx$	0.08
$\delta_{\chi^2_P}$	0.079	$\approx$	0.075
$\delta_{\chi^2_{Sym}}$	0.026	$\approx$	0.025

Figure 4.2: The selected distances are compared to each other on discrete variants $\hat{\mathbf{f}}$, $\hat{\mathbf{f}}'$, and $\mathbf{f}$ of the continuous functions displayed on the left. $\hat{\mathbf{f}}$ and $\hat{\mathbf{f}}'$ resemble $\mathbf{f}$ quite well, except for their respective bumps. The areas spanned by these two local divergences are equal, resulting in approximately equal values in most distance measures. δ_{EMD} is the only distance that takes different values because it appropriately considers the position of local divergences.

Compared to these classical distance measures, the EMD is the only distance taking the position of local divergences into account. Figure 4.2 indicates this property by evaluating δ_{EMD} and the other distance measures between some example densities. The only distance which takes different values for the two presented divergences is the EMD. Since the right bump is further away from the main part of the density function, this property is desired for distance measures that evaluate particle energy densities. Namely, a practitioner would consider the density with the right bump to be less accurate than the other one, if $\mathbf{f}$ is the true density. The EMD is therefore particularly suitable for the evaluation of deconvolution algorithms in the IACT domain.

4.2 Data Sets

The deconvolution methods are evaluated on multiple data sets. First of all, a set of toy data is presented here, which is already used to obtain the plots presented in the previous chapters. This toy data is synthetically generated and therefore available in abundance. Due to its arbitrary volume and due to the absence of unintended noise, this data set is primarily used to study the improvements already proposed for DSEA. However, it is also used in the final comparison of methods, which is conducted later in this chapter.

The other two data sets come from the IACT domain, for which the best deconvolution method among the presented ones is searched. These two data sets have been simulated for two different telescopes. They differ in the number of available examples and in the range of the energy density that is deconvolved.

4.2.1 Toy Data Set

The target density of the observed toy data set is a mixture of two Gaussian components, which is already presented as the true density $\mathbf{f}$ in Figure 1.2 and in Figure 2.3. In contrast to this observed data, the separate training data set is uniformly distributed in the target quantity. Using two different densities for training and deconvolution is motivated by the idea that a deconvolution method should produce a result which is independent from the density assumed for training. This desired property is checked by choosing an inappropriate training density on purpose.

The target quantity values of each data set are drawn at random, according to their respective density. These values are then used to generate ten observable quantities for both data sets. The values of these observable quantities are generated according to Equation 4.2, which defines the value of the k-th quantity in the n-th example based on the target value $y_n \in \mathcal{Y}$ previously drawn. The scalar $\mathcal{N}(\, y_n, \sigma_k) \in \mathbb{R}$ in this definition refers to a single value sampled from a Gaussian density with the standard deviation σ_k and the mean y_n. The real-valued parameters o_k, f_k, σ_k, and e_k of each quantity are randomly chosen beforehand. They are identical for the observed data and the training set, so that the relation between the target quantity and each observable is fixed.

$$\mathbf{x}_{n,k} \quad = \quad o_k \; + \; f_k \; \cdot \; \mathcal{N}\left(\, y_n, \sigma_k \,\right)^{e_k} \qquad \forall \; 1 \leq k \leq 10, \;\; 1 \leq n \leq N \tag{4.2}$$

Figure 4.3 presents the densities of three of the ten observable quantities in the training and deconvolution data sets. Due to the flat target density of the training set, the double-peak structure of the deconvolved density is not present there. It remains to be noted that the generation according to Equation 4.2 is slightly different from what has been published for a similar data set used to evaluate DSEA elsewhere [27]. However, the generation presented here strictly implements the computation taken out by the author of this publication[1].

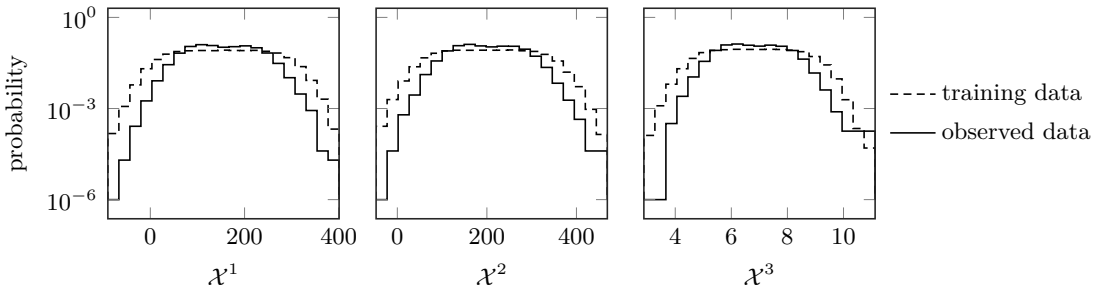


Figure 4.3: The pre-images of the observable quantities X^1, X^2, and X^3 have two slight peaks in the deconvolution data set, which are not present in the uniformly distributed training data.

[1] Source code is available to users authorized by Tim Ruhe: `https://bitbucket.org/tim_ruhe/dsea`

4.2.2 IACT Data Sets

The following presentation introduces two data sets which have been simulated for different IACTs. An overview of their common data analysis chain is given before the differences between the two data sets are identified.

Analysis Chain

The atmospheric processes involved in Cherenkov astronomy form an air shower from each initial γ particle, being already introduced in Chapter 1. An IACT records a raw video sequence from the Cherenkov light emitted by each shower. Each video sequence is then processed in the analysis steps presented in Figure 4.4.

The first step in this analysis chain is to calibrate each video sequence and to identify pixels that belong to the observed shower. In the subsequent step, features are extracted from each calibrated observation. These features represent the observations on a higher level of abstraction, which facilitates the following analysis steps. One popular choice in Cherenkov astronomy is to extract the Hillas parameters [28] of the observation, which describe geometrical features like the orientation and the size of a shower.

The third step discards observations that do not picture a γ particle. This step is mandatory because most of the observed showers are actually induced by other particles which do not come from the celestial object under study. Only for γ rays it is certain that an observed particle really originates from the monitored object. Any particle of another type may have a different origin and it therefore has to be excluded from the measurement. The separation of γ particle observations from other showers is usually performed with machine learning models trained on simulated data. Ideally, the separation results in a pure sample of γ observations, which are represented by their previously extracted features.

The last step of the analysis chain proposed by [1] is to estimate the energy of each individual γ particle. However, this step is usually omitted in deconvolution. Instead, the energy density of the entire γ sample is estimated, based on the output of the signal separation step. Deconvolution adopts each feature as one observable quantity.

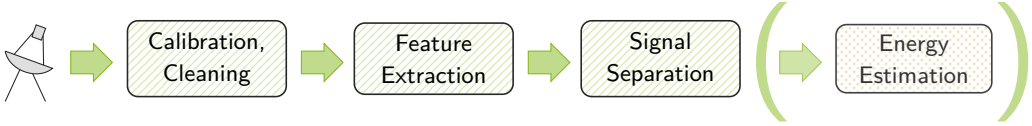


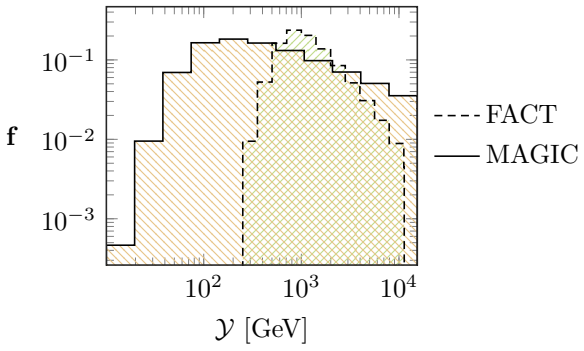
Figure 4.4: The analysis chain of an IACT calibrates the raw observations and represents them by their features. It then obtains a pure set of γ particle observations [1]. The last of these steps is usually not performed for each individual event. Instead, the energy density of the entire γ sample is estimated by deconvolution.

FACT and MAGIC

The First G-APD Cherenkov Telescope (FACT) [29] is the first IACT making use of G-APD photo-multiplier diodes. Unlike photo-multiplier tubes, which are used in other IACTs and which require a dark night sky background to observe γ radiation, these diodes can also be operated in moonlit nights. This robust design is particularly useful when a celestial γ ray source has to be monitored for several months straight. One of the major goals of the FACT telescope is therefore to monitor Active Galactic Nuclei, which have to be monitored continuously for long periods of time. However, FACT is a comparably small device, which only has a reflective surface of roughly 10m^2. Due to its modest size, FACT can only observe a rather small energy range.

The other IACT considered here is the Major Atmospheric Gamma Imaging Cherenkov (MAGIC) telescope [30, 31]. Actually, MAGIC is a stereoscopic system consisting of two independent telescopes. Its stereoscopic design allows the instrument to estimate certain properties of air showers with a particularly high precision. For example, the height of the first particle interaction is reconstructed more accurately with two telescopes, which can assess the distance of this first interaction more precisely than a single telescope can. Each of the two reflectors spans 17m in diameter, yielding a total reflective surface that is about 45 times higher than that of FACT. Due to this large surface, MAGIC is able to detect a much wider range of particle energies.

Figure 4.5 compares the two data sets that are simulated for FACT and MAGIC. It shows that the MAGIC telescope detects a much wider energy range than FACT, which is due to the different scales of the two instruments. Also, the slope of the density is higher in FACT than it is in MAGIC. Last but not least, the MAGIC data set consists of more examples than the FACT data set.



	FACT	MAGIC
γ set size	215,416 ex.	443,381 ex.
$\min(E)$	$10^{2.4}$ GeV	$10^{1.0}$ GeV
$\max(E)$	$10^{4.2}$ GeV	$10^{4.5}$ GeV

Figure 4.5: The two IACT data sets are compared with each other. The density **f** of γ particle energies spans a range that is broader for MAGIC than it is for FACT. Moreover, the MAGIC data contains more than twice as many γ particles as the FACT data set.

4.3 Evaluation Strategy

A fair and reliable comparison of deconvolution methods requires an appropriate evaluation strategy, which is presented in the following. This strategy assesses the quality of the deconvolution results repeatedly on different sub-samples of the data, allowing to compare the mean performance and the variance of the investigated methods. Bootstrapping is used here, for this purpose.

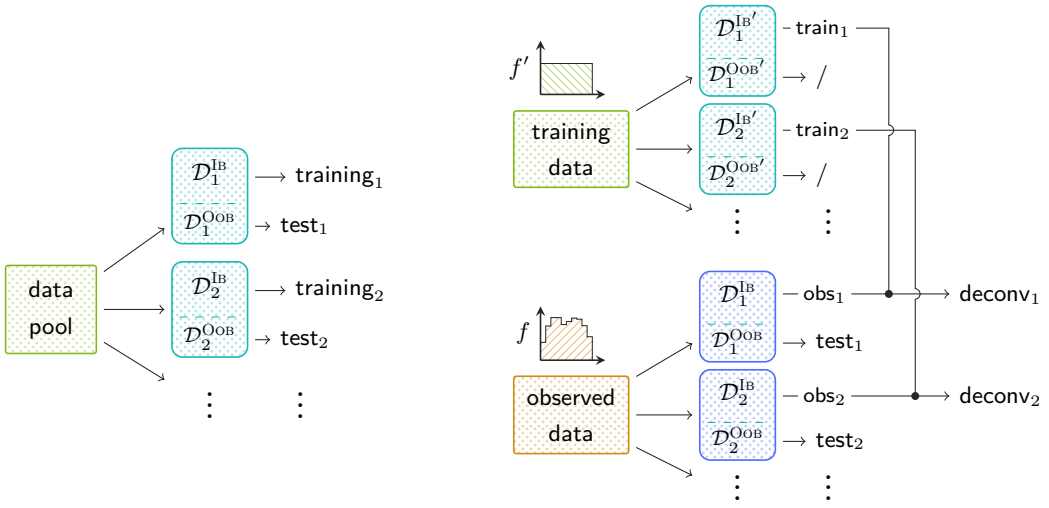
Since deconvolution methods should be robust to inappropriate training set densities, each experiment is conducted on two training sets, which have different target densities. Comparing the results obtained with these training sets allows to assess the effect of inappropriately distributed training data.

