Entropy-optimal scalable probabilistic approximation algorithms

Philipp Wolf

June 4, 2025

Johannes Gutenberg University Mainz

FB08
Institute of Mathematics
Arbeitsgruppe Numerik

Entropy-optimal scalable probabilistic approximation algorithms

Philipp Wolf

1. Reviewer Prof. Dr. Hendrik Ranocha
Institute of Mathematics - Arbeitsgruppe Numerik
Johannes Gutenberg University Mainz

2. Reviewer Dr. Mattia Cerrato
Institute of Computer Science - Data Mining Group
Johannes Gutenberg University Mainz

June 4, 2025

Philipp Wolf
Entropy-optimal scalable probabilistic approximation algorithms
Reviewers: Prof. Dr. Hendrik Ranocha and Dr. Mattia Cerrato
June 4, 2025
Johannes Gutenberg University Mainz
Arbeitsgruppe Numerik
Institute of Mathematics
FB08
Staudingerweg 9
55128 Mainz

Abstract

In the small data regime, where the number of training samples is low relative to the dimensionality of the feature space, traditional classification methods often suffer from overfitting and poor generalization. This thesis investigates a family of classification algorithms specifically designed to address these challenges, collectively known as entropy-optimal scalable probabilistic approximation (eSPA). A detailed presentation and theoretical analysis of the eSPA algorithms is provided. The main contributions of this work are: (1) the introduction of a novel model, eSPAhybrid, which aims to combine the computational efficiency of discrete segmentation with the advantages of fuzzy segmentation; (2) the development of an open-source Julia package, eSPA.jl, implementing eSPAdiscrete, eSPAfuzzy, eSPA+, GOAL and eSPAhybrid; and (3) a set of extensive experiments conducted on synthetic and real-world datasets. Experimental results demonstrate that eSPAdiscrete and eSPA+ offer the best performance, with a trade-off between the faster runtime of eSPA+ and the higher accuracy of eSPAdiscrete. It is additionally shown that eSPA+ is highly sensitive to hyperparameters, making the selection non-trivial. Furthermore, runtime analysis reveals that optimization steps involving interior-point methods are the primary computational bottlenecks. Finally, results suggest that the proposed eSPAhybrid model underperforms in most settings.

Abstract (german)

Das sogenannte "small data regime" beschreibt die Situation, in der die Anzahl der Trainingssamples im Verhältnis zur Dimensionalität der Daten gering ist. In diesem Szenario leiden herkömmliche Klassifizierungsmethoden oft unter Overfitting und schlechter Generalisierbarkeit. In dieser Thesis wird eine Familie von Algorithmen untersucht, die speziell für dieses Szenario entwickelt wurden und hier kollektiv als entropy-optimale scalable probabilistic approximation (eSPA) bezeichnet werden. Diese werden anschaulich erklärt und theoretische Ergebnisse vorgestellt. Die wichtigsten Beiträge dieser Thesis sind: (1) Die Einführung eines neuen Modells, eSPAhybrid, welches die Effizienz der diskreten Segmentierung mit den Vorteilen

der fuzzy Segmentierung kombinieren soll; (2) die Entwicklung eines Open-Source Julia-Paket, eSPA.jl, welches eSPAdiscrete, eSPAfuzzy, eSPA+, GOAL und eSPAhybrid implementiert; (3) eine Reihe von Experimenten, welche auf einem syntethischen und einem echten Datensatz durchgeführt wurden. Die Experimente zeigen, dass eSPAdiscrete und eSPA+ am besten funktionieren, wobei eSPAdiscrete die bessere Genauigkeit aufweist, während eSPA+ schneller ist. Außerdem wurde herausgefunden, dass eSPA+ äußerst empfindlich bzgl. der Wahl der Hyperparameter ist, wodurch die Auswahl der Hyperparameter deutlich erschwert wird. Darüber hinaus zeigt eine Analyse der Laufzeiten, dass die Optimierungsschritte, die Interior-Point Methoden benutzen, signifikant mehr Laufzeit benötigen, als andere Schritte. Das neu eingeführte Modell eSPAhybrid weißt in den Experimenten unterdurchschnittliche Ergebnisse auf.

Contents

1 Introduction **1**

2 Algorithms **3**

2.1 SPA . . . 3

2.2 eSPA . . . 4

2.2.1 Algorithm . . . 4

2.2.2 Theoretical results . . . 7

2.3 eSPA+ . . . 8

2.3.1 Algorithm . . . 8

2.3.2 Theoretical Results . . . 9

2.4 GOAL . . . 12

2.4.1 Algorithm . . . 12

2.4.2 Theoretical Results . . . 14

2.5 eSPAhybrid . . . 15

2.5.1 Algorithm . . . 15

2.5.2 Theoretical Results . . . 15

3 Implementation **17**

3.1 Interior Point methods . . . 17

3.2 Empty clusters . . . 18

3.3 Infinity Loss Problem . . . 18

3.4 S-Step fuzzy . . . 19

3.5 eSPAhybrid . . . 19

4 Experiments **21**

4.1 Benchmarking . . . 22

4.1.1 Experimental Setup . . . 22

4.1.2 Results on synthetic Data . . . 23

4.1.3 Results Darwin . . . 25

4.1.4 Limitations of the experimental protocol . . . 28

4.2 Hyperparameter Sensitivity . . . 30

4.2.1 Experimental Setup . . . 30

4.2.2 Results . . . 31

4.3 eSPAhybrid 33
4.3.1 Experimental setup 33
4.3.2 Results 34

5 Conclusion **37**

Bibliography **39**

List of Figures **41**

List of Tables **43**

List of Algorithms **45**

Declaration **47**

1 Introduction

In supervised learning, a subfield of machine learning, a dataset called training set (or training data) is given, denoted by $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{T}$. The x_i's are multidimensional vectors, referred to as features and the corresponding y_i is the associated label. It is typically assumed that the training data is sampled from an underlying probability distribution shared across all (x_i, y_i) pairs [Bis06]. The goal is to use this training set $\mathcal{D}$ to learn a function $\hat{f}$ that can predict the label y for an unseen feature x, assuming it follows the same distribution as the training data. When the labels take on categorical values, i.e. $y_i \in \{1, \ldots, M\}$, $\forall\, 1 \leq i \leq T$, this problem is called classification [Mur22].

Although classification is a well-established problem in machine learning, traditional methods in this field face significant challenges in what is known as the *small data regime*. This refers to the scenario in which the number of training samples T is small relative to the dimension D of the features. However, this notion is context-dependent and not precise at all: the specific threshold for what is considered as "small data" may vary with classification method and other factors. As such, a strict boundary cannot be drawn. Nevertheless, empirical criteria can be found for concrete cases. One example is a bound on the ratio between T and D ($T \geq 14\, D$) needed for long short-term memory deep learning networks [HS97] to perform well [Vec+22].

Standard classification techniques often suffer from overfitting, when used in the small data regime, where they tend to model noise in the input data rather the the underlying signal. As a result, such a model typically perform well on training data but poorly on unseen data, highlighting their inability to generalize. This leads to the need of specialized strategies for "small-data" classification. Common approaches include data augmentation, transfer learning [Vec+22], dimensionality reduction and regularization [Hor20]. However, when using data from domains such as natural sciences, data augmentation and transfer learning often underperform. Moreover, Horenko (2020)[Hor20] has shown that dimensionality reduction and regularization are less effective in empirical experiments than the methods explored in this thesis.

This thesis focuses on a family of classification algorithms designed for the small data regime, here collectively referred to as *entropy-optimal scalable probabilistic approximation algorithms*. The foundation of these algorithms is *eSPA* (entropy-optimal scalable probabilistic approximation), introduced by Horenk (2020) [Hor20], by extending *SPA* (Scalable Probabilistic Approximations) [Ger+20], a clustering method, to classification. eSPA stands out by unifying dimensionality reduction, feature selection and classification in one unified algorithm. Subsequent developments include *eSPA+* [Vec+22], which improves runtime performance mainly through utilizing a closed-form solution to an optimization problem. The most recent advancement in this line is *GOAL* (Gauge-Optimal Approximate Learning) [Vec+24], which enhances the approach of eSPA by incorporating feature space rotation and reduction.

This thesis provides three main contributions:

1. A novel model named *eSPAhybrid* is proposed, building upon eSPA and eSPA+. Explanations and theoretical results for all discussed algorithms, including eSPAhybrid, are presented in chapter 2.

2. An open-source Julia package *eSPA.jl* implementing these algorithms has been developed as part of this thesis. It is available on GitHub[1]. Implementation details and encountered problems while developing are discussed in chapter 3.

3. A set of empirical experiments has been conducted on two datasets: a synthetic dataset and a medical one. These experiments evaluate the performance of the models, analyze hyperparameter sensitivity and investigate the proposed eSPAhybrid. The experimental setup and results are detailed in chapter 4.

[1] `https://github.com/pw0lf/eSPA.jl`

2 Algorithms

This chapter introduces the algorithms belonging to the eSPA family. It starts with its predecessor SPA and proceeds in chronological order, concluding with the proposed eSPAhybrid. For each method, the core concepts are explained, detailed steps are described and theoretical results, such as convergence and computational complexity, are discussed.

2.1 SPA

The eSPA algorithms are based on SPA (Scalable Probabilistic Approximations) by Gerber et al. [Ger+20]. SPA is a clustering[1] algorithm that partitions the feature space $\mathbb{R}^{D\times T}$ into K disjoint, piecewise-linear regions, referred to as "boxes".

Given a dataset represented by a matrix $X \in \mathbb{R}^{D\times T}$, SPA determines an optimal discretization of the feature space by minimizing a loss function $\mathcal{L}_{\text{SPA}}$ (eq. 2.1). More precisely, the algorithm searches a set of K box centers $S = \{S_1, \ldots, S_K\} \in \mathbb{R}^{D\times K}$ and a stochastic matrix $\Gamma \in \Omega_\Gamma$ (eq. 2.2), where each entry $\Gamma_{k,t}$ represents the probability that a data point $X_{\cdot,t}$ belongs to the k-th box. The loss function $\mathcal{L}_{\text{SPA}}$ consists of two terms. The first term quantifies the average reconstruction error by measuring the Euclidean distance between each data point $X_{\cdot,t}$ and its approximation $(S\Gamma)_{\cdot,t}$. The second term, regulated by $\epsilon_S \geq 0$, penalizes large distances between the centers of different boxes, aiming to filter out the dimensions without an impact on the box discretizations, thus leading to so-called feature selection [Vec+22].

$$\mathcal{L}_{\text{SPA}} = \frac{1}{TD}\sum_{d=1}^{D}\sum_{t=1}^{T}(X_{dt} - (S\Gamma)_{dt})^2 + \epsilon_S \sum_{d=1}^{D}\sum_{k_1,k_2=1}^{K}\left(S_{dk_1} - S_{dk_2}\right)^2 \tag{2.1}$$

$$\Omega_\Gamma := \left\{\Gamma \in \mathbb{R}^{K\times T} \middle| \forall k,t : \Gamma_{kt} \in [0,1] \wedge \forall t : \sum_{k=1}^{K}\Gamma_{kt} = 1\right\} \tag{2.2}$$

[1] A clustering algorithm attempts to group objects based on similarity [HTF09].

Although SPA is a clustering algorithm it could be used for classification tasks in the small data regime by serving as a preprocessing step. It is a common approach in preprocessing pipelines to use feature extraction (e.g. PCA or t-SNE) followed by clustering (e.g. K-means)[Hor20]. Both can be done by SPA, followed by training a common classifier. However, this approach is inherently limited, as it does not utilize the available label information.

