Probabilistic Generalized Slow Feature Analysis for State Estimation and Classification

A Project Report

submitted by

VISHAL RISHI MK

in partial fulfilment of requirements
for the award of the dual degree of

BACHELOR OF TECHNOLOGY AND MASTER OF TECHNOLOGY

with specialization in

DATA SCIENCE

DEPARTMENT OF CHEMICAL ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY MADRAS

JUNE 2023

THESIS CERTIFICATE

This is to certify that the thesis titled **Probabilistic Generalized Slow Feature Analysis for State Estimation and Classification**, submitted by **Vishal Rishi MK**, to the Indian Institute of Technology, Madras, for the award of **Dual Degree**, is a bonafide record of the research work done by him under our supervision. The contents of this thesis, in full or in parts, have not been submitted to any other Institute or University for the award of any degree or diploma.

Prof. Arun K. Tangirala
Research Guide
Professor
Dept. of Chemical Engineering
IIT-Madras, 600 036

Place: Chennai

Date: 12th June 2023

ACKNOWLEDGEMENTS

First of all, I would like to thank my guide Dr. Arun K. Tangirala for providing me an opportunity to work with him, and I am extremely grateful for his constant guidance, encouragement, and support throughout the year by scheduling regular meetings. I appreciate that he would always listen to me and give his valuable insights, despite his busy schedule. I would like to thank my parents who are the reason for my being, becoming, and more. I would like to thank all my close friends who were always there to support me.

ABSTRACT

KEYWORDS: Slow Feature Analysis, State Estimation, Classification, Stochasticity

Invariant features of temporally varying signals are useful for analysis and classification. Slow feature analysis (SFA) is a new method for learning invariant or slowly varying features from a vectorial input signal. It is guaranteed to find the optimal solution within a family of functions directly and can learn to extract a large number of decorrelated features, which are ordered by their degree of invariance. In most studies, the notion of “invariance” or “slowness” is defined as the temporal average of the squared time derivative of a signal. In this work, we rewrite the SFA objective function in the frequency domain and discuss certain drawbacks in the original formulation. To address them, we introduce a generalized version of SFA called the Generalized Slow Feature Analysis (GSFA), which introduces a tunable filter. We find that standard linear SFA becomes a special case of GSFA. Since GSFA assumes the signals are deterministic, it is not suitable to handle stochastic signals from dynamic processes. Hence, we propose a probabilistic version of GSFA that incorporates measurement and process noise into the framework. The efficacy of the algorithm is verified after the application to tasks like state estimation and classification.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

Slow Feature Analysis (SFA) was originally developed in the context of an abstract model of unsupervised learning of invariances in the visual system of primates and was subsequently also used for learning complex-cell receptive fields and place cells in the hippocampus (Wiskott and Sejnowski (2002)). SFA is based on the slowness principle that assumes the primary sensory signals, which in general code for local properties, vary quickly while the perceived environment changes slowly. If one succeeds in extracting slow features from the quickly varying sensory signal, one is likely to obtain an invariant representation of the environment. One of the advantages of SFA is that it is a closed-form algorithm and it is guaranteed to find the optimal solution within the considered function space (linear). The concise formulation of the SFA optimization problem also permits an extended mathematical treatment so that its properties are well understood analytically (Wiskott (2003*b*)).

1.1 Literature Review

There has been substantial work on this topic and interesting connections between SFA and Fischer Discriminant Analysis (FDA) (Klampfl and Maass (2010)), Laplacian Eigen Maps (LEM) (Sprekeler (2011)), and Independent Component Analysis (ICA) (Blaschke *et al.* (2006)) have been established. The SFA objective function has also been given a frequency-domain perspective (Doumanoglou *et al.* (2019)). Generating invariant representations using SFA have been proven to be helpful in several tasks like image classification (Escalante B. and Wiskott (2013)), object recognition (Franzius *et al.* (2008)), blind source separation (Wiskott (2003*a*)), age and gender estimation (Escalante B. and Wiskott (2012)), human action recognition (Zhang and Tao (2019)), process monitoring (Shang *et al.* (2016)), anomaly detection, fault classification (Huang *et al.* (2020)), behavior analysis (Zafeiriou *et al.* (2016)), etc. Moreover, SFA has also been used as a dimensionality reduction and feature extraction technique. Considering the deterministic nature of SFA, there have been attempts to come up with a probabilistic model for SFA (Probabilistic SFA) (Turner and Sahani (2007)). Since then, several extensions have been proposed, depending on the task in which probabilistic SFA is applied (Jiang *et al.* (2021)). Though SFA is a versatile algorithm for extracting temporally invariant signals, we strongly believe that temporal invariance is better described in the frequency domain (Doumanoglou *et al.* (2019)).