4.3.1 Bootstrapping

When evaluating a predictive method like a classifier or deconvolution method, one is typically interested in the accuracy exhibited in practice, on previously unseen data. This accuracy can only be assessed by using distinct sets of training and test data, the last of which is used for estimating the accuracy that the trained model exhibits on novel examples. However, a single split into training and test sets can produce varying results because usually, there are some splits that are easier to solve than others. This problem led to two major evaluation strategies for predictive methods, which are *bootstrapping* and *cross validation*. Both strategies repeat the split into training and test data and aggregate the accuracy of the method exhibited on all test sets. The two strategies primarily differ in the way they perform the split.

In each iteration b, bootstrapping randomly splits the complete pool of data into two distinct sets $\mathcal{D}_b^{\text{IB}}$ and $\mathcal{D}_b^{\text{OOB}}$, which are referred to as "in-bag" and "out-of-bag" data. When evaluating a classifier with a single data pool, these sets are used as the training and test set, as presented in Figure 4.6a. Each $\mathcal{D}_b^{\text{IB}}$ is obtained by drawing N examples *with replacement*, where N is also the size of the complete pool. Due to the replacement, some of the examples in the pool are drawn multiple times and others are not drawn at all. In contrast, each $\mathcal{D}_b^{\text{OOB}}$ consists of the examples which are not drawn in iteration b. Each $\mathcal{D}_b^{\text{OOB}}$ is therefore distinct from the corresponding $\mathcal{D}_b^{\text{IB}}$. The probability of an individual example to be in $\mathcal{D}_b^{\text{OOB}}$, i.e. to be drawn zero out of N times, is given in Equation 4.3. This probability is also the size of $\mathcal{D}_b^{\text{OOB}}$ relative to the entire data pool, because each example has the same chance to be in $\mathcal{D}_b^{\text{OOB}}$. The e_0 bootstrap used here estimates the true quality of a model by the mean quality over all bootstrap iterations.

$$\mathbb{P}\left(\text{example is in } \mathcal{D}_b^{\text{OOB}}\right) \quad = \quad \left(1 - \frac{1}{N}\right)^N \quad \overset{N\to\infty}{=} \quad \frac{1}{e} \quad \approx \quad 0.368 \tag{4.3}$$



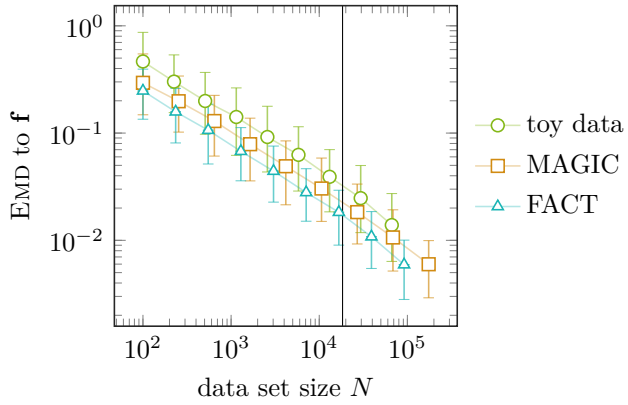
(a) Bootstrapping repeatedly splits a pool of data into B disjunctive "in-bag" and "out-of-bag" data sets $\mathcal{D}_b^{\text{IB}}$ and $\mathcal{D}_b^{\text{OOB}}$. Usually, a classifier is evaluated by using each set $\mathcal{D}_b^{\text{IB}}$ for the training and each corresponding $\mathcal{D}_b^{\text{OOB}}$ for validation. The quality values obtained in all iterations are aggregated then.

(b) Deconvolution deals with two data pools, which may follow different target densities. Consequently, both of these pools are bootstrapped separately. The in-bag sets of the training data and the observed data are combined in the deconvolution and each result is validated with the corresponding out-of-bag set of the observed data. The out-of-bag training sets are discarded here.

Figure 4.6: Bootstrapping is used to evaluate classifiers and deconvolution methods.

In deconvolution, there are two separate data pools, one for training examples and one for observations. These pools may have different sources and they may follow different densities of the target quantity. For instance, IACTs require a simulated training set because the true energy of a γ particle is only known for simulated data but never for examples actually observed with a telescope. The training set therefore comes from another source, the simulation, than the data to which the deconvolution is applied. Particularly, the observed data can follow another energy density than what is assumed for simulating the training set. When evaluating deconvolution methods on simulated data only, this situation is resembled by splitting the data into two distinct data pools. The toy data set is generated in two separate pools, already (see Subsection 4.2.1).

The two data pools are bootstrapped separately here, as presented in Figure 4.6b. Each iteration of this bootstrapping strategy combines an in-bag training set $\mathcal{D}_b^{\text{IB}'}$ with an observation set $\mathcal{D}_b^{\text{IB}}$, which is deconvolved. Note that the observable quantities of $\mathcal{D}_b^{\text{IB}}$ contribute to the deconvolution result. $\mathcal{D}_b^{\text{IB}}$ should therefore not be used to assess the deconvolution quality. Instead, the out-of-bag observation set $\mathcal{D}_b^{\text{OOB}}$ is used for this purpose. The quality of the results is thus assessed on previously unseen data, while the separation of the training pool and the observed data pool is maintained.



data set	EMD to **f**
toy data	$3.23 \cdot 10^{-02} \pm 1.44 \cdot 10^{-02}$
MAGIC	$2.25 \cdot 10^{-02} \pm 1.02 \cdot 10^{-02}$
FACT	$1.76 \cdot 10^{-02} \pm 6.93 \cdot 10^{-03}$

Figure 4.7: More examples provide a more appropriate true target density $\mathbf{f}_N$. Approximately $N = 18\,394$ examples (vertical line) are expected in each out-of-bag test set. The mean qualities and the standard deviations of $\mathbf{f}_N$ obtained with data sets of this size are presented in the table. These values correspond to the best quality obtainable with deconvolution, when only 18 394 examples are available to assess that quality.

In every experiment, the size of the observed data pool is fixed to $N = 50\,000$ examples, yielding an expected number of $(1 - \frac{1}{N})^N \approx 18\,394$ examples in $\mathcal{D}_b^{\text{OOB}}$. Figure 4.7 displays the error induced by using such finite pool for obtaining a "true" density $\mathbf{f}_N$. The y axis of this plot presents the distance of $\mathbf{f}_N$ to the density $\mathbf{f}$ that would be obtained with all available examples. The table in this figure presents the values at $N = 18\,394$, representing the lower bounds on the quality that is possible in the evaluation experiments. These bounds are only obtained with full knowledge about the target quantity in $\mathcal{D}_b^{\text{OOB}}$. The remaining error stems from the finite volume of the evaluation set alone.

Each experiment uses $B = 20$ bootstrap iterations. This number yields a reasonable run time and it provides 5% and 95% quantiles, which are used to plot the variability of each result.

4.3.2 Appropriate and Uniform Training Densities

In practice, the target density of the training set is usually different from the target density of the observed data, which is estimated by deconvolution. Therefore, deconvolution methods should be robust to inappropriate training set densities. In other words, the density of the training set should not bias the deconvolution result.

This property is checked here by conducting each experiment with two different training sets, one of which is appropriately distributed, and one of which follows a uniform target density. The first training set is referred to as "appropriate" because its target density is identical to the density of the observed data. In contrast, the uniform density of the other training set is quite different from the observed density and it is therefore

considered inappropriate. The robustness of deconvolution methods is assessed by comparing the results obtained with these two training sets. A method that is not robust will produce different results, each of which is biased towards the respective training density. In contrast, robust methods will produce results that are approximately equal for both training sets.

While the appropriate training set density is already inherent to the data, it is necessary to sub-sample each training pool in order to obtain a uniform training density. Table 4.1 presents the volume of the resulting training pools for FACT, MAGIC, and for the toy data. The size of the appropriate training set is the respective full volume containing all examples that remain after randomly taking 50 000 examples to obtain the test pool.Sub-sampling the uniform training data from this full volume reduces the number of examples available for training. This effect is particularly drastic for the FACT data, which has a large slope in the target density. The toy data has a basically arbitrary size because it is generated with negligible effort.

All experiments conducted here use at most $N' = 100\,000$ randomly chosen examples for training, which are enough to train sufficiently accurate models. A fair comparison between the two training sets is only possible with such limit. If otherwise all available examples were used, the appropriate training set would have an additional advantage from its large volume. This undesired advantage is later observed on the FACT data, which can only provide 9 048 examples for the training with a uniform density. Compared to all other experiments with $N' = 100\,000$, such training set is below 10% in volume. Consequently, poor estimates have to be expected for the uniform training density with FACT data.

data set	training density	training pool size
MAGIC	appropriate	393,381 ex.
MAGIC	uniform	126,900 ex.
FACT	appropriate	165,416 ex.
FACT	uniform	9,048 ex.
toy data	appropriate	arbitrary
toy data	uniform	arbitrary

Table 4.1: Sub-sampling the uniform prior reduces the number of training examples compared to the full training pool, from which the appropriate training set is sampled. The toy data is generated synthetically, so that arbitrary data set sizes can be obtained with negligible effort.

4.4 Model Selection

In order to find the best algorithm for deconvolution in the IACT domain, each deconvolution method is optimized by a grid search over a comprehensive grid of meta-parameters. The results of the best configurations are then compared to each other.

The dense parameter grid is presented in Table 4.2. Since preliminary experiments revealed that a target dimension $I < J$ does not produce appropriate results in RUN, the grid excludes expansion factors e that let the expanded target dimension $I' = e \cdot I$ exceed the observable dimension J. After omitting such configurations, the grid contains 1035 cells for the Toy data ($I = 21$) and 1053 cells for each of the IACT data sets ($I = 12$).

method	parameter	values
all	convergence threshold ϵ	$10^{-1}, 10^{-2}, \ldots, 10^{-9}$
DSEA$^+$	τ (regularization $\alpha_{\text{RUN}}^{(k)}$)	$10^{-3}, 10^{-4}, \ldots, 10^{-9}, 0$
IBU and RUN	number J of clusters	21, 84, 147, 210
IBU	smoothing order	$1, 2, \ldots, 9$, no smoothing
RUN	expansion factor e	1 (no expansion), $2, \ldots, 6$

Table 4.2: The grid of parameter values which is searched in the comparison experiment. DSEA and DSEA$^+$ employ a Naive Bayes classifier on the toy data and a Random Forest otherwise. Expansion factors are omitted if they result in a target dimension $I' = e \cdot I$ greater than the number J of clusters.

As already presented in the previous subsections, each configuration is applied to the three data sets, each time being trained on the two different training set densities. Each of these experiments is conducted in a 20-fold bootstrap, with which the variability of the deconvolution results is assessed. All in all, this comprehensive setup conducts 125 640 deconvolution runs.

4.4.1 Experimental Results

The best configuration of each method is selected with the mean in-bag quality of the deconvolution results. The observed out-of-bag sets thus remain unseen during the model selection. The out-of-bag quality values are only evaluated for the final quality assessment presented in Figure 4.8. This figure presents the best results obtained with the different methods on the three data sets, using the two different training densities. The horizontal lines represent the optimal values which are only obtained with full knowledge about the target quantity in the out-of-bag set. This knowledge is never available.

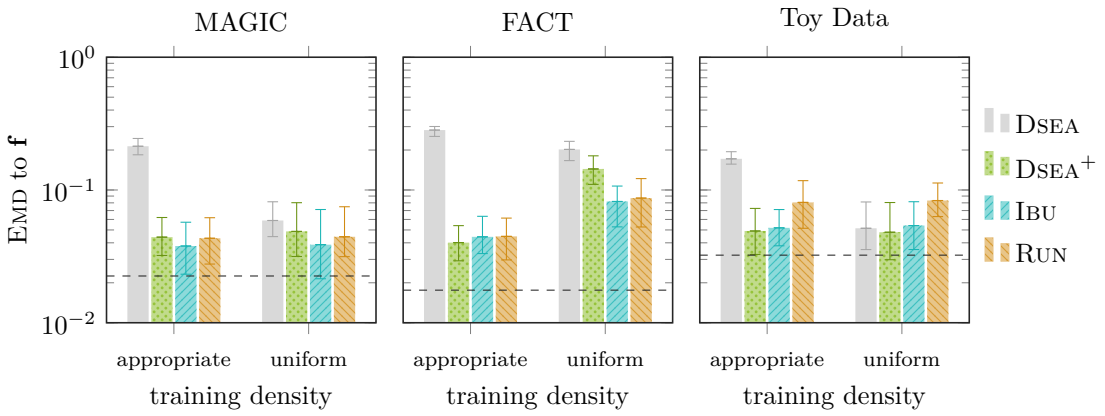


Figure 4.8: The accuracy of DSEA$^+$, IBU, and RUN is assessed with the Earth Mover's Distance between the deconvolution results and the corresponding true densities. A high value thus indicates a low quality of the respective result. Each bar is obtained with the best configuration of the corresponding method evaluated on 20 bootstrap samples. The horizontal lines present the optimal values that would only be obtained with full knowledge about the target quantity, which is never available (see Figure 4.7).