2.2 eSPA

2.2.1 Algorithm

Entropy-optimal scalable probabilistic approximation [Hor20], short eSPA, extends SPA to classification of labeled data. As in SPA, the algorithm utilizes a set of box centers S and a stochastic matrix Γ to describe the data discretization. However, eSPA introduces several modifications to align the discretization process with the classification task. A central change is the introduction of a probability density vector $W \in \mathbb{R}^D$ ($W_i \geq 0$, $\sum_{d=1}^{D} W_d = 1$), in which each component represents the likelihood of a particular dimension to contribute to the box discretization [Vec+22]. The regularization term from the original SPA loss function (second term in 2.1) is replaced by a new entropy-based term $\mathcal{L}_E$ (eq. 2.3), while the reconstruction loss remains in a modified form. The updated $\mathcal{L}_{\text{rec}}$ (eq. 2.4) integrates the vector W, thereby assigning different importance to feature dimensions during the optimization. By minimizing the functions $\mathcal{L}_E$ and $\mathcal{L}_{\text{rec}}$, eSPA penalizes uniform distribution in W, encouraging the algorithm to assign higher weights to more informative dimensions, since minimizing the weight of an important dimension would increase the reconstruction loss and vice versa. The uniform distribution gets penalized, since entropy is a measure of uncertainty of a random variable [CT06] and a uniformed distribution has the highest uncertainty. The influence of $\mathcal{L}_E$ on the loss function is controlled by the hyperparameter $\epsilon_E \geq 0$ [Vec+22].

$$\mathcal{L}_E = \frac{\epsilon_E}{D} \sum_{d=1}^{D} W_d \log W_d \tag{2.3}$$

$$\mathcal{L}_{\text{rec}} = \frac{1}{T} \sum_{d=1}^{D} W_d \sum_{t=1}^{T} (X_{dt} - \{S\Gamma\}_{dt})^2 \tag{2.4}$$

$$\mathcal{L}_{KL} = -\frac{\epsilon_{CL}}{TM} \sum_{m=1}^{M} \sum_{t=1}^{T} \Pi_{mt} \log \Big(\sum_{k=1}^{K} \Lambda_{mk} \Gamma_{kt} \Big) \tag{2.5}$$

$$\mathcal{L}_{\text{eSPA}} = \mathcal{L}_E + \mathcal{L}_{\text{rec}} + \mathcal{L}_{KL} \tag{2.6}$$

Another central addition is the label probability matrix $\Pi \in \mathbb{R}^{M \times T}$, where Π_{mt} denotes the probability that the t-th data point $X_{:,t}$ belongs to class m. The concrete Π is directly given by the labels y_i according to equation 2.8. Given the discretization of the feature matrix by S and Γ, the relationship between a certain box S_k and the label probabilities is captured by $\Pi = \Lambda\Gamma$. Thus, the matrix $\Lambda \in \mathbb{R}^{M \times K}$ encodes the conditional probability of belonging to class m, if a datapoint is assigned to box k. Consequently, Λ is constrained to be a stochastic matrix, defined as $\Omega_\Lambda = \{\Lambda \in \mathbb{R}^{M \times K} | \forall m, k : \Lambda_{mk} \in [0,1] \wedge \forall k : \sum_{m=1}^{M} \Lambda_{mk} = 1\}$. The Kullback-Leibler (KL) divergence is a measure of the distance between two probability distributions [CT06]. Thus the average KL divergence between the real label probabilities Π and the predicted label probabilities $\Lambda\Gamma$ gets minimized to determine Λ, as provided in $\mathcal{L}_{KL}$ (eq. 2.5) [Hor20].

$$\Lambda_{mk} = \mathbb{P}[\Pi_{mt} = 1, \text{ if } X_{:,t} \text{ is in box } k] \tag{2.7}$$

$$\Pi_{mt} = \begin{cases} 1 & \text{if } y_t = m \\ 0 & \text{else} \end{cases} \tag{2.8}$$

This results in an overall loss function $\mathcal{L}_{\text{eSPA}}$ (eq. 2.6), which is minimized using the algorithm 1, such that Γ, Λ are stochastic matrices and W is a probability vector. Once the model is trained by minimizing $\mathcal{L}_{\text{eSPA}}$, predictions on previously unseen data can be made using the procedure outlined in algorithm 2 [Hor20].

Algorithm 1 eSPA fit

Require: $K \in \mathbb{N}$, $\epsilon_E \geq 0$, $\epsilon_{CL} \geq 0$, $tol \geq 0$ and randomly initialized Γ and W
  $i = 1$
  $\mathcal{L}^{(1)} = \infty$, $\Delta\mathcal{L} = \infty$
  while $\Delta\mathcal{L} > tol$ **do**
    S**-Step**
    Λ**-Step**
    Γ**-Step**
    W**-Step**
    Update-Loss: $\mathcal{L}^{(i+1)} = \mathcal{L}_{\text{eSPA}}$, $\Delta\mathcal{L} = \mathcal{L}^{(i)} - \mathcal{L}^{(i+1)}$
    $i = i + 1$
  end while

In the steps of algorithm 1 the following optimization problems have to be solved:

- **S-Step:**

$$S = \text{argmin}_{S \in \mathbb{R}^{D \times T}} \big[\frac{1}{T} \sum_{d=1}^{D} W_d \sum_{t=1}^{T} \big(X_{dt} - (S\Gamma)_{dt}\big)^2 \big]$$

- Λ**-Step:**

$$\Lambda = \text{argmin}_{\Lambda \in \Omega_\Lambda} \big[\sum_{m=1}^{M} \sum_{t=1}^{T} \Pi_{mt} \log \big(\sum_{k=1}^{K} \Lambda_{mk} \Gamma_{kt} \big) \big]$$

- Γ**-Step:** for $t \in \{1, \ldots, T\}$:

$$\Gamma_{:,t} = \text{argmin}_{\Gamma_{:,t} \geq 0; \sum_{k,t} \Gamma_{kt} = 1} \big[\sum_{d=1}^{D} W_d \big(X_{dt} - (S\Gamma)_{dt}\big)^2 - \frac{\epsilon_{CL}}{M} \sum_{m=1}^{M} \Pi_{mt} \log \big(\sum_{k=1}^{K} \Lambda_{mk} \Gamma_{kt} \big) \big]$$

- **W-Step:**

$$W = \text{argmin}_{W \geq 0; \sum_d W_d = 1} \big[\frac{1}{D} \sum_{d=1}^{D} W_d \big(\epsilon_E \, log \, W_d + \frac{D}{T} \sum_{t=1}^{T} \big(X_{dt} - (S\Gamma)_{dt}\big)^2 \big) \big]$$

The Λ-Step, Γ-Step and W-Step can be addressed using an interior-point method [NW06]. The optimization of the box centers S, known as the S-Step, can be formulated as a weighted linear regression problem [SL12]. In a special case, where each data point is assigned exclusively to a single box, the optimization problem simplifies substantially, as closed-form solutions exists for the S-Step, Λ-Step and Γ-Step. Precisely this is achieved by restricting Γ to $\{\Gamma \in \{0,1\}^{K \times T} | \sum_{k=1}^{K} \Gamma_{kt} = 1 \text{ for all } t\}$. This enforces that each column of Γ contains exactly one nonzero entry. The corresponding closed-form solutions for these steps are given by equations 2.9,

2.10 and 2.11. In the following, this constrained setting will be referred to as the discrete case, while the general and unrestricted setting will be termed the fuzzy case [Hor20].

$$S_{dk} = \frac{\sum_{t=1}^{T} \Gamma_{kt} X_{dt}}{\sum_{t=1}^{T} \Gamma_{kt}} \tag{2.9}$$

$$\begin{aligned} \tilde{\Lambda} &= \Pi \, \Gamma^T \\ \Lambda_{mk} &= \frac{\tilde{\Lambda}_{mk}}{\sum_{m=1}^{M} \tilde{\Lambda}_{mk}} \end{aligned} \tag{2.10}$$

$$\begin{aligned} \text{Given } k' = \text{argmin}_{k'} \Big[& - \frac{\epsilon_{CL}}{M} \sum_{m=1}^{M} \Pi_{mt} \text{max}[log(\Lambda_{mk'}), tol] \\ & + \sum_{d=1}^{D} W_D (X_{dt} - S_{dk'})^2 \Big], \\ \text{then } \Gamma_{kt} = & \begin{cases} 1 & \text{, if } k = k' \\ 0 & \text{, else} \end{cases} \end{aligned} \tag{2.11}$$

Algorithm 2 eSPA prediction

Require: Unlabelled data X_ν and Λ, S and W as a result of algorithm 1
Discretization of X_ν: Calculate Γ_ν by evaluating eq. 2.12 with interior-point method in fuzzy case and eq. 2.13 in discrete case.
Compute labels $\Pi_\nu = \Lambda \Gamma_\nu$

$$(\Gamma_\nu)_{:,t} = \text{argmin}_{(\Gamma_\nu)_{:,t} \geq 0; \sum_{k,t} (\Gamma_\nu)_{kt} = 1} \Big[\sum_{d=1}^{D} W_d \big((X_\nu)_{dt} - (S \Gamma_\nu)_{dt} \big)^2 \Big] \tag{2.12}$$

$$(\Gamma_\nu)_{kt} = \begin{cases} 1 & \text{, if } k = \text{argmin}_{k'} \big[\sum_{d=1}^{D} W_d \big((X_\nu)_{dt} - S_{dk'} \big) \big] \\ 0 & \text{, else} \end{cases} \tag{2.13}$$

2.2.2 Theoretical results

Theorem 1 *1. The minimal solution to $\mathcal{L}_{eSPA}$, such that Γ and Λ are stochastic matrices and W is a probability vector, is given by a segmentation Γ that is piece-wise linear in the feature space.*

2. *Algorithm 1 approximates the minimal solution monotonically convergent.*

3. *With K clusters, T samples, D feature dimensions and M labels, the cost of one iteration of algorithm 1 is $\mathcal{O}(KT(D+M)+KM+T(K+M+D)+D\,log(D))$ in the discrete case*

4. *The cost of one iteration of algorithm 1 is $\mathcal{O}(KT(D+M)+K^3(M^3+T+1)+D\,log(D))$ in the fuzzy case.*

[Hor20]

Refer to Horenko (2020)[Hor20] for the proof.

2.3 eSPA+

2.3.1 Algorithm

An improvement on eSPA was proposed by Vecchi et al. [Vec+22], termed eSPA+, which introduces two key improvements. First, the order of the optimization steps and the initializations procedure are modified. Secondly, a closed-form solution for the W-step is derived.

One issue with the original eSPA algorithm is the tendency of box centers to collapse toward a common point when the number of boxes K is large. To address this, eSPA+ initializes the box centers by selecting K random data points from X, rather than computing them from the initial values of Γ and W. Furthermore, the matrices Λ and W are initialized randomly while preserving their stochastic properties. The random initialization of W introduces an initial distribution of feature dimension importance that can be adjusted, instead of starting with an uniform distribution. These modification result in the reordered optimization algorithm 3, which starts with the Γ-step [Vec+22].