1.2 Mathematical Preliminaries - Slow Feature Analysis (SFA)

Let $\mathbf{y}[k] = [y_1[k], y_2[k], ..., y_M[k]]^T$, $k \in \{0, 1, ..., N-1\}$ denote a multi-dimensional discrete signal. The SFA's objective is to find a function
$\mathbf{f}(\mathbf{y}[k]) = [f_1(\mathbf{y}[k]), f_2(\mathbf{y}[k]), ..., f_P(\mathbf{y}[k])]^T$ generating the P-dimensional signal $\mathbf{x}[k] = [x_1[k], x_2[k], ..., x_P[k]]^T$ such that $x_j[k] = f_j(\mathbf{y}[k])$. The function $f_j(\mathbf{y}[k])$ should be able to minimize the following objective function ψ_{SFA_j} sequentially, for every $j \in \{1, ..., P\}$

$$\min_{f_j} \psi_{SFA_j} = \min_{f_j} \frac{1}{N-1} \sum_{k=1}^{N-1} (x_j[k] - x_j[k-1])^2 \tag{1.1}$$

under the constraints given below (for every $i, j \in \{1, ..., P\}$):

$$< x_j > = 0 \tag{1.2}$$

$$\mathbf{x}_i^T \mathbf{x}_j = N\delta_{ij} \tag{1.3}$$

Here, $< x > = \frac{1}{N} \sum_{k=0}^{N-1} x[k]$ is the time-average operator. Linear SFA assumes the function to be linear such that $\mathbf{x}[k] = \mathbf{f}(\mathbf{y}[k]) = \mathbf{W}^T \mathbf{y}[k]$, $\mathbf{W} \in \mathbb{R}^{M \times P}$. When such a constraint is imposed, the optimization reduces to solving an eigenvalue equation (Wiskott and Sejnowski (2002)).

Substituting $x_j[k] = f_j(\mathbf{y}[k]) = \mathbf{w}_j^T \mathbf{y}[k]$ ($\mathbf{w}_j$ is the j^{th} column of $\mathbf{W}$) in equation (1.1), we get

$$\min_{f_j} \psi_{SFA_j} = \min_{\mathbf{w}_j} \mathbf{w}_j^T \left(\frac{1}{N-1} \sum_{k=1}^{N-1} (\mathbf{y}[k] - \mathbf{y}[k-1])(\mathbf{y}[k] - \mathbf{y}[k-1])^T \right) \mathbf{w}_j \quad (1.4)$$

under the constraints given below (for every $i, j \in \{1, ..., P\}$):

$$< y_j > = 0 \quad (1.5)$$

$$\mathbf{w}_i^T \mathbf{w}_j = \delta_{ij} \quad (1.6)$$

The solution to the above optimization problem is the P eigenvectors with the lowest eigenvalues of the matrix $\left(\frac{1}{N-1} \sum_{k=1}^{N-1} (\mathbf{y}[k] - \mathbf{y}[k-1])(\mathbf{y}[k] - \mathbf{y}[k-1])^T\right)$. In other words, $\mathbf{w}_j$ is obtained such that,

$$\left(\frac{1}{N-1} \sum_{k=1}^{N-1} (\mathbf{y}[k] - \mathbf{y}[k-1])(\mathbf{y}[k] - \mathbf{y}[k-1])^T \right) \mathbf{w}_j = \lambda_j \mathbf{w}_j \quad (1.7)$$

Here, $\lambda_1 \leq \lambda_2 \leq ... \leq \lambda_P$.

1.3 Motivation for the proposed method - GSFA

The slowness objective in SFA is defined in the time domain. It is the temporal average of the squared time derivative of $x[k]$. Higher the magnitude of ψ_{SFA}, the faster the signal. The lower the magnitude of ψ_{SFA}, the slower the signal. This notion of "slowness" can be better described in the frequency domain. The optimization problem of SFA can be viewed as minimizing the variance of the signal $z_j[k]$, where

$$\begin{aligned} z_j[k] &= x_j[k] - x_j[k-1] && (1.8) \\ &= (1 - q^{-1}) x_j[k] && (1.9) \\ &= G_{SFA}(q^{-1}) x_j[k] && (1.10) \end{aligned}$$