First of all, the extended DSEA$^+$ algorithm outperforms the original DSEA in each of the experiments. This improvement is in the order of one magnitude with respect to the EMD. DSEA$^+$ yields an accuracy that is about equal to the quality of the classical approaches IBU and RUN. Moreover, the variability of the results is about equal between IBU, RUN, and Dsea$^+$. The minor differences between the mean quality values stay within the error bars. Thus, IBU, RUN, and DSEA$^+$ are capable of producing results of equal accuracy. Figure 4.9 shows the similarity of the estimates obtained with the different methods.

IBU, RUN, and DSEA$^+$ are robust to the inappropriate training set density. Namely, each method performs equally well on both considered densities. Exceptionally, all of these methods are inaccurate with the uniform density on the FACT data set.

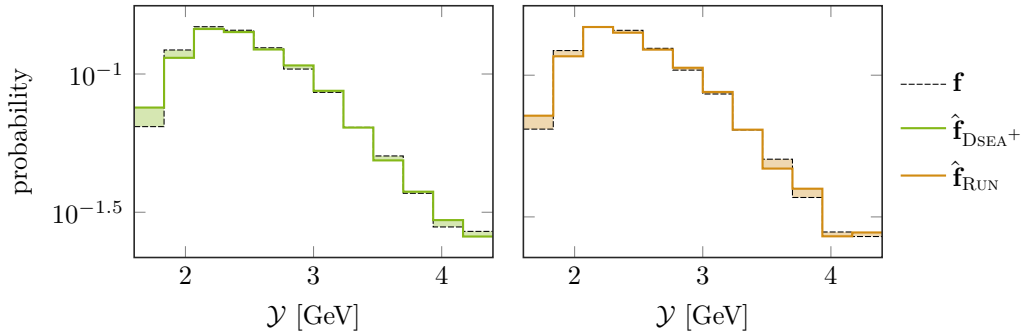


Figure 4.9: The deconvolution results of DSEA$^+$ (left) and RUN (right) are remarkably similar on the uniform training density of the MAGIC data set. This observation extends to the other methods, data sets, and training set densities.

The reason for the extraordinarily low accuracy on the uniform density with FACT is the small size of the corresponding training set. Due to the uniform sub-sampling, only about 10% of the data is available for training, compared to all other experiments (see Table 4.1). DSEA$^+$ suffers most from this small volume. Apparently, more data is required for reliably fitting the embedded classifier than for directly fitting the target density with maximum likelihood (RUN) or with Bayes' rule (IBU).

4.4.2 Influence of the Meta-Parameters

The meta-parameter grid employed for model selection allows us to assess the influence of the individual parameters. This is done for DSEA$^+$, IBU, and RUN. The non-adaptive step size strategies proposed for DSEA are also investigated.

DSEA$^+$ is configured with the regularization strength τ used in the step size adaption and with the convergence threshold ϵ. Figure 4.10 fixes ϵ to two different values, thus focusing on the influence of τ. It becomes apparent, that equally accurate results are obtained with all regularization strengths. Consequently, maximizing the likelihood of the estimate in the search direction of DSEA$^+$ does not require regularization at all. Moreover, the influence of the convergence threshold is small, as well. DSEA$^+$ can thus be applied "safely" without extensively optimizing these meta-parameters. Namely, optimizing them will not improve the results significantly, compared to the default parameter values, which already produce accurate results.

The effect of the different meta-parameter values in IBU and RUN is investigated in Figure 4.11. One can see that both methods are only marginally influenced by the number J of observable clusters, even though a large value of J yields slightly better results than low values. This effect is observed for all smoothing orders and regularization strengths.

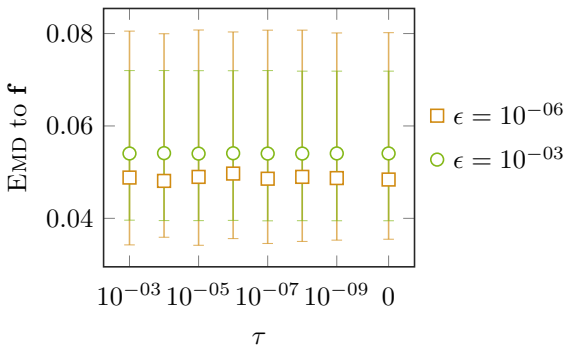


Figure 4.10: The accuracy of DSEA$^+$ is almost constant with respect to the regularization strength τ. Moreover, the convergence threshold ϵ only exhibits a small influence on the accuracy. This observation is made here with the uniform training density on MAGIC data. It perfectly extends to the other data sets and training set densities.

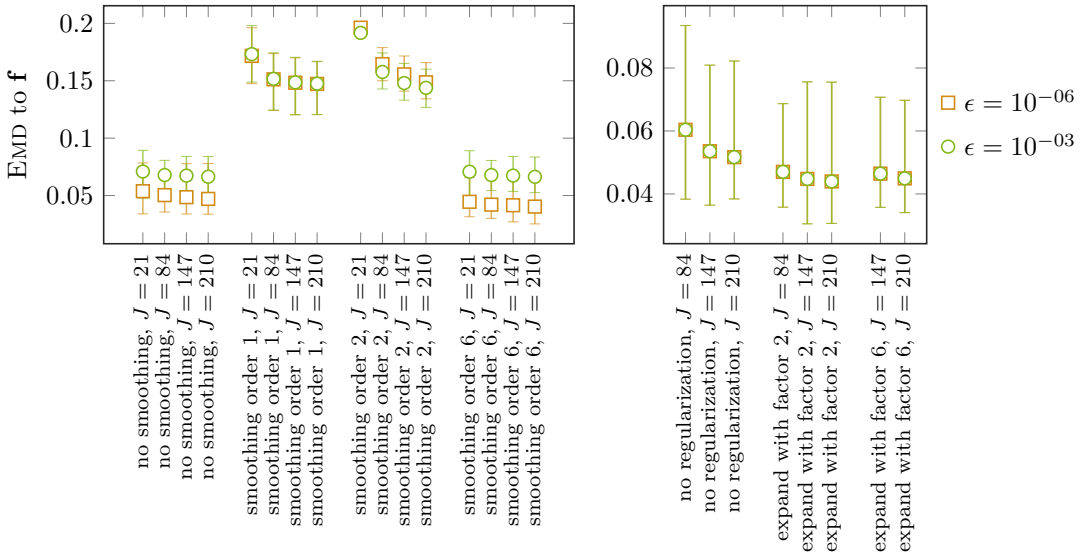


Figure 4.11: The influence of meta-parameters is assessed for IBU (left) and RUN (right). For IBU, these parameters are the order of the polynomial smoothing and the number J of observable clusters. Apparently, smoothing can only improve the result marginally and only if a high smoothing order is chosen. If no smoothing is applied, the choice of J only has a small influence. RUN works best with a large expansion factor, i.e. with a high amount of regularization.

Surprisingly, the smoothing of IBU exhibits a negative impact on the accuracy. Only high polynomial orders perform well in the experiments conducted here. However, these high orders contradict the intention behind the smoothing, which is to approximate the prior to regularize the estimates. In fact, the target densities estimated here are not accurately reproduced by low-order polynomials. A smoothing with low-order polynomials consequently regularizes the estimates towards mistaken prior assumptions. Apparently, smoothing is only beneficial with an analytic form that represents an appropriate prior.

RUN is configured by the expansion factor, which determines the strength of the regularization. High factors, corresponding to strong regularization, generally perform best in the experiments conducted here. However, the improvement is rather small.

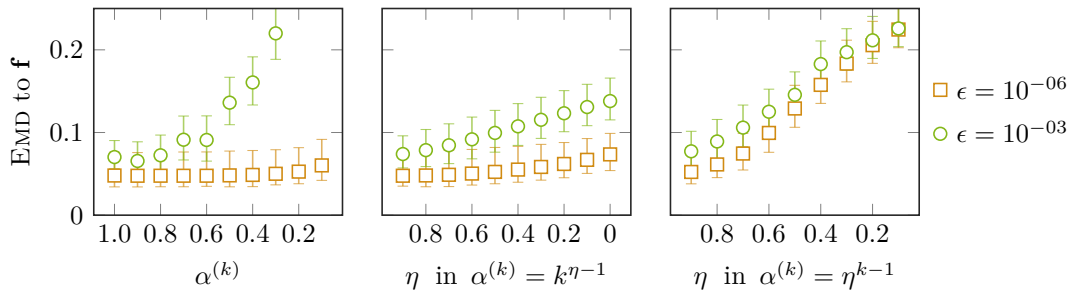


Figure 4.12: Small constant step sizes and fast decay rates (for which η is small) require small convergence thresholds ϵ in order to produce accurate results. This observation is made on all non-adaptive step-size strategies.

Figure 4.12 displays the influence of different constant step sizes and decay rates for the non-adaptive step size strategies in DSEA$^+$. Even though these strategies are not considered in the comparison from the previous subsection, they are also optimized over a meta-parameter grid in order to assess the influence of their parameters. The figure shows that small step sizes—either constantly small or rapidly decaying—require small convergence thresholds in order to produce accurate results.

Naturally, small thresholds require many iterations in the non-adaptive step size strategies. This expectation is validated in Figure 4.13, where the required iteration numbers are plotted against the convergence threshold. While many iterations are required by the constant step sizes, it can be seen that even the smallest thresholds are reasonable with the adaptive step size. With this adaptive strategy, there is no additional iteration after $\epsilon = 10^{-6}$ is reached. Since a classifier has to be trained in each iteration of DSEA / DSEA$^+$, a considerable run time has to be expected for every single iteration. Therefore, only a reasonably small number of iterations should be performed. The adaptive step size from DSEA$^+$ meets this goal by providing accurate estimates after a few iterations.

Summarizing the experimental results, one can see that DSEA$^+$ produces accurate estimates without the need for an extensive meta-parameter tuning. In fact, it performs well with all reasonable regularization strengths and convergence thresholds. This property is great for the applied deconvolution in the field, where the resources for meta-parameter tuning may not be available. In contrast, IBU and RUN produce insufficient results, if they are not appropriately parametrized.

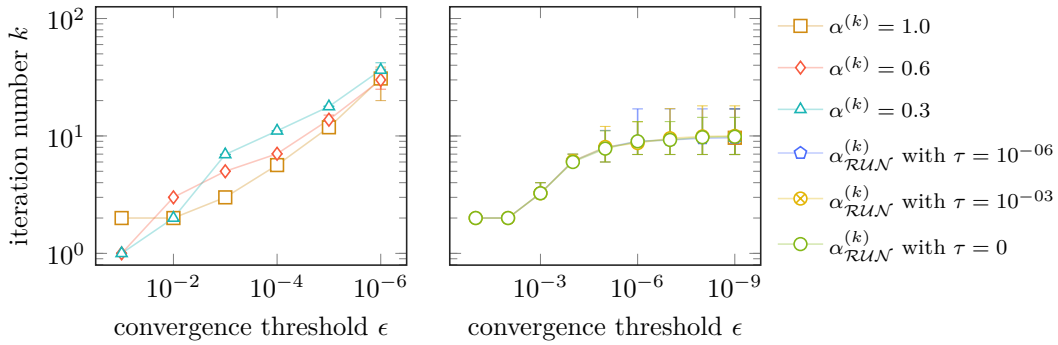


Figure 4.13: The number of iterations required for attaining the different convergence thresholds is presented for different constant step sizes (left) and for the adaptive step size in DSEA$^+$ (right). While the number of iterations constantly increases for the constant step sizes, there is a saturation in the right plot. The adaptive strategy thus converges fully after $\epsilon = 10^{-6}$ is attained. Also note that the two decaying step size strategies increase the number of iterations, similar to the constant step size presented in the left plot. This experiment is conducted on the uniform training density on MAGIC data, but it perfectly extends to the other densities and data sets.