Additionally, algorithm 3 uses the closed-form solution for of W-Step (eq. 2.14). The other steps and the loss function remain identical to those used in eSPA. Similarly, the prediction method (alg. 2) also remains unchanged.

$$W = \frac{exp\Big(-\frac{D}{T\epsilon_E}\sum_{t=1}^{T}\left(X_{dt}-S\Gamma_{dt}\right)^2 - \mathbb{1}_D\Big)}{\mathbb{1}_D^T exp\Big(-\frac{D}{T\epsilon_E}\sum_{t=1}^{T}\left(X_{dt}-S\Gamma_{dt}\right)^2 - \mathbb{1}_D\Big)} \tag{2.14}$$

Algorithm 3 eSPA+ fit

Require: K,ϵ_E ,ϵ_{CL}, tol and randomly initialized Λ, W and S, with the centers S being randomly drawn from X.
$i = 1$
$\mathcal{L}^{(1)} = \infty$, $\Delta\mathcal{L} = \infty$
while $\Delta\mathcal{L} > tol$ **do**
  Γ**-Step**
  W-Step
  S-step
  Λ**-Step**
  Update-Loss: $\mathcal{L}^{(i+1)} = \mathcal{L}_{\text{eSPA}}$, $\Delta\mathcal{L} = \mathcal{L}^{(i)} - \mathcal{L}^{(i+1)}$
  $i = i + 1$
end while

2.3.2 Theoretical Results

Theorem 2 *Assuming the discrete case, the minimization $\mathcal{L}_{eSPA}$, such that Γ and Λ are stochastic matrices and W is a probability vector, has the following properties:*

1. *The optimal solution is given by a segmentation Γ that is piece-wise linear in the feature space.*
2. *Algorithm 3 approximates this solution monotonically convergent.*
3. *The cost of one iteration of algorithm 3 is $\mathcal{O}(T(D+DK+K+KM+M)+KM)$.*

Refer to Vecchi et al. (2022) [Vec+22] for the proof.

Theorem 3 *Given*

$$f(W) = \frac{\epsilon_E}{D}\sum_{d=1}^{D} W_d \, log W_d + \frac{1}{T}\sum_{d=1}^{D} W_d b_d \tag{2.15}$$

and

$$\Omega_W := \left\{ W \in \mathbb{R}^D | W \geq 0 \wedge \sum_{d=1}^{D} W_d = 1 \right\} \tag{2.16}$$

the constrained minimization of the functional

$$argmin_{W \in \Omega_W} f(W), \tag{2.17}$$

where $b \in \mathbb{R}^d$ is a constant vector, has a unique closed-form solution W^*:

$$W^* = \frac{exp(-\frac{D}{T\epsilon_E} b - \mathbb{1}_D)}{\mathbb{1}_D^T exp(-\frac{D}{T\epsilon_E} b - \mathbb{1}_D)}. \tag{2.18}$$

Proof: This proofs follows the idea and structure of Vecchi et al. (2022) [Vec+22].

The Lagrangian function corresponding to the optimization problem in equation 2.17, without considering the constraint $W \geq 0$, is given by:

$$\mathcal{L}(W, \lambda_E) = f(W) + \lambda_E(\mathbb{1}_D^T W - 1), \tag{2.19}$$

with the Lagrange multiplier λ_E.

The necessary optimality conditions are given by the Karush-Kuhn-Tucker (KKT) system of equations, which require the partial derivatives of the Lagrangian with respect to all variables to be zero.

For each component $\hat{d} = 1, \ldots, D$ of the gradient $\nabla f(W) \in \mathbb{R}^D$, the partial derivative is computed via the chain rule:

$$\frac{\partial f(W)}{\partial W_{\hat{d}}} = \frac{\epsilon_E}{D}\Big(\frac{\partial W_{\hat{d}}}{\partial W_{\hat{d}}} log W_{\hat{d}} + W_{\hat{d}} \frac{\partial \, log W_{\hat{d}}}{\partial W_{\hat{d}}}\Big) + \frac{1}{T}\frac{\partial W_{\hat{d}} b_{\hat{d}}}{W_{\hat{d}}}, \tag{2.20}$$

$$= \frac{\epsilon_E}{D}(log\, W_{\hat{d}} + 1) + \frac{1}{T} b_{\hat{d}}. \tag{2.21}$$

In vector notation, this becomes:

$$\nabla f(W) = \frac{\epsilon_E}{D}(logW + \mathbb{1}_D) + \frac{1}{T} b. \tag{2.22}$$

The derivative of the Lagrangian with respect to W is then:

$$\nabla_W \mathcal{L}(W, \lambda_E) = \nabla_W f(W) + \lambda_E \nabla_W (\mathbb{1}_D^T W - 1), \tag{2.23}$$

$$= \frac{\epsilon_E}{D}(logW + \mathbb{1}_D) + \frac{1}{T} b + \lambda_E \mathbb{1}_D. \tag{2.24}$$

Setting this derivative to zero and rearranging the terms gives:

$$\frac{\epsilon_E}{D}(logW + \mathbb{1}_D) + \frac{1}{T}b + \lambda_E \mathbb{1}_D = 0 \tag{2.25}$$

$$\Rightarrow log\, W = -\frac{D}{T\epsilon_E}b - \frac{D}{\epsilon_E}\lambda_E \mathbb{1}_D - \mathbb{1}_D \tag{2.26}$$

$$\Rightarrow W = exp\Big(-\frac{D}{T\epsilon_E}b - \frac{D}{\epsilon_E}\lambda_E \mathbb{1}_D - \mathbb{1}_D\Big) \tag{2.27}$$

$$\Rightarrow W = exp\Big(-\frac{D}{\epsilon_E}\lambda_E\Big) exp\Big(-\frac{D}{T\epsilon_E}b - \mathbb{1}_D\Big). \tag{2.28}$$

Note that the first exponential term in equation 2.28 is a scalar and the second is a vector in $\mathbb{R}^d$.

The second KKT condition requires the derivative of the Lagrangian function, with respect to λ_E, to be zero:

$$\frac{\partial L(W, \lambda_E)}{\partial \lambda_E} = \mathbb{1}_D^T W - 1 = 0. \tag{2.29}$$

Substituting 2.28 into 2.29 yields:

$$exp\Big(-\frac{D}{\epsilon_E}\lambda_E\Big)\mathbb{1}_D^T\, exp\Big(-\frac{D}{T\epsilon_E}b - \mathbb{1}_D\Big) = 1 \tag{2.30}$$

$$\Rightarrow exp\Big(-\frac{D}{\epsilon_E}\lambda_E\Big) = \frac{1}{\mathbb{1}_D^T\, exp\Big(-\frac{D}{T\epsilon_E}b - \mathbb{1}_D\Big)}. \tag{2.31}$$

Inserting 2.31 into 2.28 and denoting the solution as W^* leads to:

$$W^* = \frac{1}{\mathbb{1}_D^T\, exp\Big(-\frac{D}{T\epsilon_E}b - \mathbb{1}_D\Big)}\, exp\Big(-\frac{D}{T\epsilon_E}b - \mathbb{1}_D\Big) \tag{2.32}$$

Since the exponential function is always positive, $W^* > 0$. Furthermore, due to the normalization through the denominator, $\sum_{d=1}^{D} W_d^* = 1$. Thus $W^* \in \Omega_W$

It remains to show to show that W^* is indeed a minimizer. For this to be true, the Hessian $\nabla^2 f(W^*)$ needs to be a symmetric positive-definite quadratic form, as this would imply that f is strictly convex in some neighborhood of W^*.

Taking the second derivative of equation 2.21 for any $\hat{d}, \bar{d} = 1, \ldots, D$:

$$\frac{\partial^2 f(W)}{\partial W_{\bar{d}} \partial W_{\hat{d}}} = \frac{\partial}{\partial W_{\bar{d}}}\Big(\frac{\partial f(W)}{\partial W_{\hat{d}}}\Big) = \frac{\partial}{\partial W_{\bar{d}}}\Big(\frac{\epsilon_E}{D}(log\, W_{\hat{d}} + 1) + \frac{1}{T}b_{\hat{d}}\Big) \tag{2.33}$$

If $\bar{d} \neq \hat{d}$, this expression is zero, so the Hessian is a diagonal matrix. If $\bar{d} = \hat{d}$ we obtain that

$$\frac{\partial^2 f(W)}{\partial W_{\bar{d}} \partial W_{\hat{d}}} = \frac{\partial}{\partial W_{\bar{d}}} \left(\frac{\epsilon_E}{D} (log\, W_{\hat{d}} + 1) + \frac{1}{T} b_{\hat{d}} \right) = \frac{\epsilon_E}{D W_{\hat{d}}}. \tag{2.34}$$

This is strictly positive as $W_{\hat{d}} > 0$. Hence, all diagonal entries of the Hessian are positive and the matrix is symmetric and positive-definite. Therefore, $f(W)$ is strictly convex and W^* is the unique minimizer of equation 2.17. □

Applying theorem 3 with b_d defined as follows:

$$b_d = \sum_{t=1}^{T} (X_{dt} - (S\Gamma)_{dt})^2, \tag{2.35}$$

results in the closed-form solution for the W-Step given in equation 2.14.

2.4 GOAL

2.4.1 Algorithm

The Gauge-Optimal Approximate Learning (GOAL) algorithm, introduces by Vecchi et al. [Vec+24],is a further extension of eSPA. Its key innovation lies in optimally rotating and reducing the feature space with an optimal gauge rotation [Aro09]. After the optimal gauge rotation is determined, the algorithm identifies the prediction-optimal box discretization that minimizes the Kullback-Leibler divergence.

$$\mathcal{L}_{rec} = \frac{1}{T} \sum_{d=1}^{D} \sum_{t=1}^{T} (X_{dt} - \{RS\Gamma\}_{dt})^2 \tag{2.36}$$

$$\mathcal{L}_{\text{GOAL}} = \mathcal{L}_{KL} + \mathcal{L}_{rec} \tag{2.37}$$

In detail, the transformation from the original feature matrix $X \in \mathbb{R}^{D \times T}$ to the G-dimensional gauge-space is given by a rotation matrix $R \in \Omega_R$ (eq. 2.38), where G is a tunable hyperparameter. As in eSPA, the matrix S represents K box centers, but the centers are in gauge space rather than in feature space, so $S \in \mathbb{R}^{G \times K}$.

$$\Omega_R := \left\{ R \in \mathbb{R}^{D\times G} \middle| R^T R = I_G \right\} \tag{2.38}$$

The matrices Γ and Λ retain the same roles as in eSPA. The algorithm minimizes the loss function $\mathcal{L}_{\text{GOAL}}$ (eq. 2.37), which combines a reconstruction error and a classification loss measured by the Kullback-Leibler divergence. The reconstruction term $\mathcal{L}_{rec}$ (eq. 2.36) differs from that in eSPA (eq. 2.4) by using the rotation matrix R to map the reconstruction from gauge-space $S\Gamma$ back to feature space. The classification loss $\mathcal{L}_{KL}$ remains the same as in eSPA (eq. 2.5) and is likewise controlled by a non-negative regularization parameter $\epsilon_{CL} \geq 0$ [Vec+24].

Although in principle GOAL can be used for fuzzy segmentation, the discrete case provides closed-form solutions for each optimization step:

- S**-Step:**

$$(RS)_{dk} = \frac{\sum_{t=1}^{T} \Gamma_{kt} X_{dt}}{\sum_{t=1}^{T} \Gamma_{kt}}$$
$$S = R^T (RS)$$

- Γ**-Step:**

$$\Gamma_{kt} = \begin{cases} 1 & \text{, if } k = k' \\ 0 & \text{, else} \end{cases}$$
$$k' = \text{argmin}_{k'} \Big[-\frac{\epsilon_{CL}}{M} \sum_{m=1}^{M} \Pi_{mt} \text{max}[log(\Lambda_{mk'}), tol] + \sum_{d=1}^{D} (X_{dt} - (RS)_{dk'})^2 \Big]$$

- Λ**-Step:**

$$\Lambda_{mk} = \frac{(\Pi\, \Gamma^T)_{mk}}{\sum_{m=1}^{M} (\Pi\, \Gamma^T)_{mk}}$$

- R**-Step:** Compute $R = U\tilde{I}V$, with U and V obtained from the singular value decomposition (SVD) [Str09] of $X\, \Gamma^T\, S^T$ and $\tilde{I} \in \mathbb{R}^{D\times G}$ is a matrix with an identity matrix in the upper part and zeros elsewhere.