Chapter 5

Conclusion

The present work surveyed existing deconvolution methods, on which a novel unified view has been established from the perspective of machine learning. The suitability of these methods for Cherenkov astronomy has been investigated quantitatively, in a comprehensive benchmark. One of the considered methods is DSEA, which employs machine learning to train an embedded classifier. The original version of this algorithm suffers from divergence, but the present thesis proposed the improved version DSEA$^+$. This improved version does not diverge and its accuracy matches the state of the art in deconvolution. For Cherenkov astronomy, DSEA$^+$ and both methods from the state of the art are equally suited.

5.1 Contributions and Findings

The novel unified view on deconvolution rephrases the problem in the language of machine learning. Within that view, essential building blocks of the existing methods have been identified. These blocks can now be studied independently. Moreover, they are exchangeable between the methods, yielding numerous possibilities for recombining them in new algorithms. In contrast, existing studies have treated each deconvolution method as a single monolithic instance, which does not open the problem to these directions of future work.

The machine learning perspective already lead to the proof that DSEA/DSEA$^+$ and the Iterative Bayesian Unfolding are equivalent, if the DSEA classifier is accordingly chosen. This proof is a result of the present work and it has not been published before.

Even though the original DSEA is already a promising approach, it suffers from divergence. Here, this concern was tracked down to an inappropriate re-weighting of the training examples. Resolving this issue lead to the novel algorithm DSEA$^+$, which does not diverge. Another aspect of DSEA$^+$ is an adaptive step size, which approaches the optimal

solution faster than the original Dsea. The adaptive step size also recognizes when to stop the iterative procedure, thus saving resources by not wasting iterations.

The experimental comparison of Dsea$^+$, the Iterative Bayesian Unfolding (Ibu), and the Regularized Unfolding (Run) is the first evaluation of deconvolution methods that employs different training set densities and a separate bootstrapping of the data pools. Moreover, it is the first evaluation that assesses the quality of deconvolution results in terms of the Earth Mover's Distance, which is well motivated for this purpose. Thus, the evaluation setup is exceptionally thorough. All results are fully reproducible with the published Julia code, which is well documented and extensible with respect to the implemented methods and the considered data sets.

The comprehensive evaluation revealed that the accuracy of Dsea$^+$ matches the accuracy of Ibu and Run, which represent the state of the art in deconvolution. This achievement is highly valuable for Cherenkov astronomy, because Dsea$^+$ is the only one among these algorithms which returns the contribution of each individual observation to the deconvolution result. This feature is required for a time-dependent deconvolution, which is outlined in the following section.

For deconvolution without time dependency, the evaluation showed that Dsea$^+$, Ibu, and Run are equally suitable because they provide comparable accuracies on IACT data. Moreover, they are robust to inappropriate priors of the target density. This robustness is important for the reliability of deconvolution results. It was shown that the original Dsea does not accomplish these qualities.

5.2 Outlook

Future work on deconvolution will be based on the novel unified view on the subject. This view enables researchers to study the building blocks of existing algorithms separately. Moreover, new deconvolution methods can be developed by recombining these blocks. Some ideas in this direction are outlined in Subsection 5.2.1.

Other directions of future work will leverage the contributions of individual observations to the deconvolution result, which are only obtained with Dsea/Dsea$^+$. Subsections 5.2.2 and 5.2.3 sketch two of these directions. The first direction aims at time series analyses, which detect concept shifts/drifts in a time-dependent deconvolution result. Such result is obtained by aggregating over a sliding window of individual contributions. The second direction employs these contributions to control the simulation that produces the training data. It aims at saving resources while improving the deconvolution results. Both directions are difficult to pursue with classical approaches, which do not return the contributions of individual observations.

5.2.1 Additional Approaches and Re-combinations

The present thesis presented IBU and RUN as two popular representatives of classical deconvolution. However, more classical approaches to deconvolution exist, some of which have already been mentioned here. Investigating these approaches in more detail can single out additional building blocks of deconvolution methods. Particularly, the approach based on neural networks should be studied in the unified view developed here. Neural nets gained a lot of attention in the past years, due to their performance in image classification and other modeling tasks. The rapid developments in this area are promising directions for future work on deconvolution.

Having identified the essential building blocks of the existing deconvolution methods, researchers are able to recombine these blocks in new algorithms. One example of such recombination was already given here with the objective function of RUN being employed for the adaptive step size in DSEA$^+$. The possibilities of other re-combinations are numerous. One particular task of future work is to identify which building blocks work for which kind of input data.

5.2.2 DSEA for Time Series Analyses

Some γ ray sources studied in Cherenkov astronomy change their emission over time. Consequently, these sources are only modeled accurately with a deconvolution that takes their time dependency into account. Indeed, the FACT telescope is specifically targeted at measuring changes in the energy emission, thus rendering time-dependent deconvolution indispensable.

DSEA/DSEA$^+$ reconstructs the target density from the confidence values predicted for individual observations. Instead of reconstructing the density of the entire data set,

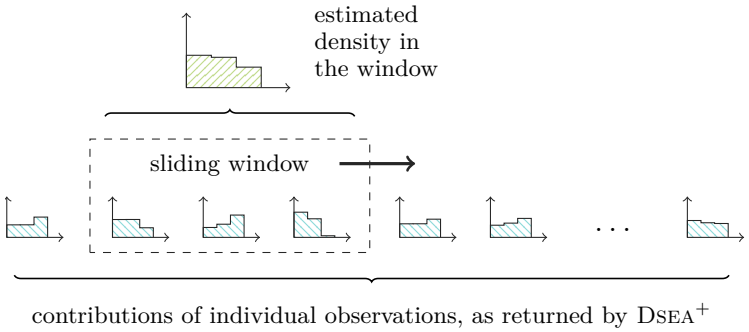


Figure 5.1: Time-dependent deconvolution is performed in a sliding window, which aggregates the contributions of individual observations. This window moves over the contributions, which are ordered by the time of their observation. Each position of the window yields one estimated density for this position. The individual contributions are not obtained with classical deconvolution methods – only DSEA/DSEA$^+$ provides this feature.

a practitioner can aggregate the contributions of individual observations over a subset of the data. If this subset is time-dependent, he obtains a deconvolution result that reflects this time dependency. Figure 5.1 sketches such time-dependent aggregation over a sliding window. First, DSEA$^+$ deconvolves the entire data set, as usual. The embedded classifier is then queried for the contribution of each individual item to the deconvolution result. Instead of aggregating all of these contributions, only those in a sliding window are aggregated, according to the usual reconstruction rule. Each position of the sliding window thus yields an estimated density. The time-dependent changes of the observed data set are reflected in the changes of these estimates.

Figure 5.2 presents a preliminary experiment on time-dependent deconvolution with DSEA$^+$. In this experiment, a concept shift is generated artificially by arranging two subsets of the MAGIC data, one with low energy values and one with high energy values. In the beginning, only low particle energies are observed. Suddenly, the concept changes and only high energy values follow. DSEA$^+$ deconvolves both groups together, returning the contribution of each observation in the entire data set. In fact, it does not know about the two artificial groups. The returned contributions are then aggregated in each position of a sliding window, which moves over the ordered contributions. For simplicity, each reconstructed density is represented only by its mode, i.e. the position of its peak. This characteristic feature is plotted as the univariate time series presented in Figure 5.2. It clearly shows the concept shift that was artificially generated and standard methods can detect this shift automatically. This result promises that time-dependent deconvolution can detect concept shifts and concept drifts reliably. Future work has to validate this promise for Cherenkov astronomy and other areas of research.

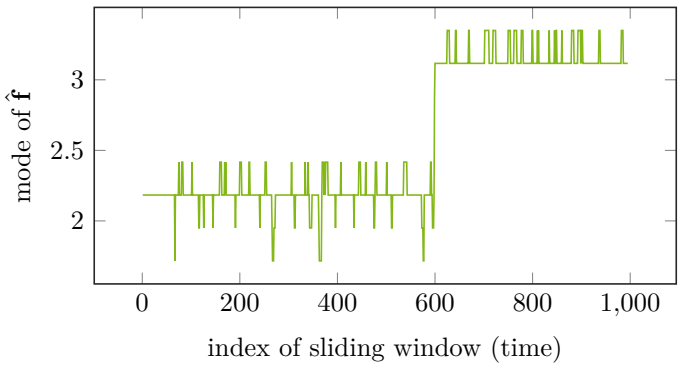


Figure 5.2: The mode of a time-dependent deconvolution result is studied as a univariate time series. Clearly visible is the concept shift that is artificially introduced by arranging two groups of observations. Standard methods can automatically detect the concept shift from this time series representation.

5.2.3 DSEA for the Smart Control of Monte-Carlo Simulations

Simulating the training data for IACTs requires the execution of the fine-grained stochastic processes involved in air showers. Such execution is costly in terms of run time and energy consumption. The smart control of Monte-Carlo simulations [32] aims at saving these resources during the generation of training data while improving the accuracy of the models that are fitted to the simulation output. The concept is based on the idea that a simulation s is a process which produces training examples $(\mathbf{x}, y) \in \mathcal{X} \times \mathcal{Y}$ from simulation parameters $\mathbf{p} \in \mathcal{P}$. This process is controlled by selecting those inputs $\mathbf{p}$ which produce relevant examples for training.

$$s : \mathcal{P} \to \mathcal{X} \times \mathcal{Y} \tag{5.1}$$

Usually, a fixed distribution over $\mathcal{P}$ is assumed to sample simulation inputs from this space. The simulation is run on a single batch of these inputs, producing a fixed set $\mathcal{D}_{\text{train}}$ of training examples. The smart control of simulations generates examples in multiple small batches $P \subset \mathcal{P}$, introducing a feedback loop that evaluates the latest model to select additional inputs for the next simulation run. Figure 5.3 presents this concept. It relates to Active Class Selection [33], which employs a similar feedback loop for selecting the labels for which additional observations are required.

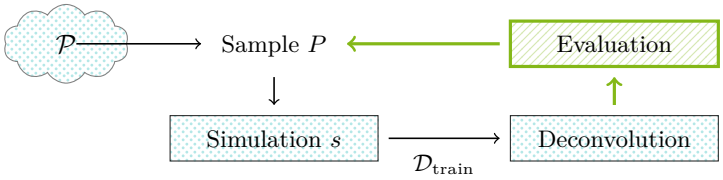


Figure 5.3: The smart control of simulations [32] is adapted to the deconvolution problem. Each run of the simulation is parametrized by an input batch $P \subset \mathcal{P}$ of parameters, from which it produces additional training data $\mathcal{D}_{\text{train}}$ for the deconvolution algorithm. The input choice of the next iteration is based on the evaluation of the current estimate.

Figure 5.4 presents a first experiment, in which DSEA$^+$ is put in control of the example generation. For simplicity, the toy data set is used here and $\mathcal{P}$ is identified with $\mathcal{Y}$, i.e. labels are chosen for which examples are generated (just like in Active Class Selection). The baseline of this experiment is the uniform sampling, which imposes a uniform prior over $\mathcal{P}$. The smart control strategy leverages the latest estimate of DSEA$^+$, generating training data that follows the distribution of the observed data set. This estimate is weighted by the *redistriction* strategy [33, 32], which favors labels for which many predictions changed during the last iteration. Such weighting is only possible with DSEA$^+$, from which the embedded classifier is evaluated to produce the weights. The experiment suggests that the smart control produces better deconvolution results than the uniform baseline. Validating this observation remains for future work.