Algorithm 4 GOAL fit

Require: K, G, ϵ_{CL}, tol and randomly initialized, but feasible Γ and R
  $i = 1$
  $\mathcal{L}^{(1)} = \infty$, $\Delta\mathcal{L} = \infty$
  while $\Delta\mathcal{L} > tol$ **do**
    S**-Step**
    Λ**-Step**
    Γ**-Step**
    R**-Step**
    Update-Loss: $\mathcal{L}^{(i+1)} = \mathcal{L}_{\text{GOAL}}$, $\Delta\mathcal{L} = \mathcal{L}^{(i)} - \mathcal{L}^{(i+1)}$
    $i = i + 1$
  end while

Algorithm 5 GOAL prediction

Require: Unlabelled data X_ν and Λ, S and R as a result of algorithm 4
  Discretization of X_ν**:** Calculate Γ_ν by evaluating equation 2.39.
  Compute labels $\Pi_\nu = \Lambda\Gamma_\nu$

$$(\Gamma_\nu)_{kt} = \begin{cases} 1 & \text{, if } k = \text{argmin}_{k'}\left[\sum_{d=1}^{D}\left((X_\nu)_{dt} - (RS)_{dk'}\right)\right] \\ 0 & \text{, else} \end{cases} \tag{2.39}$$

2.4.2 Theoretical Results

Theorem 4 *Assuming the discrete case, the minimization $\mathcal{L}_{GOAL}$, such that Γ, Λ are stochastic matrices and R is a rotation matrix, has the following properties:*

1. *The optimal solution is given by a segmentation Γ that is piece-wise linear in the feature space.*

2. *Algorithm 4 approximates this solution monotonically convergent.*

3. *The cost of one iteration of algorithm 4 is $\mathcal{O}(T(KD + K + D + KM + M) + D(G^2 + KG) + MK)$.*

Refer to Vecchi et al. (2024) [Vec+24] for the proof.

2.5 eSPAhybrid

2.5.1 Algorithm

Although the discrete variants of eSPA, eSPA+ and GOAL offer improved time complexity, the fuzzy versions explore a larger search space. Specifically, a fuzzy box assignment Γ may yield better predictive performance but such solutions might be missed when Γ is restricted to discrete values.

The eSPAhybrid algorithm aims to unify the strengths of both approaches within a single algorithm. It starts with a discrete phase, during which optimization is performed under the constraint of a discrete Γ and thus more time-efficient. This is followed by a fuzzy phase, where optimization proceeds without that restriction. There are various strategies for balancing these two phases. In this work $\Delta\mathcal{L}$, the difference in loss between two iterations, is used as a stopping criterion, i.e. if $\Delta\mathcal{L} \leq tol$ the discrete phase is terminated. The algorithm proceeds with a fixed number of fuzzy optimization steps, controlled by the hyperparameter *num_fuzzy_steps*. Optimization in the discrete phase follows the procedure used in discrete eSPA+, while the fuzzy phase uses the procedure of fuzzy eSPA. Consequently, predictions are also made according to fuzzy eSPA. A detailed explanation on these design choices is provided in chapter 3.5

While the discrete versions of eSPA, eSPA+ and GOAl have a better time complexity, the search space of the fuzzy versions is bigger. In detail, it is possible that a box assignment Γ, which is fuzzy, leads to better predictions, but is not found since Γ is restricted to the discrete case. The idea of eSPAhybrid is to combine both approaches in one algorithm. The algorithm should start with a discrete phase in which an optimization is done for the restricted Γ, followed by a fuzzy phase in which the optimization is done in a fuzzy way. There are many ways to balance the both phases.

2.5.2 Theoretical Results

Theorem 5 *The eSPAhybrid algorithm minimizes $\mathcal{L}_{eSPA}$, such that Γ, Λ are stochastic matrices and W is a probability vector and is monotonical convergent.*

Proof: It follows from theorem 2 that the discrete phase minimizes $\mathcal{L}_{\text{eSPA}}$ monotonically, such that Γ, Λ are stochastic matrices and W is a probability vector. The fuzzy

phase uses the Γ and W that resulted of the discrete phases as an initialization. As theorem 1 states, the optimization steps used in the fuzzy phase also minimize $\mathcal{L}_{\text{eSPA}}$ monotonically and also keep Γ, Λ as stochastic matrices and W as a probability vector. As a conclusion the monotonic minimization of the whole algorithm follows. As the loss function $\mathcal{L}_{\text{eSPA}}$ is always greater or equal to zero (see A.3 in Horenko (2020) [Hor20]), the monotonic sequence of loss functions will converge.

□

The iteration costs of the phases can be taken from theorem 1 for the fuzzy phase and from 2 for the discrete phase.

3 Implementation

A central contribution of this thesis is the development of the open-source package eSPA.jl[1], which provides implementations of the discussed algorithms in Julia [Bez+17]. The package includes implementations of both the discrete and the fuzzy version of eSPA, the discrete versions eSPA+ and GOAL, and the newly introduced eSPAhybrid, as proposed in this thesis. From this point onward, references to eSPA+ and GOAL will exclusively refer to their discrete variants. The discrete version of eSPA will be also referred to as eSPAdiscrete or simply Discrete, while the fuzzy version will be denoted as eSPAfuzzy or Fuzzy, respectively. This chapter presents implementation details that are not covered in the original publications but were essential to ensure execution of the algorithms. It also discusses the decisions on the implementation of eSPAhybrid.

3.1 Interior Point methods

The Λ-Step, Γ-Step in eSPAfuzzy, as well as the W-Step in both eSPAfuzzy, are solved using the interior-point method. In eSPA.jl, these optimization steps are implemented using the JuMP.jl [Lub+23] and Ipopt.jl [Dc] packages. JuMP.jl serves as a modeling interface, used to define the optimization problem in Julia code. Ipopt.jl provides a Julia interface for Ipopt [WB06], a C++ library designed for solving non-linear optimization problems using interior-point methods.

Although Horenko (2020) [Hor20] suggests that performing a single interior point iteration is sufficient for the convergence, eSPA.jl does not limit the number of iterations per step. This choice is motivated by the observation, made during development, that allowing more iterations does not significantly impact runtime.

[1] `https://github.com/pw0lf/eSPA.jl`

3.2 Empty clusters

In all algorithms that employ a discrete box discretization, i.e. every datapoint gets assigned to exactly one box, it was occasionally observed that some clusters were empty at some point of the optimization. If this is the case, during the discrete Λ-Step (eq. 2.10), the matrix-product $\tilde{\Lambda} = \Pi\,\Gamma^T$ will lead to a matrix which has a column consisting entirely of zeros. Consequently, when computing Λ_{mk}, a division by the sum of columns of $\tilde{\Lambda}$ will be executed, leading to an error.

To address this issue, a function named "no-empty-clusters" was introduced. This function checks for empty clusters and, if any are found, identifies the largest existing cluster and reassigns half of its data points to the empty one. Alternative strategies are also imaginable, such as moving only a single point to the empty box or randomly selecting which cluster to split. However, the chosen method was preferred, with the hope that initializing the previously empty cluster with multiple points reduces the likelihood of it becoming empty again in future iterations.

The introduction of this step was necessary, but also raises concerns about its effect on the convergence properties of the algorithms. The proofs for convergence may be invalidated by this change. In fact, it was observed that the use of the "no-empty-clusters" function can lead to non-monotonic behavior in the loss function, with occasional increases in loss between iterations. This introduces additional complications: the increases results in a negative $\Delta\mathcal{L}$, which satisfies the stopping condition $\Delta\mathcal{L} \leq tol$, causing termination of the optimization process. To mitigate this issue, eSPA.jl redefines the criterion by taking the absolute change in loss: $\Delta\mathcal{L} := |\mathcal{L}^{(i)} - \mathcal{L}^{(i+1)}|$. This change has the side-effect that infinite loops could appear, if the loss value jumps between two different values. Consequently, a maximum limit on the number of iterations can be configurated, with a default value of 100.

3.3 Infinity Loss Problem

Another issue arises in the computation of the KL-divergence loss $\mathcal{L}_{KL}$. It is possible for some $m \in \{1, \ldots, M\}$ and $t \in \{1, \ldots, T\}$, that the expression $\sum_{k=1}^{K} \Lambda_{mk}\Gamma_{kt}$ evaluates to zero. In practical terms, this means that data point t is assigned only to boxes that assign zero probability to label m, resulting in an over-all probability of zero for that label. This is a valid outcome in classification. However, since $\mathcal{L}_{KL}$ includes the logarithm of this term, taking the logarithm of zero leads to negative

infinity and consequently the total loss becomes infinite. When this is the case, the algorithm terminates prematurely after two iterations with a loss of infinity.

To address this, eSPA.jl includes a safeguard during the computation of $\mathcal{L}_{KL}$. Whenever $\sum_{k=1}^{K} \Lambda_{mk}\Gamma_{kt}$ is zero, instead of adding negative infinity, -100 is added to the loss. This value should be large enough to be significant for the loss but avoids the issues that come with infinite values. It is important to note that this adjustment affects only the evaluation of the loss function which will be used for the stopping condition $\Delta\mathcal{L} \leq tol$. It does not influence the optimization iterations itself.

3.4 S-Step fuzzy

In the fuzzy version of eSPA, the S-Step uses weighted least squares to calculate S. This involves solving a linear system of equations. However, if the coefficient matrix is not full-rank, the system has no or infinite solutions [Str09]. Such rank deficiencies may arise due to structure of the data or as a result of numerical errors during computation. As a solution, eSPA.jl uses the Moore-Penrose pseudo-inverse, which provides a stable least-squares solution even when the matrix is singular [BG03].

3.5 eSPAhybrid

As outlined in 2.5, eSPAhybrid is composed of two phases: a discrete phase that follows the optimization procedure of discrete eSPA+, and a fuzzy phase that applies the optimization steps from fuzzy eSPA. This design choice was motivated by practical considerations. eSPA+ offers efficient computation due to its closed-form solutions in each step. Meanwhile, fuzzy eSPA was selected for the fuzzy phase because it is the only fuzzy variant already implemented in eSPA.jl. The reuse of already implemented components allowed a faster implementation, increases readability and allows for a better understanding and maintenance of the code. As a results, all implementation decisions described in the chapter also apply to eSPAhybrid.

4 Experiments

This chapter presents the experiments done on the five eSPA algorithms discussed throughout this thesis. All experiments are conducted using the implementations of eSPA.jl. As outlined there, several design choices were made that influence the behavior of the algorithms. Consequently, different implementation choices may lead to varying results.

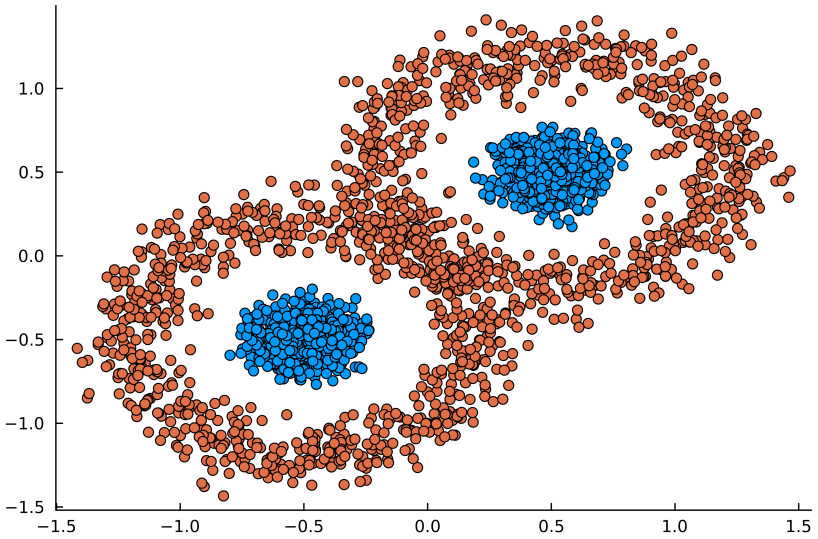


Fig. 4.1: Visualizing first two dimensions of the synthetic data. The colors indicate the label.

Two datasets were used for evaluation. The first is a synthetic dataset, introduced in Horenko (2020)[Hor20]. It is constructed such that the binary label solely depends on two specific dimensions, as illustrated in Figure 4.1. The remaining $D - 2$ dimensions are generated independently from a normal distribution. The second dataset is the Darwin dataset [Fon22], which stems from a study on the handwriting of Alzheimer's patients and a control group. In this study the participants were asked to perform 25 tasks, such as e.g. joining two points with a vertical line or copying letters. 18 features were measured for every task, including e.g. the total time spend to perform the task or the mean pressure exerted by the pen tip [Cil+18][Cil+22]. With 451 total features and only 174 samples, this dataset can be clearly categorized under the small data regime.

All experiments were executed on MOGON 2[1], using Intel Xeon Gold 6130 CPUs. Each experiment, defined as training a single model on one dataset across 50 randomly sampled hyperparameter configurations, was limited to a maximum runtime of 8 hours.

4.1 Benchmarking

4.1.1 Experimental Setup

This experiment was designed to compare the performance of all eSPA algorithms discussed in this work, with a focus on both classification accuracy and computational runtime. In addition to measuring the total runtime of each model, the duration of individual steps within the optimization process is also recorded.

The models were evaluated on both datasets, with six variations of the synthetic dataset differing in sample size and feature dimensionality. Concretely all combinations of $T \in \{100, 1000\}$ and $D \in \{4, 10, 30\}$ were used. For both datasets a simple train-test split is applied, where 80% of the data is used for training and the remaining 20% for testing. Model accuracy is evaluated on the test set.

For each algorithm and dataset, 50 sets of hyperparameters were randomly sampled from predefined search spaces. Specifically, all models used the following ranges: $K \in [5, 100]$, $\epsilon_{CL} \in [10^{-1}, 100]$ and $tol \in [10^{-10}, 10^{-1}]$. All algorithms except GOAL used $\epsilon_E \in [10^{-2}, 10]$. The GOAL algorithm used the gauge-space dimension $G \in [1, D-1]$, and eSPAhybrid used an additional hyperparameter num_fuzzy_steps $\in [3, 8]$. The search spaces were chosen partially informed by Horenk (2020) (section 2.3 in [Hor20]) and partly by practical experience gained during the development and testing of eSPA.jl. It should be noted that alternative search spaces may yield other results.

[1]Parts of this research were conducted using the supercomputer MOGON 2 and/or advisory services offered by Johannes Gutenberg University Mainz (hpc.uni-mainz.de), which is a member of the AHRP (Alliance for High Performance Computing in Rhineland Palatinate, www.ahrp.info) and the Gauss Alliance e.V. The authors gratefully acknowledge the computing time granted on the supercomputer MOGON 2 at Johannes Gutenberg University Mainz (hpc.uni-mainz.de).

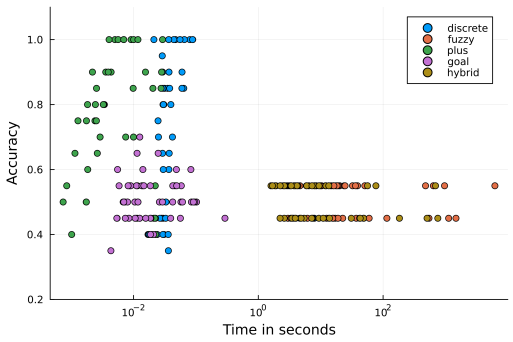
4.1.2 Results on synthetic Data

Figure 4.2 presents consistent results across all six experiments conducted on the synthetic dataset. Among the evaluated models, eSPA+ and eSPAdiscrete achieve the highest accuracies, with eSPA+ clearly outperforming in terms of runtime. GOAL shows lower accuracy compared to these two models and requires a runtime comparable to eSPAdiscrete.

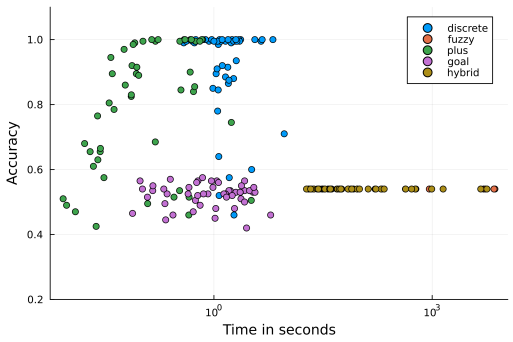
A notable observation is that eSPAfuzzy and eSPAhybrid typically yield only one or two distinct accuracy values, resulting in horizontal lines in the plots. These models also exhibit the longest runtimes by far, up to five orders of magnitude slower than eSPA+. Only when using $T = 1000, D = 30$, the runtime and accuracy difference between eSPAfuzzy/eSPAhybrid and eSPAdiscret/GOAL are less prominent.

Overall, the results strongly favor eSPA+, as it consistently matches or exceeds the accuracy of eSPAdiscrete while being significantly faster. The high runtimes of eSPAfuzzy and eSPAhybrid are expected, as these algorithms involve fewer steps using closed-form solutions for its optimization. However, the notably lower accuracies of these two models are unexpected, given that they are solving the same underlying optimization problem as the other models, excluding GOAL.

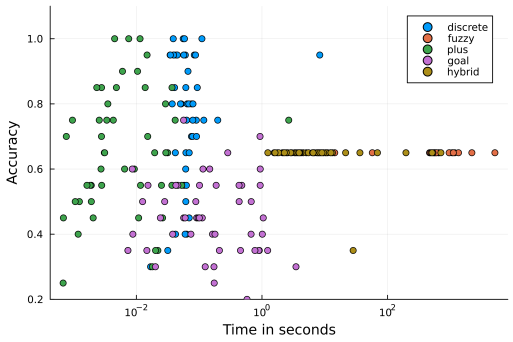
Table 4.1 further shows the distribution of runtime across the individual optimizations steps. As expected, the W-Step dominates the runtime in eSPAdiscrete, since it is the only step requiring the usw of interior-point methods. This likely explains the difference in runtime between ePSAdiscrete and eSPA+. For both GOAL and eSPAfuzzy, the Γ-Step is the most time-intensive. eSPA+ stands out as the only model without a single dominant bottleneck: in four out of six experiments, the Γ-Step takes the most time, while in the remaining two, the W-Step is the most time-consuming. eSPAhybrid is left-out of table 4.1, as insights on the runtime behavior of eSPAhybrid are provided in another experiment (see Section 4.3), which specifically focuses on this algorithm.



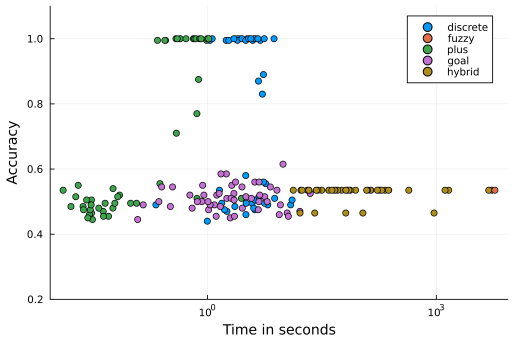
(a) $D = 4, T = 100$



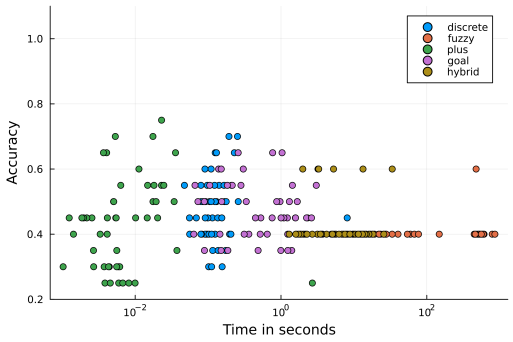
(b) $D = 4, T = 1000$



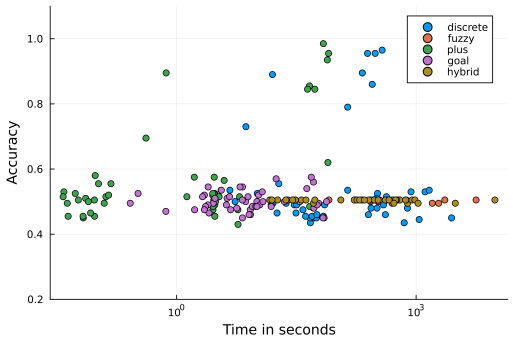
(c) $D = 10, T = 100$



(d) $D = 10, T = 1000$



(e) $D = 30, T = 100$



(f) $D = 30, T = 1000$

Fig. 4.2: Accuracy-Runtime trade-off on synthetic dataset for the tested hyperparameter configurations.

	Discrete	Plus	Fuzzy	GOAL
Initialization	0.16	1.19	<0.01	0.55
S-Step	2.20	5.69	6.72	5.88
Γ-Step	22.43	**66.22**	**60.69**	**83.48**
W-Step	**71.50**	12.14	10.32	-
R-Step	-	-	-	1.27
Λ-Step	0.46	0.88	22.19	0.46
No-Empty-Cluster	0.93	5.20	-	1.72
Loss	1.24	4.16	0.05	3.09

(a) $D = 4, T = 100$

	Discrete	Plus	Fuzzy	GOAL
Initialization	0.03	0.30	<0.01	0.13
S-Step	2.94	4.77	1.47	4.93
Γ-Step	26.13	**51.07**	**68.88**	**89.91**
W-Step	**67.93**	38.69	0.36	-
R-Step	-	-	-	0.47
Λ-Step	0.39	0.43	29.29	0.29
No-Empty-Cluster	1.43	1.77	-	1.11
Loss	1.00	2.54	<0.01	2.68

(b) $D = 4, T = 1000$

	Discrete	Plus	Fuzzy	GOAL
Initialization	0.10	1.66	<0.01	0.26
S-Step	3.49	10.87	1.34	7.27
Γ-Step	18.19	**61.68**	**81.99**	**86.04**
W-Step	**74.67**	13.44	2.15	-
R-Step	-	-	-	1.15
Λ-Step	0.41	0.73	14.51	0.38
No-Empty-Cluster	0.77	5.03	-	0.70
Loss	0.89	3.14	<0.01	2.03

(c) $D = 10, T = 100$

	Discrete	Plus	Fuzzy	GOAL
Initialization	0.03	0.29	<0.01	0.06
S-Step	2.15	6.97	0.66	6.58
Γ-Step	7.54	37.61	**96.90**	**90.40**
W-Step	**89.35**	**51.24**	0.11	-
R-Step	-	-	-	0.29
Λ-Step	0.23	0.46	2.33	0.13
No-Empty-Cluster	0.29	1.39	-	0.61
Loss	0.35	1.72	<0.01	1.61

(d) $D = 10, T = 1000$

	Discrete	Plus	Fuzzy	GOAL
Initialization	0.07	1.26	<0.01	0.12
S-Step	3.65	22.00	8.07	8.95
Γ-Step	9.17	**51.33**	**64.46**	**86.70**
W-Step	**85.71**	15.89	8.69	-
R-Step	-	-	-	1.63
Λ-Step	0.39	0.61	18.77	0.23
No-Empty-Cluster	0.26	3.88	-	0.38
Loss	0.40	3.07	<0.01	0.94

(e) $D = 30, T = 100$

	Discrete	Plus	Fuzzy	GOAL
Initialization	<0.01	0.12	<0.01	0.01
S-Step	0.63	4.85	8.80	9.90
Γ-Step	0.75	14.91	**39.55**	**80.78**
W-Step	**98.28**	**77.45**	26.93	-
R-Step	-	-	-	5.81
Λ-Step	<0.01	0.19	24.58	0.03
No-Empty-Cluster	0.05	0.24	-	0.18
Loss	0.26	2.08	0.13	3.11

(f) $D = 30, T = 1000$

Tab. 4.1: Average percentage of total runtime by algorithm step on the synthetic dataset.