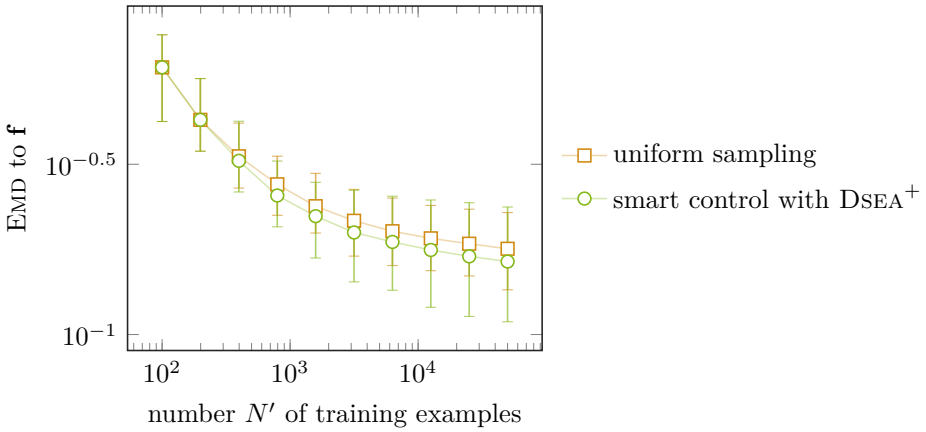


Figure 5.4: The deconvolution result improves as more training data is generated. The uniform strategy imposes a uniform prior over the parameter space $\mathcal{P}$ of the simulation. Since this strategy does not adapt to the deconvolution result, it is used as the baseline of the experiment. The second strategy aims at improving the deconvolution result by evaluating the embedded classifier of DSEA$^+$ on a separate validation set. Apparently, this strategy succeeds by slightly outperforming the baseline.

Appendix A

Further Information on Classical Deconvolution

Some additional aspects on classical deconvolution methods are collected here. First of all, Section A.1 relates the conventional notation from particle physics to the notation employed throughout this thesis. Subsequently, the B-spline discretization originally proposed for RUN is described in Section A.2. However, this discretization scheme is abandoned in the experiments conducted here. Finally, Section A.3 provides proof for the theorems that motivate the RUN algorithm.

A.1 Notation from Particle Physics

Most publications on deconvolution in particle physics refer to the early work of Blobel [5], which outlines the deconvolution problem. By adapting the notation from this work, they follow the conventional notation from the field [3, 2, 34, 6, 7, 22, 20]. Some slight differences remain between the publications, but mapping their notation to Blobel's work is straight-forward.

The usual notation of the deconvolution problem is presented in Equation A.1. Structurally, it is rather similar to the notation established in the present thesis. However, as already pointed out, the roles of x and y are switched here. The usual notation represents the values of the relevant quantity with x and the measured values with y. In contrast, the present thesis chose y for the target quantity to comply with machine learning.

$$g(y) \;=\; \int_a^b A(y,x)\, f(x)\; \mathrm{d}x \;+\; b(y) \tag{A.1}$$

There are more subtle differences between the conventional notation from particle physics and the notation established in the present work:

- The interval in Equation A.1 is bounded by scalars a and b, which limit the notation (strictly speaking) to the univariate case. Integrating over the pre-image $\mathcal{Y}$ of the target density is thus more comprehensive. Some publications on deconvolution omit the range of the integral, implicitly assuming a univariate pre-image.
- g and f of the conventional notation represent *frequency distributions* instead of densities. Thus, the entire system is scaled with the number of observed examples. This subtlety has to be considered during the maximum likelihood fit, which only models the likelihood appropriately with absolute numbers of examples. In other methods, like IBU, the scale of the system does not make a difference. In the present thesis, g and f represent probability density functions in order to relate the deconvolution problem to probability theory.
- The additional summand $b(y)$ represents the known (or previously estimated) background noise of the measurement at the observed value y. This background is simply subtracted from the solution of the difficult integral part of the equation. Thus, the background suppression is a trivial post-processing step in deconvolution, which is omitted here due to its simplicity and separate applicability.

It may have been noticed that the present work favors the term "deconvolution" over the term "unfolding", which is the more conventional term in particle physics. This choice has been made because "convolution" is the appropriate English term for the operation to be inverted. The term "folding" is less common in English, rather being a direct translation of the German term used for this operation.

A.2 B-Splines in the Regularized Unfolding

B-Splines are the original discretization scheme of the RUN algorithm [5]. However, this scheme is abandoned in the most recent publication on the method [2], which favors equidistant histograms. Here, the B-Spline discretization is only given for completeness. It is omitted in the experiments because the equidistant discretization is better suited for comparing the considered deconvolution methods with each other.

Generally, RUN discretizes the target density $f : \mathcal{Y} \to \mathbb{R}$ with a set of I orthogonal basis functions $p_i : \mathcal{Y} \to \mathbb{R}$, which are fixed. In order to represent f from these basis functions, the factors a_i are fit to a set of data, according to Equation A.2. For example, simple histograms match this general notion, being defined by $p_i(y) = 1$ if $\mathcal{Y}_{i-1} \leq y < \mathcal{Y}_i$ and $p_i(y) = 0$ otherwise. Plugging Equation A.2 into the deconvolution problem yields Equation A.3, where the solution is represented by the factors a_i [5].

$$f(y) \;=\; \sum_{i=1}^{I} a_i \, p_i(y) \tag{A.2}$$

$$\begin{aligned} g(x) &= \sum_{i=1}^{I} a_i \, R_i(x) \\ \text{where} \quad R_i(x) &= \int_{\mathcal{Y}} R(x \,|\, y) \cdot p_i(y) \; \mathrm{d}y \end{aligned} \tag{A.3}$$

The discrete deconvolution problem is thus altered according to Equation A.4, where the vector $\mathbf{a} = (a_1, a_2, \ldots, a_I)$ contains the factors which determine the solution to this problem. The matrix $\mathbf{R}'$ is estimated similar to $\mathbf{R}$ from Subsection 2.2.1, i.e. from relative frequencies in the training data set. However, each training example with the target value y is weighted by $p_i(y)$ when considered in the i-th matrix column. Thus, $\mathbf{R}'$ is a weighted variant of $\mathbf{R}$, the weights being determined by the basis functions. An estimate of f is obtained from Equation A.2, plugging in the the vector $\hat{\mathbf{a}}$ which is estimated from Equation A.4.

$$\mathbf{g} = \mathbf{R}' \, \mathbf{a} \tag{A.4}$$

Cubic B-Splines are one kind of orthogonal basis functions, which is chosen in the original version of RUN. These functions, $p_i(y) = B_{i,4}(y)$, are defined recursively by Equation A.5, where t_i is one of the I knots. k is referred to as the *order* of a spline. B-Splines smoothly interpolate between the given knots.

$$\begin{aligned} B_{i,k}(y) &= \frac{y - t_i}{t_{i+k-1} - t_i} B_{i,k-1}(y) + \frac{t_{i+k} - y}{t_{i+k} - t_{i+1}} B_{i+1,k-1}(y) \\ B_{i,1}(y) &= \begin{cases} 1 & t_i \leq y < t_{i+1} \\ 0 & \text{otherwise} \end{cases} \end{aligned} \tag{A.5}$$

A.3 Proofs for the Regularized Unfolding

The proofs of the theorems from Section 2.4 are presented in the following. These theorems motivate the RUN algorithm.

Proof of Theorem 2.1 The loss function ℓ is obtained by plugging the two assumptions of the RUN algorithm into the likelihood function $\mathbb{L}$. Terms that are constant with respect to $\mathbf{f}$ are omitted because they do not influence the maximization of $\mathbb{L}$.

1) Minimizing the negative log-likelihood is equivalent to maximizing $\mathbb{L}$:

$$\begin{aligned} \hat{\mathbf{f}}_{\max \mathbb{L}} &= \arg\max_{\mathbf{f}} \; \mathbb{L}(\mathbf{f} \,|\, \mathbf{g}) \\ &= \arg\min_{\mathbf{f}} \; -\ln \mathbb{L}(\mathbf{f} \,|\, \mathbf{g}) \\ &= \arg\min_{\mathbf{f}} \; -\ln \mathbb{P}\,(\mathbf{g} \,|\, \mathbf{f}) \end{aligned}$$

2) RUN assumes independent components of the observed density and that the absolute number $\bar{\mathbf{g}}_j$ in each component is Poisson-distributed with the rate $\mathbf{R}_{\bullet j}^T\mathbf{f}$. These assumptions are plugged into the likelihood function:

$$\begin{aligned}
-\ln \mathbb{P}\left(\mathbf{g} \,|\, \mathbf{f}\right) &= -\ln \prod_{j=1}^{J} \mathbb{P}\left(\mathbf{g}_j \,|\, \mathbf{f}\right) \\
&= -\sum_{j=1}^{J} \ln \mathbb{P}\left(\mathbf{g}_j \,|\, \mathbf{f}\right) \\
&= -\sum_{j=1}^{J} \ln e^{-\left(\mathbf{R}_{\bullet j}^T\mathbf{f}\right)} \frac{\left(\mathbf{R}_{\bullet j}^T\mathbf{f}\right)^{\bar{\mathbf{g}}_j}}{\bar{\mathbf{g}}_j!} \\
&= -\sum_{j=1}^{J} -\mathbf{R}_{\bullet j}^T\mathbf{f} + \bar{\mathbf{g}}_j \ln\left(\mathbf{R}_{\bullet j}^T\mathbf{f}\right) - \ln(\bar{\mathbf{g}}_j!)
\end{aligned}$$

3) The term $\ln(\bar{\mathbf{g}}_j!)$ is constant with respect to $\mathbf{f}$ and can thus be omitted in the minimization, yielding the loss function ℓ. Minimizing this function is therefore equivalent to maximizing the likelihood.

$$\begin{aligned}
\hat{\mathbf{f}}_{\max \mathbb{L}} &= \arg\min_{\mathbf{f}} \; -\ln \mathbb{P}\left(\mathbf{g} \,|\, \mathbf{f}\right) \\
&= \arg\min_{\mathbf{f}} \; -\sum_{j=1}^{J} \bar{\mathbf{g}}_j \ln\left(\mathbf{R}_{\bullet j}^T\mathbf{f}\right) - \mathbf{R}_{\bullet j}^T\mathbf{f} \quad = \quad \arg\min_{\mathbf{f}} \; \ell(\mathbf{f})
\end{aligned}$$

□

Proof of Theorem 2.2 The surrogate form follows directly from the given definitions. We start from the regularized local model which is minimized in each Newton iteration:

$$\hat{\ell}_r^{(k)}(\mathbf{f}) \;=\; \frac{1}{2}\,\mathbf{f}^T\mathbf{H}\,\mathbf{f} \;-\; \mathbf{f}^T(\mathbf{H}\,\hat{\mathbf{f}}^{(k)} - \boldsymbol{\ell}) \;+\; r(\mathbf{f})$$

1) The Hessian is eigen-decomposed into $\mathbf{H} = \mathbf{U}\mathbf{D}\mathbf{U}^T$. The matrix $\mathbf{U}$ is thus orthogonal, i.e. $\mathbf{U}^T\mathbf{U} = I$. Moreover, let $\mathbf{D}^{-\frac{1}{2}} = diag(\frac{1}{\sqrt{\mathbf{D}_{11}}}, \dots \frac{1}{\sqrt{\mathbf{D}_{II}}})$. The first transformation is $\mathbf{f} \;=\; \mathbf{U}\mathbf{D}^{-\frac{1}{2}}\mathbf{f}' \;\Leftrightarrow\; \mathbf{f}' \;=\; \mathbf{D}^{\frac{1}{2}}\mathbf{U}^T\mathbf{f}$.