4.1.3 Results Darwin

The results obtained on the Darwin dataset are similar to those observed for the synthetic dataset. eSPA+ remains the fastest algorithm, followed by eSPAdiscrete. However, in contrast to the synthetic dataset, eSPAdiscrete achieves both a higher maximum accuracy and a higher average accuracy than eSPA+. The other three models, eSPAfuzzy, eSPAhybrid, and GOAL, continue to perform significantly worse

in terms of both accuracy and runtime. A breakdown of the runtime across individual optimization steps reveals almost the same bottlenecks identified on the synthetic experiments. Only eSPAfuzzy shows a different bottleneck with the W-Step.

Table 4.3 provides a comparison of the best accuracies achieved by eSPAdiscrete and eSPA+ with the performance of various machine learning models, evaluated on Darwin in [Cil+18]. It is important to emphasize that the experimental protocols differ between these works. The Darwin dataset consists of features extracted from 25 handwriting tasks, with 18 features per task. In the pipeline described in [Cil+18], a separate classifier is trained for each individual task. For each individual classifier, hyperparameters are tuned, and the best configuration is selected. The reported results correspond to the mean and standard deviation of 25 such models, each evaluated over 20 random sample splits. In contrast, the experiments conducted for eSPAdiscrete and eSPA+ involved a single model on all 451 features with a fixed train-test split. For each algorithm, 50 hyperparameter configurations were randomly sampled and evaluated. Consequently, the reported accuracies for eSPAdiscrete and eSPA+ reflect the best-performing model found during the experiment, not an average over multiple runs. Despite these differences, the results should allow for a reasonably comparison.

The best accuracy achieved by eSPAdiscrete is 91.43%, which falls within the standard deviation range reported for the Random forest and lies near the upper bound for the Multilayer perceptron. The best results for eSPA+ lies within the standard deviation range of all models except k-Nearest Neighbor. These findings suggest that eSPA+ performs comparable to the other classifiers, while eSPAdiscrete reaches the performance level of the top-performing classifiers. As the classifiers in [Cil+18] are trained using only the features from individual tasks, that is only $\frac{1}{25}$ of the total available features, per model. One might argue that this excludes relevant information, potentially making the classification more difficult. Contrarily, this restriction could simplify the task by reducing the number of features and thus the risk of overfitting. GOAL, eSPAfuzzy and eSPAhybrid are excluded from this comparison, as their accuracies fall clearly below those of eSPAdiscrete and eSPA+, as illustrated in figure 4.3.

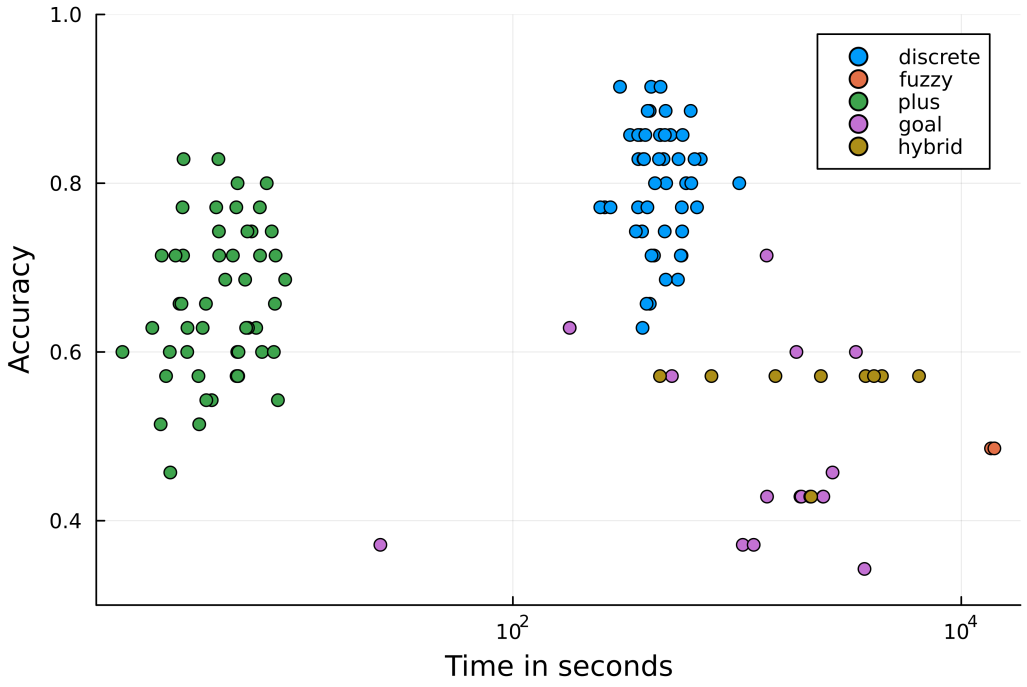


Fig. 4.3: Accuracy-Runtime trade-off on Darwin for the tested hyperparameter configurations.

	Discrete	Plus	Fuzzy	GOAL
Initialization	<0.01	0.12	<0.01	0.02
S-Step	0.20	13.85	3.52	1.89
Γ-Step	0.11	14.45	34.67	**93.11**
W-Step	**99.50**	**63.90**	**58.29**	-
R-Step	-	-	-	4.77
Λ-Step	<0.01	0.18	3.46	<0.01
No-Empty-Cluster	<0.01	0.12	-	<0.01
Loss	0.17	7.27	0.06	0.19

Tab. 4.2: Average percentage of total runtime by algorithm step on Darwin

	eSPAdiscrete	eSPA+	RF	LR	KNN
Accuracy	91.43	82.86	88.29 ± 4.90	81.86 ± 7.20	71.43 ± 8.34

LDA	GNB	SVM	DT	MLP	LVQ
72.14 ± 8.44	85.00 ± 5.47	79.00 ± 7.55	78.57 ± 7.21	83.14 ± 7.97	77.43 ± 7.41

Tab. 4.3: Accuracy on Darwin. For eSPAdiscrete and eSPA+ the best result is reported. For the remaining models, the mean accuracy and standard deviation across 20 train-test splits are reported, based on table 6 from [Cil+18]. RF = Random Forest, LR = Logistic regression, KNN = K-Nearest Neighbor, LDA = Linear Discriminant Analysis, GNB = Gaussian Naive Bayes, SVM = Support Vector Machine, DT = Decision Tree, MLP = Multilayer Perceptron, LVQ = Learning Vector Quantization.

4.1.4 Limitations of the experimental protocol

As analyzing this experiment, several weaknesses of the experimental setup become apparent. The comparison of eSPA models to the machine learning models on the Darwin dataset is not optimal. While the models, reported by [Cil+18], were evaluated across 20 different random data splits, the eSPA models were only trained and tested on a single split. Conducting multiple runs with varied splits for the eSPA models would lead to more robust and comparable results.

In general, benchmarking across multiple data splits would be beneficial, as it reduces reliance on a single random partition. A particular split might, by chance, make the classification harder than others. However, it can be excluded that the used data split is unbalanced regarding the class labels. Across all datasets, the largest observed class imbalance was found in a test set of the synthetic dataset with $D = 4, T = 100$, showing a 60/40 split between the two classes.

Some of the observed results were unexpected and may be caused by the experimental protocol. For instance, GOAL performed significantly worse in terms of accuracy compared to eSPA+ and eSPAdiscrete. One possible explanation could be the chosen range for the hyperparameter G. Although Vecchi et al. (2024) [Vec+24] suggest that G should be significantly lower than D, the experiments sample G from $[1, D-1]$. However, as illustrated in table 4.4, the correlations between G and accuracy show no clear link.

Another plausible explanation is the imposed limit on the number optimization iterations. Table 4.5 indicates that a substantial proportion of GOAL runs terminated due to the limit. Interestingly, it can be observed that the number of iterations appears to

scale with the feature dimension D. In retrospect this parameter should have been adjusted for GOAL. Other eSPA variants were not affected by this limitation. The large number of iterations could also explains the relatively high runtimes, despite using no interior-point method.

	4_100	4_1000	10_100	10_1000	30_100	30_1000	Darwin
Coeff.	-0.11	-0.47	-0.17	0.07	0.03	0.09	0.27

Tab. 4.4: Pearson correlation coefficient between G and accuracy [Was10].

	4_100	4_1000	10_100	10_1000	30_100	30_1000	Darwin
Mean number of iterations	10.0	36.8	49.7	58.5	74.2	73.0	100
Percentage of runs with 100 iterations	2.0	4.0	40.0	44.0	64.0	62.0	100.0

Tab. 4.5: Mean number of iterations and percentage of runs with 100 iterations for GOAL.

The performance of eSPAfuzzy and eSPAhybrid was also weaker than expected. A likely explanation is the time constraint of eight hours per experiment. Due to their high computational complexity, eSPAfuzzy and eSPAhybrid were unable to complete the full set of 50 hyperparameter trials. Specifically, eSPAfuzzy misses out on 184 out of 350 planned runs, over 50% in total. The most extreme case occurred on the Darwin dataset, where only two runs were completed before the time limit was reached. eSPAhybrid missed only 20 out of 350 runs. It is plausible, especially in the case of eSPAfuzzy, that better performance could have been achieved with more trials.

In the case of eSPAhybrid, the low number of fuzzy steps, ranging from three to eight, might also have limited performance. Increasing this number could improve results, but as shown in section 4.3, the fuzzy phase already takes up the majority of runtime. Additional iterations would increase this phenomenon.

Overall, the selection of hyperparameter ranges plays a critical role in the outcome of these experiments, as there is no guarantee that the chosen ranges are near optimal. The influence of hyperparameters will be further explored in the following experiment (sec. 4.2).

4.2 Hyperparameter Sensitivity

4.2.1 Experimental Setup

This experiment investigates how sensitive the performance of eSPA+ is on the choice of hyperparameters. Selecting optimal hyperparameters is a non-trivial task. A high sensitivity would make this process even more challenging. Additionally, all eSPA algorithms present a unique situation regarding its hyperparemeters, as many of these are dependent on each other. For instance, there is a direct relationship between ϵ_{CL} and ϵ_E, as they weight difference components of the loss function. If both are scaled by the same factor, the balance between classification accuracy and feature selection remains unchanged, only their relative influence to the reconstruction term is affected. Since the reconstruction terms does not have an own parameter, this is also the only way to balance the loss terms. This also influences the tol parameter indirectly. When ϵ_{CL} and ϵ_E are increased, the overall magnitude of the loss also increases. In such cases the same tolerance may cause the optimization to stop earlier.

The experiment is conducted on the Darwin dataset and focuses solely on eSPA+. As shown in earlier experiments, eSPA+ has the shortest runtime on this dataset, which allows a larger number of hyperparameters to be tested within reasonable time.

As a reference point, the best-performing hyperparameter combination of eSPA+ on Darwin is used:

- $K = 34$
- $\epsilon_{CL} = 0.95$
- $\epsilon_E = 0.013$
- $tol = 3.47 \cdot 10^{-8}$.

In the experiment three out of four parameters are kept fixed while the other parameter is varied. This ensures that the outcome is only affected by the modified parameter, preventing the previously mentioned interdependency from influencing it. Specifically the following ranges are used:

- $K \in [14, 54]$
- 1000 values for $\epsilon_{CL} \in [0.095, 9.5]$

- 1000 values for $\epsilon_E \in [0.0013, 0.13]$
- 1000 values for $tol \in [3.47 \cdot 10^{-10}, 3.47 \cdot 10^{-6}]$.

For each configuration, both accuracy and runtime are recorded. To evaluate the sensitivity, two metrics are used: a rolling standard deviation and a rolling maximum difference. For a given window size, the standard deviation and the maximum difference of accuracy values are computed within each window of neighboring hyperparameter values. This window then slides across the entire range, providing a local measure of how much performance varies with small changes in the parameter.

4.2.2 Results

The results of this experiment indicate that eSPA+ is highly sensitive to its hyperparameters. Figures 4.4 and 4.5 illustrate how both accuracy and runtime can fluctuate when individual hyperparameters are pertubated.

For accuracy, no clear trend emerges. Modifying a parameter in either direction does not consistently improve performance. In contrast, runtime exhibits some mild trends. Increasing the number of boxes K tends to reduce runtime, while increasing ϵ_{CL} tends to increase it. Although the trend for K seems counterintuitive, as the runtime of the optimizations steps scales with K per iteration step. But the total runtime is also dependent of the number of iterations.

Despite these variations, runtime sensitivity appears relatively low. The rolling standard deviation is only about 1.7 seconds, which is small compared to the average runtime of around 7 seconds (table 4.7). In contrast, even small changes in hyperparmeters results in average in an expected change of 7 percentage points in accuracy, while the maximum observed difference reaches up to 40 percentage points (table 4.6). As already a change of 7% is significant, this highlights the importance of hyperparameter selection.

While this experiment focuses solely on eSPA+, similar sensitivity may affect other eSPA variants, given their structural similarities. If this is the case, hyperparameter selection will be an even greater challenge, due to the increased time needed for testing configurations of the other models.

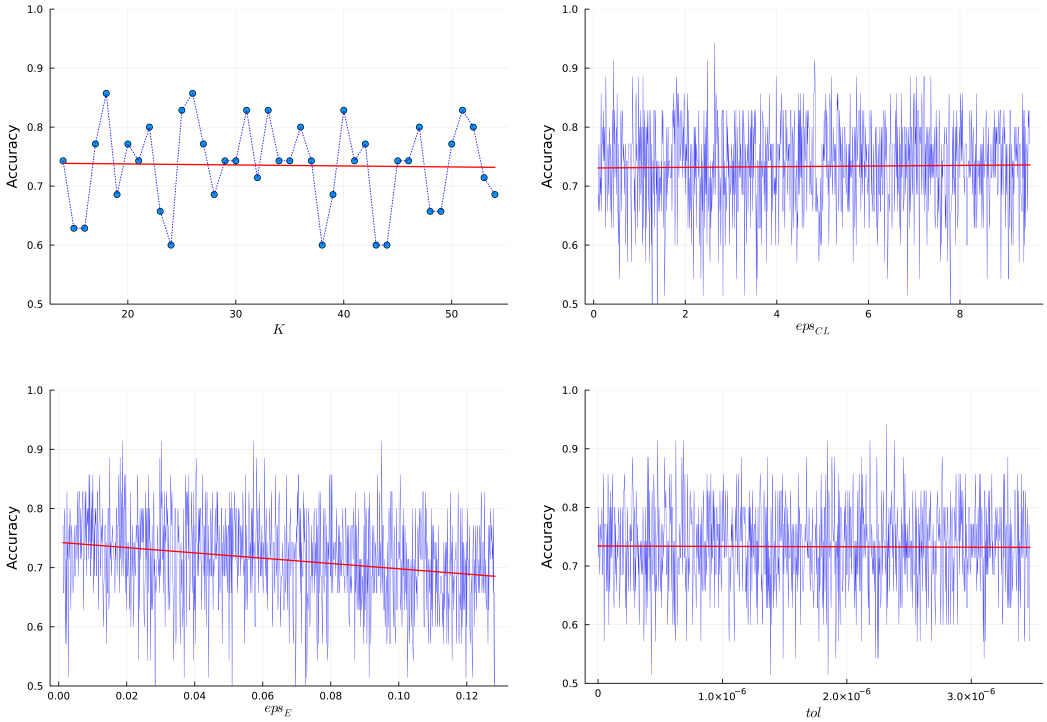


Fig. 4.4: Accuracy of eSPA by hyperparameter. The redline shows the linear regression on the data.

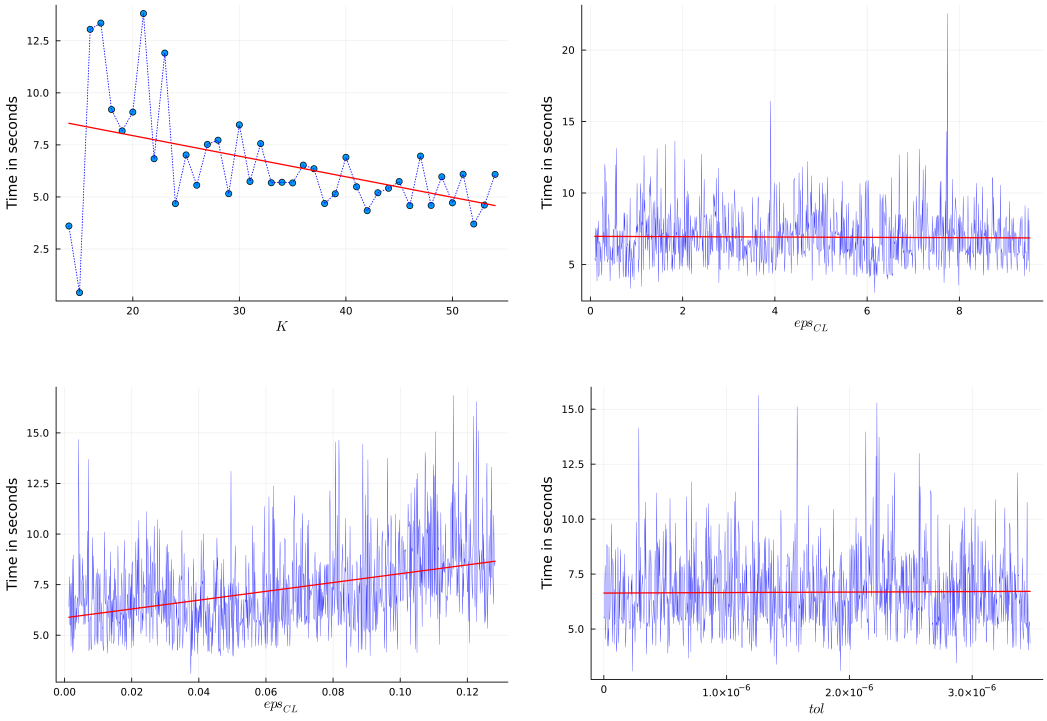


Fig. 4.5: Runtime in seconds of eSPA by hyperparameter. The redline shows the linear regression on the data.

		Std			Max diff		
hyperparameter	window size	mean	max	min	mean	max	min
K	3	0.07	0.14	0.03	0.13	0.26	0.06
eps_CL	0.095	0.07	0.13	0.03	0.23	0.40	0.09
eps_E	0.0013	0.07	0.12	0.02	0.22	0.37	0.09
tol	3.5e-8	0.07	0.13	0.03	0.23	0.40	0.06

Tab. 4.6: Rolling standard deviation and Rolling maximum difference of eSPA accuracy by hyperparameter.

		Std			Max diff		
hyperparameter	window size	mean	max	min	mean	max	min
K	3	1.70	7.39	0.02	3.20	12.94	0.03
eps_CL	0.095	1.72	5.71	0.44	5.45	18.84	1.54
eps_E	0.0013	1.79	3.77	0.51	5.58	10.89	1.63
tol	3.5e-8	1.64	3.74	0.62	5.12	11.45	2.16

Tab. 4.7: Rolling standard deviation and Rolling maximum difference of eSPA runtime in seconds by hyperparameter.

4.3 eSPAhybrid

4.3.1 Experimental setup

This experiment investigates whether the additional fuzzy phase in eSPAhybrid provides any benefit. Specifically, the difference in accuracy and runtime between the prediction using the parameters obtained after the discrete phase versus those obtained after the full process, including the fuzzy phase, is compared. As in the first experiment, the synthetic dataset is used with $T \in \{100, 1000\}$ and $D \in \{4, 10, 30\}$. For each setup, 50 hyperparameter combinations are tested.

4.3.2 Results

As shown in the boxplot (fig. 4.6), the impact of the additional fuzzy steps varies depending on the selected hyperparameters. In some instances, these steps lead to improved accuracy compared to using only the discrete phase. However, in most cases, the additional fuzzy phase results in a significant drop in performance, sometimes exceeding a decrease of 40 percentage points. The only noticeable exception is when $T = 1000, D = 30$ are chosen, where, except of some outliers, the fuzzy phase shows no effect.

Table 4.8 highlights the percentage of the computational cost that the fuzzy phase is taking. It shows that the fuzzy steps account for more than 99% of the total runtime, again with $T = 1000, D = 30$ as an exception.

These findings clearly suggest that eSPAhybrid should not be used instead of eSPA+. While increasing the number of fuzzy steps beyond the tested range of 3 to 8 might improve performance, this would also come at the cost of even longer runtimes.

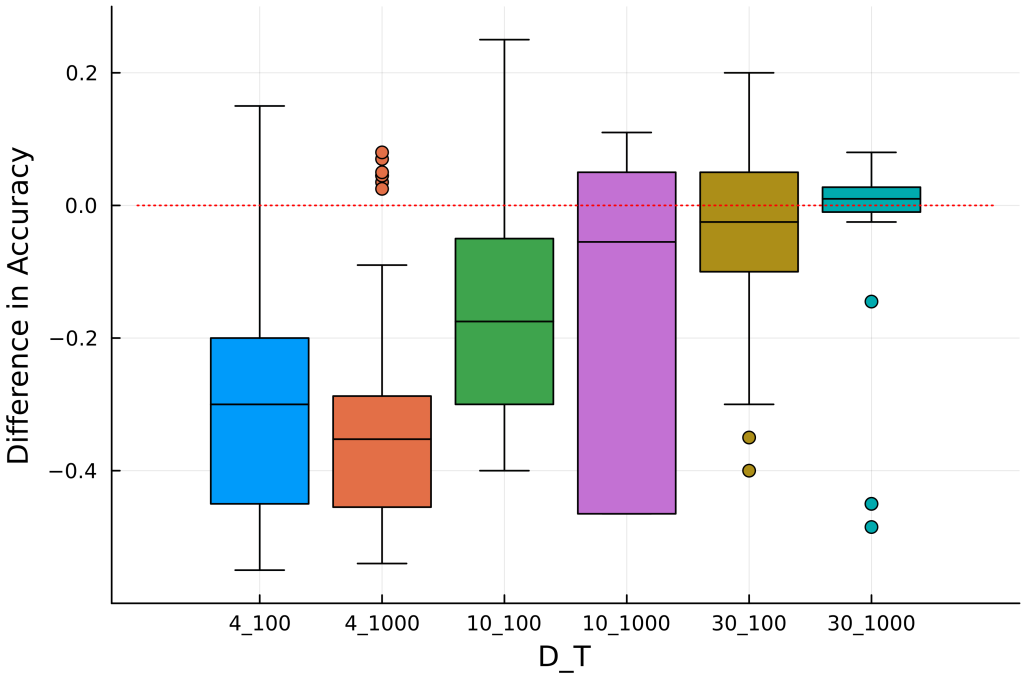


Fig. 4.6: Boxplot of difference in accuracy between eSPAhybrid with and without the fuzzy phase on the synthetic dataset with D_T. Negative values indicate that skipping the fuzzy phase results in higher accuracy. Each boxplot represents the distribution across 50 hyperparameter configurations. The box represents the middle 50% of values with the horizontal line showing the median. The whiskers extend to the smallest and largest value within 1.5 times the range of the box and outliers beyond are shown as individual points [MTa78]

N	100			1000		
D	4	10	30	4	10	30
mean	99.33	99.37	99.36	99.40	99.20	93.56
Std	2.78	1.58	1.27	0.50	0.82	7.59

Tab. 4.8: Runtime of fuzzy phase as percentage of total runtime for eSPAhybrid.