$$\Rightarrow \quad \hat{\ell}_r^{(k)}(\mathbf{f}) \;=\; \frac{1}{2}(\mathbf{U}\mathbf{D}^{-\frac{1}{2}}\mathbf{f}')^T\mathbf{H}(\mathbf{U}\mathbf{D}^{-\frac{1}{2}}\mathbf{f}') \;-\; (\mathbf{U}\mathbf{D}^{-\frac{1}{2}}\mathbf{f}')^T(\mathbf{H}\mathbf{f}^{(k)} - \boldsymbol{\ell}) \;+\; r(\mathbf{f})$$

$$\begin{aligned}
(\mathbf{U}\mathbf{D}^{-\frac{1}{2}}\mathbf{f}')^T\mathbf{H}(\mathbf{U}\mathbf{D}^{-\frac{1}{2}}\mathbf{f}') &= (\mathbf{U}\mathbf{D}^{-\frac{1}{2}}\mathbf{f}')^T\mathbf{U}\mathbf{D}\mathbf{U}^T(\mathbf{U}\mathbf{D}^{-\frac{1}{2}}\mathbf{f}') \\
&= (\mathbf{U}\mathbf{D}^{-\frac{1}{2}}\mathbf{f}')^T\mathbf{U}\mathbf{D}(\mathbf{D}^{-\frac{1}{2}}\mathbf{f}') \\
&= (\mathbf{U}\mathbf{D}^{-\frac{1}{2}}\mathbf{f}')^T\mathbf{U}\mathbf{D}^{\frac{1}{2}}\mathbf{f}' \\
&= (\mathbf{f}'^T\mathbf{D}^{-\frac{1}{2}^T}\mathbf{U}^T)\mathbf{U}\mathbf{D}^{\frac{1}{2}}\mathbf{f}' \\
&= (\mathbf{f}'^T\mathbf{D}^{-\frac{1}{2}}\mathbf{U}^T)\mathbf{U}\mathbf{D}^{\frac{1}{2}}\mathbf{f}' \\
&= (\mathbf{f}'^T\mathbf{D}^{-\frac{1}{2}})\mathbf{D}^{\frac{1}{2}}\mathbf{f}' \quad = \quad \mathbf{f}'^T\mathbf{f}'
\end{aligned}$$

$$
\begin{aligned}
r(\mathbf{f}) \;=\; \frac{\tau}{2}\mathbf{f}^T\mathbf{C}\mathbf{f} \;&=\; \frac{\tau}{2}(\mathbf{U}\mathbf{D}^{-\frac{1}{2}}\mathbf{f}')^T\mathbf{C}(\mathbf{U}\mathbf{D}^{-\frac{1}{2}}\mathbf{f}') \\
&=\; \frac{\tau}{2}\mathbf{f}'^T(\mathbf{D}^{-\frac{1}{2}}\mathbf{U}^T\mathbf{C}\mathbf{U}\mathbf{D}^{-\frac{1}{2}})\mathbf{f}' \\
&=\; \frac{\tau}{2}\mathbf{f}'^T\mathbf{C}'\mathbf{f}'
\end{aligned}
$$

$$
\Rightarrow\;\; \hat{\ell}_r^{(k)}(\mathbf{f}) \;=\; \frac{1}{2}\mathbf{f}'^T\mathbf{f}' \;-\; (\mathbf{U}\mathbf{D}^{-\frac{1}{2}}\mathbf{f}')^T(\mathbf{H}\mathbf{f}^{(k)} - \boldsymbol{\ell}) \;+\; \frac{\tau}{2}\mathbf{f}'^T\mathbf{C}'\mathbf{f}'
$$

2) The second transformation is based on the eigen-decomposition of $\mathbf{C}' = \mathbf{U}'\mathbf{D}'\mathbf{U}'^T$. It is defined as $\mathbf{f}' \;=\; \mathbf{U}'\mathbf{f}'' \;\Leftrightarrow\; \mathbf{f}'' \;=\; \mathbf{U}'^T\mathbf{f}'$. Note that $\mathbf{U}'^T\mathbf{U}' = I$.

$$
\begin{aligned}
\hat{\ell}_r^{(k)}(\mathbf{f}) \;&= \frac{1}{2}(\mathbf{U}'\mathbf{f}'')^T(\mathbf{U}'\mathbf{f}'') - (\mathbf{U}\mathbf{D}^{-\frac{1}{2}}\mathbf{U}'\mathbf{f}'')^T(\mathbf{H}\mathbf{f}^{(k)} - \boldsymbol{\ell}) + \frac{\tau}{2}(\mathbf{U}'\mathbf{f}'')^T\mathbf{C}'(\mathbf{U}'\mathbf{f}'') \\
&= \frac{1}{2}\mathbf{f}''^T\mathbf{f}'' \;-\; (\mathbf{U}\mathbf{D}^{-\frac{1}{2}}\mathbf{U}'\mathbf{f}'')^T(\mathbf{H}\mathbf{f}^{(k)} - \boldsymbol{\ell}) \;+\; \frac{\tau}{2}(\mathbf{U}'\mathbf{f}'')^T\mathbf{U}'\mathbf{D}'\mathbf{U}'^T(\mathbf{U}'\mathbf{f}'') \\
&= \frac{1}{2}\mathbf{f}''^T\mathbf{f}'' \;-\; (\mathbf{U}\mathbf{D}^{-\frac{1}{2}}\mathbf{U}'\mathbf{f}'')^T(\mathbf{H}\mathbf{f}^{(k)} - \boldsymbol{\ell}) \;+\; \frac{\tau}{2}\mathbf{f}''^T\mathbf{D}'\mathbf{f}'' \\
&= \frac{1}{2}\mathbf{f}''^T(I + \tau\cdot\mathbf{D}')\mathbf{f}'' \;-\; (\mathbf{U}\mathbf{D}^{-\frac{1}{2}}\mathbf{U}'\mathbf{f}'')^T(\mathbf{H}\mathbf{f}^{(k)} - \boldsymbol{\ell}) \\
&= \frac{1}{2}\mathbf{f}''^T(I + \tau\cdot\mathbf{D}')\mathbf{f}'' \;-\; \mathbf{f}''^T(\mathbf{U}\mathbf{D}^{-\frac{1}{2}}\mathbf{U}')^T(\mathbf{H}\mathbf{f}^{(k)} - \boldsymbol{\ell})
\end{aligned}
$$

□

Proof of Theorem 2.3 The minimizer of the surrogate form is obtained by setting the gradient of this form to zero.

$$
\nabla_{\mathbf{f}''}\hat{\ell}_r^{(k)}(\mathbf{f}) \;=\; (I + \tau\cdot\mathbf{D}')\mathbf{f}'' \;-\; (\mathbf{U}\mathbf{D}^{-\frac{1}{2}}\mathbf{U}')^T(\mathbf{H}\mathbf{f}^{(k)} - \boldsymbol{\ell}) \;\stackrel{!}{=}\; 0
$$

$$
\begin{aligned}
\Rightarrow\quad (I + \tau\cdot\mathbf{D}')\hat{\mathbf{f}}'' \;&= (\mathbf{U}\mathbf{D}^{-\frac{1}{2}}\mathbf{U}')^T(\mathbf{H}\mathbf{f}^{(k)} - \boldsymbol{\ell}) \\
\Rightarrow\qquad\qquad \hat{\mathbf{f}}'' \;&= (I + \tau\cdot\mathbf{D}')^{-1}(\mathbf{U}\mathbf{D}^{-\frac{1}{2}}\mathbf{U}')^T(\mathbf{H}\mathbf{f}^{(k)} - \boldsymbol{\ell})
\end{aligned}
$$

For $\tau = 0$, the un-regularized solution $\hat{\hat{\mathbf{f}}}'' = (\mathbf{U}\,\mathbf{D}^{-\frac{1}{2}}\mathbf{U}')^T(\mathbf{H}\,\mathbf{f}^{(k)} \;-\; \boldsymbol{\ell})$ is obtained. The relation between the regularized estimate $\hat{\mathbf{f}}''$ and $\hat{\hat{\mathbf{f}}}''$ is therefore $\hat{\mathbf{f}}'' \;=\; (I + \tau\cdot\mathbf{D}')^{-1}\hat{\hat{\mathbf{f}}}''$. Equivalently,

$$
\hat{\mathbf{f}}_i'' \;=\; \frac{1}{1 + \tau\mathbf{D}'_{ii}}\hat{\hat{\mathbf{f}}}_i''
$$

□

Appendix B

Additional Extensions for DSEA

With the aim of improving the original Dsea, some additional extensions have been developed during the work on this thesis. However, in contrast to the improvements presented in Chapter 3, they do not enhance the algorithm. Therefore, they are omitted in the novel algorithm Dsea$^+$. Their concepts are documented here, along with the experimental results that indicate their low performance.

The first additional extension is an intermediate smoothing of the prior, presented in Section B.1. It adapts the smoothing of the Iterative Bayesian Unfolding. The second extension employs the expansion/reduction strategy originally proposed for the Run algorithm. Section B.2 considers this approach.

B.1 Smoothing

The smoothing operation applied in the Ibu algorithm aims at regularizing the deconvolution results under the assumption that the solution is smooth. Here, this operation is adapted to Dsea. As presented in Subsection 2.3.2, it fits an analytic form to the respective latest estimate $\hat{\mathbf{f}}^{(k-1)}$. This form is then used as the prior of the next iteration, instead of the actual $\hat{\mathbf{f}}^{(k-1)}$. The default analytic form of Ibu is a polynomial of some low order, which is employed in the Dsea extension as well. Like in Ibu, neither the initial prior, nor the last estimate are smoothed. The intention behind the smoothing extension is to regularize Dsea towards smooth solutions.

Figure B.1 presents the experimental results obtained by smoothing the intermediate Dsea estimates. Low- and high-order polynomials are investigated, ranging from the order 2 to the order 12. All smoothings are compared to a solution obtained without the smoothing extension. Apparently, this regular solution is more accurate than any of the smoothed estimates. The only smoothing which competes with the non-smoothed solution applies polynomials of the order 12, which almost do not smooth the estimates, at all.

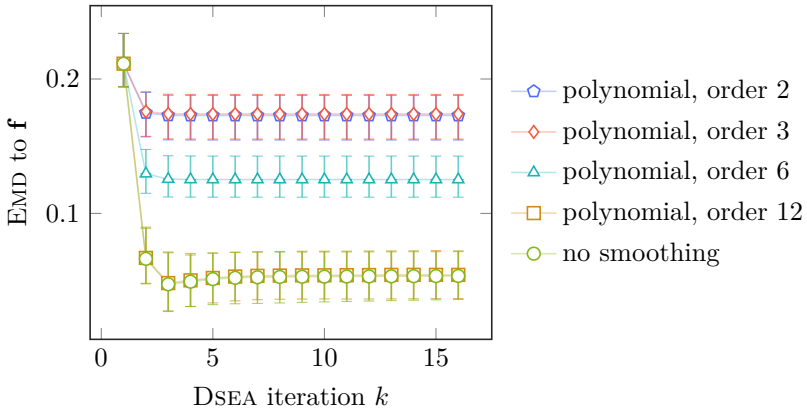


Figure B.1: Polynomials of several low and high orders are used to smooth the intermediate estimates of DSEA. In this experiment on the toy data set, the best results are obtained without smoothing. Low orders, which correspond to a strong smoothing, perform significantly worse. Here, DSEA uses a constant step size of $\alpha^{(k)} = 1$.

This finding is consistent with the effect of the smoothing operation in IBU. As presented in Subsection 4.4.2, the polynomial smoothing is not capable of improving the estimates—at least not for the data sets considered here. However, using an analytic form which represents more accurate assumptions than a low-order polynomial may produce results that outperform non-smoothed estimates. For now, DSEA$^+$ omits the smoothing extension due to the lack of such an accurate analytic form.

B.2 Expansion and Reduction

Figure B.2 applies DSEA$^+$ to different expansions of the deconvolution problem, following the expand/reduce strategy proposed for the RUN algorithm (see Subsection 2.4.4). The experimental results from Subsection 4.4.2 show that this strategy improves the RUN estimates slightly. For DSEA$^+$, such an improvement is not observed here.

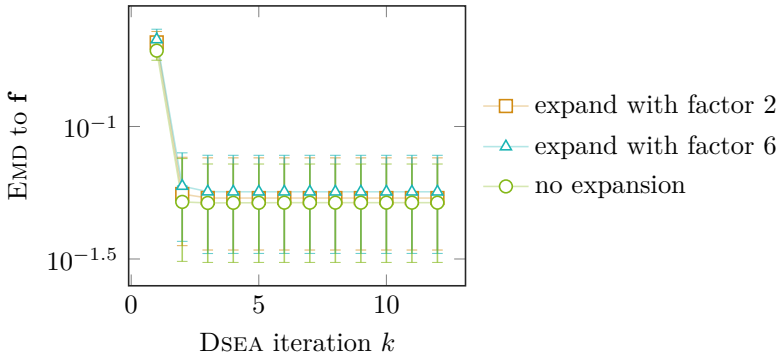


Figure B.2: DSEA$^+$ is applied to several expansions of the deconvolution problem, which do not improve the estimates in comparison to the non-expanded solution.

Appendix C

Experiments Handbook

The experiments presented in this thesis are centered around the novel Julia package `CherenkovDeconvolution.jl`, which bundles the investigated deconvolution methods for their application in the field. This package has been developed as a part of the present work, being presented in Section C.1. The actual experiments are implemented in another repository, which is presented in Section C.2. For reproducibility, both of these repositories are publicly available and well documented.

Since other studies on deconvolution may use other measures of performance than the Earth Mover's Distance favored here, four additional quality measures are computed for every experiment conducted in this thesis. These measures are defined in Section C.3. Their results are available with the supplementary material, enabling comparisons between the present study and others.

C.1 CherenkovDeconvolution.jl

The Julia package `CherenkovDeconvolution.jl` bundles the investigated deconvolution methods for their application in Cherenkov astronomy and other use cases from experimental physics. It is publicly available on GitHub[1], where it is backed by an issue tracker and a continuous integration platform to run the unit tests on. A little button on the web site of the package always indicates if the latest test run has been successful. The code coverage of the tests is assessed in another integrated tool. This collection of tools helps to achieve a high reliability of the software. `CherenkovDeconvolution.jl` is licensed under the GNU General Public License.

The documentation of the package is provided by an instructive "read-me" file and a Jupyter notebook, which contains examples on how to use the package. Moreover, each

[1] `https://github.com/mirkobunse/CherenkovDeconvolution.jl`

function is documented comprehensively. The API follows common patterns encountered in Julia packages, which makes the software easy to use for Julia developers. However, Julia's clear syntax is also understandable for programmers who are new to the language.

Each deconvolution method is run with a single function call. For instance, `ibu( x_data, x_train, y_train)` deconvolves the density of observations in the array `x_data` with IBU, the two other arrays representing the training data. Optional keyword arguments are used to configure each deconvolution run. A collection of utility functions—e.g. the decision tree clustering—supplements the package.

The embedded classifier from DSEA/DSEA$^+$ is adapted from the popular ScikitLearn library. Thus, a vast number of algorithms is readily available for classification.

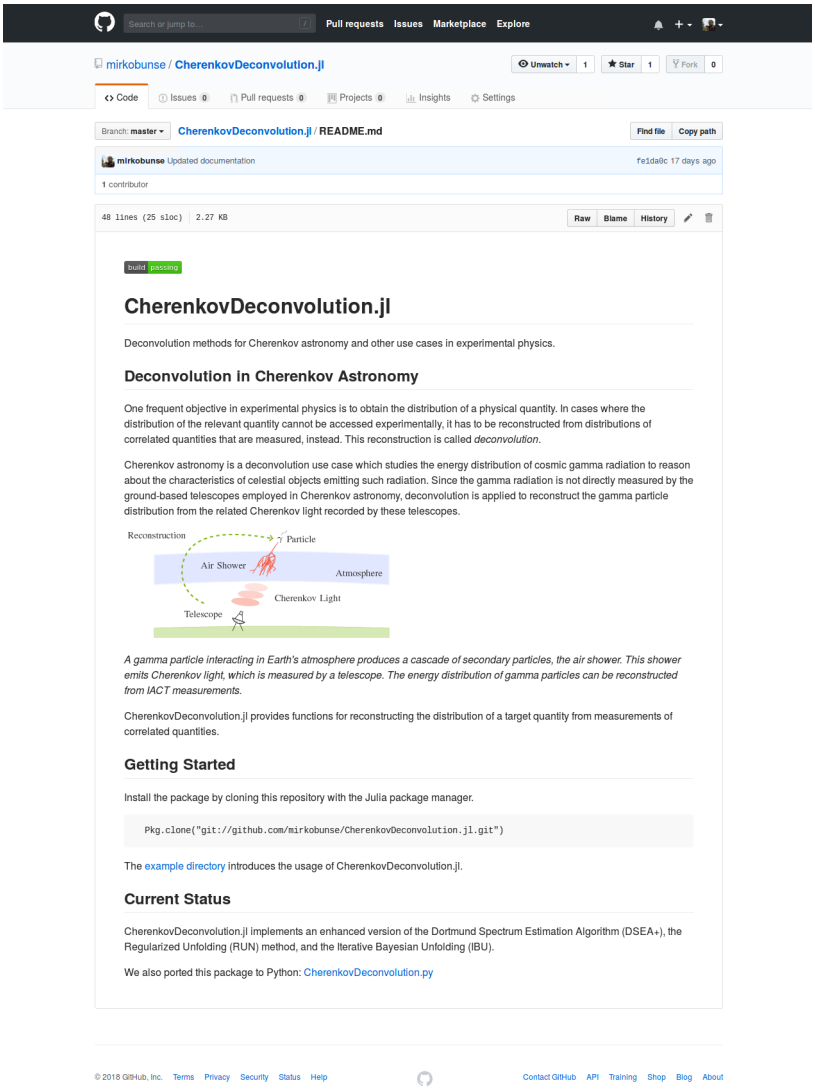


Figure C.1: The Julia package on GitHub is well documented and easy to use. In addition to the "read-me" file shown here, a Jupyter notebook provides examples on the usage of this package.

C.2 Reproducing the Results

All experiments conducted in this thesis are implemented in a second GitHub repository[2], which is also publicly available. The documentation of this repository describes the setup and the deployment of the experiments—locally and in a Docker cluster. Most of the installation is performed by the Julia package manager and a script written for this purpose. The Docker setup is established by building a Docker image, which is also straight-forward.

Each experiment is configured with a YAML file. This means that entirely new sets of experiments are easily obtained by reconfiguration. However, the provided configurations are comprehensive already, producing reliable and generalizable results. Each YAML file is read by an associated Julia function which conducts the experiment.

Each experiment returns a result file containing the deconvolution results produced during the run. The quality metrics are computed subsequently from these estimated densities. In a last step, the plots are generated from the quality metric files. This three-step procedure provides the practitioner with a high level of versatility. Namely, the evaluation can be changed without re-running the costly experiments.

Each plot is generated for all quality measures presented in the next subsection. All in all, a tremendous amount of figures is generated by the experiments. In order to keep track of these results, a graphical tool has been developed, with which multiple plots can be viewed side by side. It allows you to tick and un-tick boxes which correspond to experimental configurations to compare with each other.

Note that the IACT data sets must not be published with the experiments, due to restrictions of the telescope collaborations.

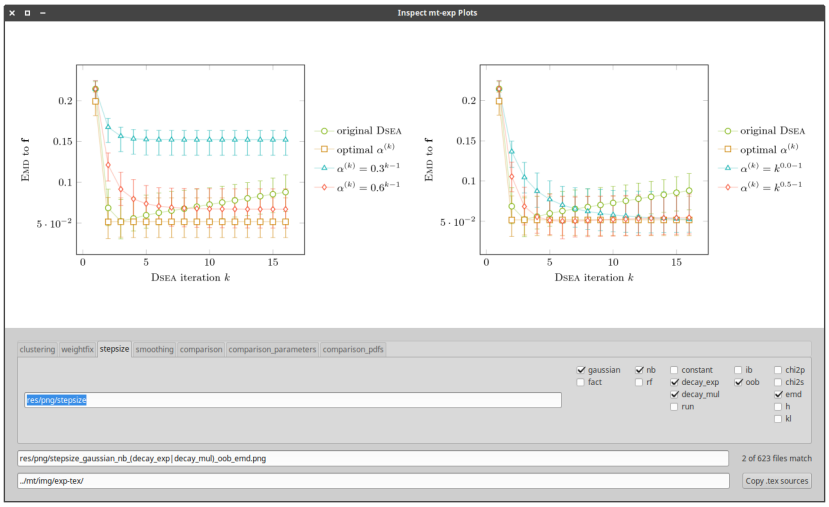


Figure C.2: The experimental results are compared with each other in the inspection tool.

[2] `https://github.com/mirkobunse/deconv-exp`

C.3 More Statistical Distance Measures

Given the vast number of statistical distance measures, evaluating multiple such distances increases the level of comparability to other studies, which may only use a subset—or even a single one—of these measures. The following measures are thus computed in addition to the Earth Mover's Distance presented in Section 4.1:

- The Kullback-Leibler divergence
- The Hellinger distance
- Pearson's χ^2 divergence
- The probabilistic symmetric χ^2 distance

These distances are among the most prominent measures of statistical distance, being presented in the following subsections. They are computed with the Julia package `Distances.jl`[3], which strictly follows the definitions given here. All measures are evaluated for each experiment. Their values are provided with the supplementary material of this thesis. With all of these measures, similar interpretations of the experimental results are obtained (see Figure C.4).

C.3.1 The Kullback-Leibler Divergence

The Kullback-Leibler divergence δ_{KL} [35] from Equation C.1 is interpreted as the amount of information which is lost by using a discrete density function $\hat{\mathbf{f}}$ instead of a more appropriate $\mathbf{f}$. This interpretation is justified by information theory, a field that quantifies how much information is contained in data. Since deconvolution results are evaluated by comparing estimated densities with a corresponding true—i.e. more appropriate—density, this interpretation fits our use case quite well. In other words, δ_{KL} quantifies the amount of information gained if the practitioner knew the truth, compared to what he already knows from the deconvolution result. A small value of δ_{KL} indicates that he can stick to the estimate, even if he knew the corresponding true density.

$$\delta_{KL}(\hat{\mathbf{f}} \,|\, \mathbf{f}) \;=\; \sum_{i=1}^{I} \hat{\mathbf{f}}_i \ln \frac{\hat{\mathbf{f}}_i}{\mathbf{f}_i} \tag{C.1}$$

Note that δ_{KL} is not symmetric, i.e. $\delta_{KL}(\hat{\mathbf{f}} \,|\, \mathbf{f}) \;\neq\; \delta_{KL}(\mathbf{f} \,|\, \hat{\mathbf{f}})$. Thus, it does not qualify as a metric distance. However, the asymmetry of δ_{KL} is acceptable for our evaluation because $\mathbf{f}$ and $\hat{\mathbf{f}}$ have fixed roles which are never switched. Also note, that since δ_{KL} divides by the bin values $\mathbf{f}_i$, it is usually assumed that $\mathbf{f}_i > 0 \;\forall\; 1 \leq i \leq I$. We deal with this issue by replacing zeros with a small value $\epsilon = 10^{-9}$, as suggested elsewhere [23].

[3] `https://github.com/JuliaStats/Distances.jl`

C.3.2 The Hellinger Distance

The Hellinger distance δ_H has already been applied to assess the quality of DSEA in other studies [4]. In contrast to the Kullback-Leibler Divergence, it has a geometric interpretation which provides a different point of view on the distance between probability density functions. It is a true metric, which can be helpful in comparing densities which do not take the roles of an estimate and the truth, e.g. in comparing multiple deconvolution results with each other. δ_H is defined in Equation C.2, where the right hand side is often multiplied by 2 [23, 25]. This factor is omitted here, so that δ_H is normalized to the unit interval. A value of $\delta_H = 1$ thus means no similarity at all and $\delta_H = 0$ means perfect similarity between $\mathbf{f}$ and $\hat{\mathbf{f}}$.

$$\delta_H(\hat{\mathbf{f}}, \mathbf{f}) = \sqrt{1 - \sum_{i=1}^{I} \sqrt{\hat{\mathbf{f}}_i \mathbf{f}_i}} \tag{C.2}$$

The sum of geometric means, $\sum_{i=1}^{I} \sqrt{\hat{\mathbf{f}}_i \mathbf{f}_i}$ in Equation C.2, is also referred to as the *Hellinger affinity* or the *Bhattacharyya coefficient* [23, 25]. It is a measure of similarity between discrete probability density functions which is motivated by the geometric considerations displayed in Figure C.3. To understand this measure, let $\mathbf{f}^{\surd} = (\sqrt{\mathbf{f}_1}, \dots \sqrt{\mathbf{f}_I})$ and $\hat{\mathbf{f}}^{\surd}$ be defined accordingly. Since $\mathbf{f}$ and $\hat{\mathbf{f}}$ are densities, i.e. their respective values sum up to one, we find that $\mathbf{f}^{\surd}$ and $\hat{\mathbf{f}}^{\surd}$ are vectors on the unit sphere in an I-dimensional space [36]. The angle α between these two vectors is a measure of similarity. For convenience, we do not regard that angle directly, but it's cosine, which is given by the sum of geometric means, i.e. $cos(\alpha) = \sum_{i=1}^{I} \sqrt{\hat{\mathbf{f}}_i \mathbf{f}_i}$ [36]. Since this cosine also captures the similarity between $\mathbf{f}$ and $\hat{\mathbf{f}}$, δ_H is a measure of dissimilarity.