5 Conclusion

This thesis aimed at investigating the family of eSPA algorithms, which are specifically designed for classification in the small data regime. Theoretical foundations were explored, including proofs of monotonic convergence and computational complexity. As part of this work, a new variant, eSPAhybrid, was proposed, aiming to combine the strengths of discrete and fuzzy segmentation. Theoretical results were extended to this new model.

A central contribution of this thesis was the development of the open-source Julia package eSPA.jl, which implements all discussed algorithms. A special focus was made on implementation challenges, such as handling empty clusters and infinite loss values, and solutions proposed. Although these solutions introduced further limitations, such as non-monotonic loss sequences or early termination.

Experiments were conducted on both synthetic and real-world datasets. Benchmarking results revealed eSPA+ and eSPAdiscrete as the best-performing models, with eSPA+ having the faster runtime but eSPAdiscrete the higher accuracy. In contrast, GOAL, eSPAfuzzy and eSPAhybrid perform significantly worse. Runtime analysis shows that the primary bottleneck was the use of interior-point methods, whereas for GOAL the high number of iterations was the limiting factor. The poor accuracy of these models remains mostly unexplained, but some possible explanations were given. The experiments show that eSPA+ is highly sensitive to hyperparameters and the similarity of the models suggests that the other models also suffer under this sensitivity. While the proposed eSPAhybrid model theoretically combines the strengths of both segmentation approaches, the experiments show that the additional fuzzy phase leads to higher runtime with mostly lower accuracy.

While the experimental part showed meaningful insights, the experimental protocol itself has several limitations. One major category of limitations lies in the choice of parameters, such as the runtime constraint and the maximum limit for the number of iterations, which directly impacts the performance of certain models. Another important limitation is the inconsistency between the experimental setup used in this work and that used for the other machine learning models reported by [Cil+18], making direct comparison less meaningful.

Several directions for future work are worth pursuing. Refining the experimental protocol could lead to more accurate results. The implementation in eSPA.jl still has considerable room for optimization. The most prominent starting point would be accelerating the interior-point method, which was identified as the primary bottleneck. Improving this component would substantially reduce computational costs. As a side effect, this would allow for more experiments regarding hyperparameter sensitivity. In that regard, it remains an open question whether other models in the eSPA family show similar sensitivity as observed for eSPA+.

Bibliography

[Aro09] Raman Arora. “On Learning Rotations”. In: *Advances in Neural Information Processing Systems*. Ed. by Y. Bengio, D. Schuurmans, J. Lafferty, C. Williams, and A. Culotta. Vol. 22. Curran Associates, Inc., 2009 (cit. on p. 12).

[BG03] Adi Ben-Israel and Thomas Greville. *Adi Ben-Israel and Thomas N.E. Greville, Generalized Inverses: Theory and Applications, ISBN 0-387-00293-6*. Jan. 2003 (cit. on p. 19).

[Bez+17] Jeff Bezanson, Alan Edelman, Stefan Karpinski, and Viral B Shah. “Julia: A fresh approach to numerical computing”. In: *SIAM review* 59.1 (2017), pp. 65–98 (cit. on p. 17).

[Bis06] C.M. Bishop. *Pattern recognition and machine learning*. Vol. 4. Springer New York, 2006 (cit. on p. 1).

[Cil+22] Nicole D. Cilia, Giuseppe De Gregorio, Claudio De Stefano, et al. “Diagnosing Alzheimer’s disease from on-line handwriting: A novel dataset and performance benchmarking”. In: *Engineering Applications of Artificial Intelligence* 111 (2022), p. 104822 (cit. on p. 21).

[Cil+18] Nicole Dalia Cilia, Claudio De Stefano, Francesco Fontanella, and Alessandra Scotto Di Freca. “An Experimental Protocol to Support Cognitive Impairment Diagnosis by using Handwriting Analysis”. In: *Procedia Computer Science* 141 (2018). The 9th International Conference on Emerging Ubiquitous Systems and Pervasive Networks (EUSPN-2018) / The 8th International Conference on Current and Future Trends of Information and Communication Technologies in Healthcare (ICTH-2018) / Affiliated Workshops, pp. 466–471 (cit. on pp. 21, 26, 28, 37).

[CT06] Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory 2nd Edition (Wiley Series in Telecommunications and Signal Processing)*. Wiley-Interscience, July 2006 (cit. on pp. 4, 5).

[Dc] Oscar Dowson and contributors. *Ipopt.jl: Julia interface to Ipopt*. `https://github.com/jump-dev/Ipopt.jl`. Accessed: 2025-06-03 (cit. on p. 17).

[Fon22] Francesco Fontanella. *DARWIN*. UCI Machine Learning Repository. DOI: https://doi.org/10.24432/C 2022 (cit. on p. 21).

[Ger+20] S. Gerber, L. Pospisil, M. Navandar, and I. Horenko. “Low-cost scalable discretization, prediction, and feature selection for complex systems”. In: *Science Advances* 6.5 (2020), eaaw0961. eprint: `https://www.science.org/doi/pdf/10.1126/sciadv.aaw0961` (cit. on pp. 2, 3).

[HTF09] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The elements of statistical learning: data mining, inference and prediction*. 2nd ed. Springer, 2009 (cit. on p. 3).

[HS97] Sepp Hochreiter and Jürgen Schmidhuber. "Long Short-Term Memory". In: *Neural Computation* 9 (Nov. 1997), pp. 1735–1780 (cit. on p. 1).

[Hor20] Illia Horenko. "On a Scalable Entropic Breaching of the Overfitting Barrier for Small Data Problems in Machine Learning". In: *Neural Computation* 32.8 (Aug. 2020), pp. 1563–1579. eprint: `https://direct.mit.edu/neco/article-pdf/32/8/1563/1864989/neco\_a\_01296.pdf` (cit. on pp. 1, 2, 4, 5, 7, 8, 16, 17, 21, 22).

[Lub+23] Miles Lubin, Oscar Dowson, Joaquim Dias Garcia, et al. "JuMP 1.0: Recent improvements to a modeling language for mathematical optimization". In: *Mathematical Programming Computation* 15 (2023), pp. 581–589 (cit. on p. 17).

[MTa78] Robert Mcgill, John W. Tukey, and Wayne A. Larsen and. "Variations of Box Plots". In: *The American Statistician* 32.1 (1978), pp. 12–16. eprint: `https://www.tandfonline.com/doi/pdf/10.1080/00031305.1978.10479236` (cit. on p. 34).

[Mur22] Kevin P. Murphy. *Probabilistic Machine Learning: An introduction*. MIT Press, 2022 (cit. on p. 1).

[NW06] Jorge Nocedal and Stephen J. Wright. *Numerical optimization*. 2. ed. Springer series in operations research and financial engineering. New York, NY: Springer, 2006. XXII, 664 (cit. on p. 6).

[SL12] G.A.F. Seber and A.J. Lee. *Linear Regression Analysis*. Wiley Series in Probability and Statistics. Wiley, 2012 (cit. on p. 6).

[Str09] Gilbert Strang. *Introduction to Linear Algebra*. Fourth. Wellesley, MA: Wellesley-Cambridge Press, 2009 (cit. on pp. 13, 19).

[Vec+24] Edoardo Vecchi, Davide Bassetti, Fabio Graziato, Lukáš Pospíšil, and Illia Horenko. "Gauge-Optimal Approximate Learning for Small Data Classification". In: *Neural Computation* 36.6 (May 2024), pp. 1198–1227. eprint: `https://direct.mit.edu/neco/article-pdf/36/6/1198/2467115/neco\_a\_01664.pdf` (cit. on pp. 2, 12–14, 28).

[Vec+22] Edoardo Vecchi, Lukáš Pospíšil, Steffen Albrecht, Terence J. O'Kane, and Illia Horenko. "eSPA+: Scalable Entropy-Optimal Machine Learning Classification for Small Data Problems". In: *Neural Computation* 34.5 (2022), pp. 1220–1255 (cit. on pp. 1–4, 8–10).

[WB06] Andreas Wächter and Lorenz T. Biegler. "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming". In: *Mathematical Programming* 106.1 (2006), pp. 25–57 (cit. on p. 17).

[Was10] Larry Wasserman. *All of statistics : a concise course in statistical inference*. New York: Springer, 2010 (cit. on p. 29).

List of Figures

4.1 Visualizing first two dimensions of the synthetic data. The colors indicate the label. 21
4.2 Accuracy-Runtime trade-off on synthetic dataset for the tested hyperparameter configurations. 24
4.3 Accuracy-Runtime trade-off on Darwin for the tested hyperparameter configurations. 27
4.4 Accuracy of eSPA by hyperparameter. The redline shows the linear regression on the data. 32
4.5 Runtime in seconds of eSPA by hyperparameter. The redline shows the linear regression on the data. 32
4.6 Boxplot of difference in accuracy between eSPAhybrid with and without the fuzzy phase on the synthetic dataset with D_T. 34

List of Tables

4.1 Average percentage of total runtime by algorithm step on the synthetic dataset. 25
4.2 Average percentage of total runtime by algorithm step on Darwin 27
4.3 Accuracy on Darwin. 28
4.4 Pearson correlation coefficient between G and accuracy [Was10]. 29
4.5 Mean number of iterations and percentage of runs with 100 iterations for GOAL. 29
4.6 Rolling standard deviation and Rolling maximum difference of eSPA accuracy by hyperparameter. 33
4.7 Rolling standard deviation and Rolling maximum difference of eSPA runtime in seconds by hyperparameter. 33
4.8 Runtime of fuzzy phase as percentage of total runtime for eSPAhybrid. . . 35

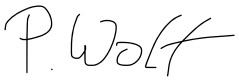
List of Algorithms

1 eSPA fit 6
2 eSPA prediction 7
3 eSPA+ fit 9
4 GOAL fit 14
5 GOAL prediction 14

Declaration

I hereby declare that I have written the present thesis independently and without use of other than the indicated means. I also declare that to the best of my knowledge all passages taken from published and unpublished sources have been referenced. The paper has not been submitted for evaluation to any other examining authority nor has it been published in any form whatsoever.

Mainz, June 4, 2025

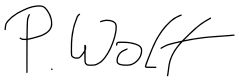


Philipp Wolf

I hereby declare that in the process of this bachelor's thesis, I have made us of artifical intelligence tools in the following way:

- I used ChatGPT by OpenAI to support formulation by asking for suggestions how to formulate a specific sentence or by asking for synonyms for certain words.
- I used DeepL for translations between german and english.
- I used ChatGPT by OpenAI while coding for specific questions regarding the Julia Programming Language.
- AI tools were not used for generating or analyzing experimental results, writing proofs, or conducting any core scientific work.

Mainz, June 4, 2025



Philipp Wolf