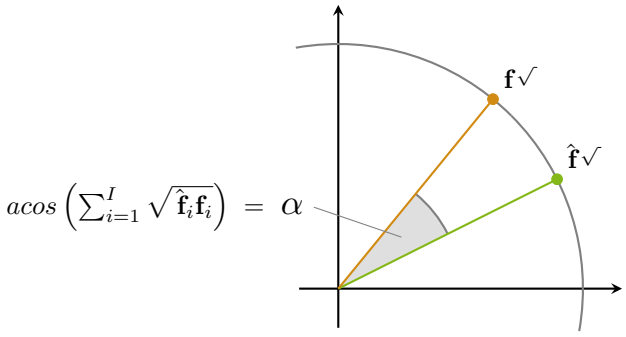


Figure C.3: The sum of geometric means $\sum_{i=1}^{I} \sqrt{\hat{\mathbf{f}}_i \mathbf{f}_i}$ is the cosine of the angle α between the vectors $\mathbf{f}^{\surd}$ and $\hat{\mathbf{f}}^{\surd}$. It is a measure of similarity between the densities $\mathbf{f}$ and $\hat{\mathbf{f}}$, because the vectors are projections of these densities onto the unit sphere. $I = 2$ in this illustration.

C.3.3 χ^2 Distances

The concept of χ^2 distance is rather ambiguous. Several definitions are found in literature, e.g., Pearson's χ^2_P divergence and multiple symmetric variants [23]. Moreover, different authors have referred to identical variants with different names. For example, the definition of Pearson's χ^2_P in [23] is referred to as Neyman's χ^2_N divergence in [25]. This ambiguity is problematic if another scientific reference does not explicitly define which specific χ^2 distance is used. For example, the Iterative Bayesian Unfolding from Section 2.3 suggests to use *some* χ^2 distance to define a stopping criterion without defining the particular measure explicitly [6].

To achieve a high level of comparability in spite of these difficulties, two different χ^2 distances are chosen here. The asymmetric χ^2_P divergence defined in Equation C.3 is regarded as the corner stone of χ^2 distances, being directly motivated by the well-known χ^2 test. The probabilistic symmetric χ^2_{Sym} distance from Equation C.4 is additionally chosen due to its popularity. Both measures are defined according to [23].

$$\chi^2_P(\hat{\mathbf{f}}, \mathbf{f}) = \sum_{i=1}^{I} \frac{\left(\hat{\mathbf{f}}_i - \mathbf{f}_i\right)^2}{\mathbf{f}_i} \tag{C.3}$$

$$\chi^2_{\text{Sym}}(\hat{\mathbf{f}}, \mathbf{f}) = 2 \cdot \sum_{i=1}^{I} \frac{\left(\hat{\mathbf{f}}_i - \mathbf{f}_i\right)^2}{\hat{\mathbf{f}}_i + \mathbf{f}_i} \tag{C.4}$$

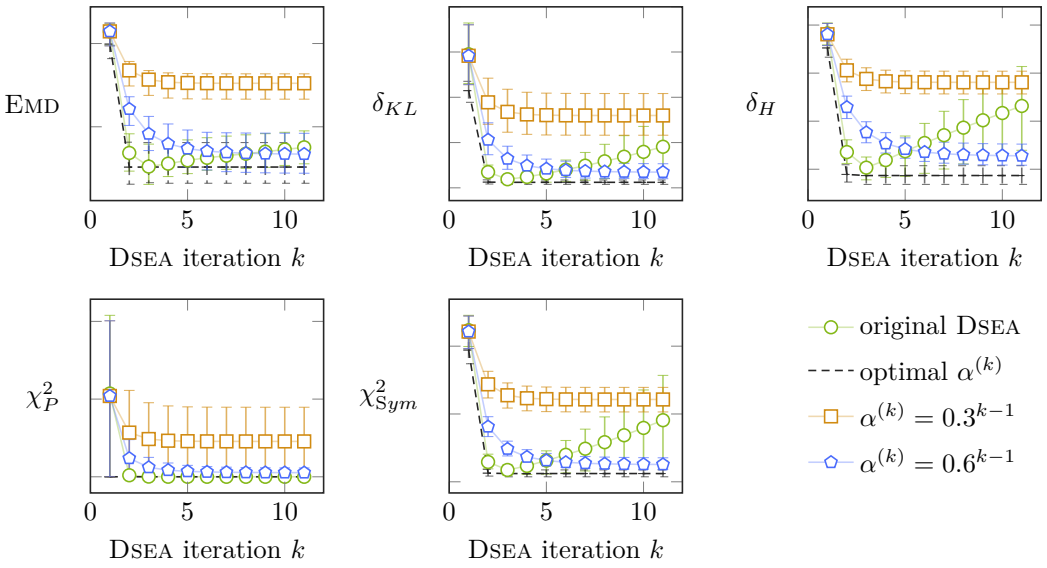


Figure C.4: All distance measures produce plots with similar interpretations. This is shown for the exponential decay strategy from Figure 3.4, where convergence happens too early.

List of Figures

List of Algorithms

Bibliography

[1] C. Bockermann, K. Brügge, J. Buss, A. Egorov, K. Morik, W. Rhode, and T. Ruhe, "Online analysis of high-volume data streams in astroparticle physics," in *Proc. of the ECML-PKDD 2015*. Springer, 2015.

[2] V. Blobel, "An unfolding method for high energy physics experiments," in *Advanced Statistical Techniques in Particle Physics*, 2002, pp. 258–267, long version available: http://www.desy.de/~blobel/durhamlong02.ps.

[3] T. Ruhe, M. Schmitz, T. Voigt, and M. Wornowizki, "DSEA: A data mining approach to unfolding," in *Proc. of the 33rd International Cosmic Ray Conference*, 2013.

[4] T. Ruhe, M. Börner, M. Wornowizki *et al.*, "Mining for spectra - the Dortmund spectrum estimation algorithm," in *Astronomical Data Analysis Software and Systems XXVI*. Astronomical Society of the Pacific, 2016, accepted for publication.

[5] V. Blobel, "Unfolding methods in high-energy physics experiments," CERN, Tech. Rep., 1985.

[6] G. D'Agostini, "A multidimensional unfolding method based on bayes' theorem," *Nuclear Instruments and Methods in Physics Research Section A*, vol. 362, no. 2-3, pp. 487–498, 1995.

[7] G. D'Agostini, "Improved iterative bayesian unfolding," *arXiv e-prints*, 2010.

[8] G. H. John and P. Langley, "Estimating continuous distributions in bayesian classifiers," in *Proc. of the 11th Conference on Uncertainty in Artificial Intelligence*. Morgan Kaufmann Publishers Inc., 1995, pp. 338–345.

[9] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.

[10] IceCube Collaboration, "Development of a general analysis and unfolding scheme and its application to measure the energy spectrum of atmospheric neutrinos with IceCube," *The European Physical Journal C*, vol. 75, no. 3, p. 116, 2015.

[11] R. K. Bock, A. Chilingarian, M. Gaug, F. Hakl, T. Hengstebeck, M. Jiřina, J. Klaschka *et al.*, "Methods for multidimensional event classification: a case study using images from a Cherenkov gamma-ray telescope," *Nuclear Instruments and Methods in Physics Research Section A*, vol. 516, no. 2-3, pp. 511–528, 2004.

[12] FACT Collaboration, "FACT – first energy spectrum from a SiPM Cherenkov telescope," *Proceedings of Science*, p. 707, 2015.

[13] K. Berger, T. Bretz, D. Dorner, D. Hoehne, and B. Riegel, "A robust way of estimating the energy of a gamma ray shower detected by the MAGIC telescope," in *Proc. of the 29th International Cosmic Ray Conference*, 2005, pp. 100–104.

[14] L. Breiman, "Bagging predictors," *Machine Learning*, vol. 24, no. 2, pp. 123–140, 1996.

[15] L. Breiman, J. Friedman, R. A. Olshen, and C. J. Stone, *Classification and regression trees*. Chapman and Hall/CRC, 1984.

[16] J. L. Mueller and S. Siltanen, "Truncated singular value decomposition," in *Linear and Nonlinear Inverse Problems with Practical Applications*. Society for Industrial and Applied Mathematics (SIAM), 2012, ch. 4, pp. 53–61.

[17] M. Börner, T. Hoinka, M. Meier, T. Menne, W. Rhode, and K. Morik, "Measurement/simulation mismatches and multivariate data discretization in the machine learning era," in *Astronomical Data Analysis Software and Systems XXVII*. Astronomical Society of the Pacific, 2017, accepted for publication.

[18] C. F. Eick, N. Zeidat, and Z. Zhao, "Supervised clustering – algorithms and benefits," in *Proc. of the 16th International Conference on Tools with Artificial Intelligence*. IEEE, 2004, pp. 774–776, long version available: https://pdfs.semanticscholar.org/8058/1bd42eeeb1fe22c2a3a48024b637038932cf.pdf.

[19] S. J. Wright and J. Nocedal, *Numerical optimization*, 2nd ed., Operations Research and Financial Engineering. New York: Springer, 2006.

[20] A. Hoecker and V. Kartvelishvili, "SVD approach to data unfolding," *Nuclear Instruments and Methods in Physics Research Section A*, vol. 372, no. 3, pp. 469–481, 1996.

[21] M. Feindt, "A neural bayesian estimator for conditional probability densities," *arXiv e-prints*, 2004.

[22] N. Gagunashvili, "Machine learning approach to inverse problem and unfolding procedure," *arXiv e-prints*, 2010.

[23] S. H. Cha, "Comprehensive survey on distance/similarity measures between probability density functions," *International Journal of Mathematical Models and Methods in Applied Sciences*, vol. 1, no. 4, 2007.

[24] Y. Rubner, C. Tomasi, and L. J. Guibas, "A metric for distributions with applications to image databases," in *Proc. of the 6th International Conference on Computer Vision*. IEEE, 1998, pp. 59–66.

[25] E. Deza and M. M. Deza, "Distances in probability theory," in *Dictionary of Distances*. Amsterdam: Elsevier, 2006, ch. 14, pp. 176–188.

[26] S. H. Cha and S. N. Srihari, "On measuring the distance between histograms," *Pattern Recognition*, vol. 35, no. 6, pp. 1355–1370, 2002.

[27] T. Ruhe, "Data mining on the rocks: A measurement of the atmospheric muon neutrino flux using IceCube in the 59-string configuration and a novel data mining based approach to unfolding," Ph.D. dissertation, TU Dortmund, 2013.

[28] M. A. Hillas, "Cerenkov light images of EAS produced by primary gamma," in *Proc. of the 19th International Cosmic Ray Conference*, 1985.

[29] FACT Collaboration, "Design and operation of FACT - the first G-APD Cherenkov telescope," *Journal of Instrumentation*, vol. 8, 2013.

[30] MAGIC Collaboration, "The MAGIC-II gamma-ray stereoscopic telescope system," *Nuclear Instruments and Methods in Physics Research Section A*, vol. 623, no. 1, pp. 437–439, 2010.

[31] MAGIC Collaboration, "Performance of the MAGIC telescopes in stereoscopic mode," *arXiv e-prints*, 2009.

[32] M. Bunse, C. Bockermann, J. Buss, K. Morik, W. Rhode, and T. Ruhe, "Smart control of monte carlo simulations for astroparticle physics," in *Astronomical Data Analysis Software and Systems XXVII*. Astronomical Society of the Pacific, 2017, accepted for publication.

[33] R. Lomasky, C. E. Brodley, M. Aernecke, D. Walt, and M. Friedl, "Active class selection," in *Proc. of the 18th European Conference on Machine Learning*. Springer, 2007, pp. 640–647.

[34] G. Cowan, "A survey of unfolding methods for particle physics," in *Advanced Statistical Techniques in Particle Physics*, 2002.

[35] S. Kullback and R. A. Leibler, "On information and sufficiency," *The Annals of Mathematical Statistics*, vol. 22, no. 1, pp. 79–86, 1951.

[36] A. Bhattacharyya, "On a measure of divergence between two multinomial populations," *Sankhyā: The Indian Journal of Statistics*, vol. 7, no. 4, pp. 401–406, 1946.

Ich versichere hiermit an Eides statt, dass ich die vorliegende Masterarbeit mit dem Titel 'DSEA Rock-Solid' selbstständig und ohne unzulässige fremde Hilfe erbracht habe. Ich habe keine anderen als die angegebenen Quellen und Hilfsmittel benutzt sowie wörtliche und sinngemäße Zitate kenntlich gemacht. Die Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

Dortmund, 16. Juli 2018

_____

Mirko Bunse