Here, q^{-1} is the backward difference operator ($q^{-1} x_j[k] = x_j[k-1]$). Under the assumptions of infinitely long and absolutely convergent signals $z_j[k]$ and $x_j[k]$,

$$Z_j(\omega) \;=\; (1 - e^{-i\omega})X_j(\omega) \tag{1.11}$$

Here, $X_j(\omega)$ and $Z_j(\omega)$ are the Discrete Time Fourier Transforms (DTFT) of $x_j[k]$ and $z_j[k]$, respectively. By Parseval's theorem, the energy of the signal $z_j[k]$ is proportional to $\int_{-\pi}^{\pi} |Z_j(\omega)|^2 \, d\omega$. Thus, we can write,

$$\psi_{SFA_j} \;\propto\; \int_{-\pi}^{\pi} (1 - \cos\omega))|X_j(\omega)|^2 \, d\omega \tag{1.12}$$

We can see that $W(\omega) \;=\; 1 - \cos\omega$ is a high-pass filter. Intuitively, SFA gives more importance to the high-frequency contents of a signal, and it wants to minimize the high-frequency contents to achieve "slowness." We strongly feel that this objective can be put in a straight forward way, where the low-frequency contents of a signal are maximized. Also, instead of a predefined choice for $W(\omega)$, we can make a choice for $W(\omega)$, which might be more suitable than traditional SFA (where $W(\omega) \;=\; 1 - \cos\omega$), for certain tasks. By choosing a different function $W(\omega)$, we aim to shape the frequency spectrum of the signal $x_j[k]$. This generalization allows us to generate a large class of features. This is the motivation to come up with our proposed method, Generalized SFA (GSFA)

1.4 Contributions

Our contributions are as follows:

1) We propose Generalized Slow Feature Analysis (GSFA), where we come up with a generalized and straightforward objective function in the frequency domain and obtain an analytical solution to the optimization problem.

2) Since GSFA does not take into account the stochasticity of the signals, it is not suitable for tasks like state estimation and fault classification. Hence, we propose probabilistic GSFA, which is the probabilistic version of GSFA.

To illustrate the advantages of our proposed methods, we apply them to tasks like state estimation and classification (GTSRB and fault classification in the TE process). Our results demonstrate that the generalization of the SFA objective function in the frequency domain enhances the performance in the above applications. Especially in applications like state estimation, probabilistic GSFA achieves the best results because it incorporates stochastic signals.

Section (2) explains the optimization problem of GSFA and derives the analytical solution. Section (3) elaborates on the probabilistic GSFA and derives an algorithm to estimate the parameters of the model. In section (4), GSFA and probabilistic GSFA is evaluated by applying them to tasks like state estimation and classification.

CHAPTER 2

GENERALIZED SLOW FEATURE ANALYSIS

2.1 Problem Statement

Let $\mathbf{y}[k] = [y_1[k], y_2[k], ..., y_M[k]]^T$, $k \in \{0, 1, ..., N-1\}$ denote a multi-dimensional discrete signal. The generalized SFA's objective is to find a function
$\mathbf{f}(\mathbf{y}[k]) = [f_1(\mathbf{y}[k]), f_2(\mathbf{y}[k]), ..., f_P(\mathbf{y}[k])]^T$ generating the P-dimensional signal $\mathbf{x}[k] = [x_1[k], x_2[k], ..., x_P[k]]^T$ such that $x_j[k] = f_j(\mathbf{y}[k])$. The function $f_j(\mathbf{y}[k])$ should be able to maximize the following objective function ψ_j sequentially, for every $j \in \{1, ..., P\}$

$$\max_{f_j} \psi_j = \max_{f_j} \frac{1}{N} \left\|\mathbf{G}\mathbf{x}_j\right\|^2 \tag{2.1}$$

under the constraints given below (for every $i, j \in \{1, ..., P\}$):

$$< x_j > = 0 \tag{2.2}$$

$$\mathbf{x}_i^T \mathbf{x}_j = N\delta_{ij} \tag{2.3}$$

Here, $\mathbf{G} \in \mathbb{R}^{N \times N}$ is a predefined filter, and $\mathbf{x}_j = [x_j[0], x_j[1], ..., x_j[N-1]]^T$ for every $j \in \{1, ..., P\}$.

2.2 Linear GSFA

This learning problem in section (2.1) is an optimization problem of variational calculus and in general, is difficult to solve. However, for the case that the input-output function components $\mathbf{f}(\mathbf{y}[k])$ are constrained to be linear ($\mathbf{x}[k] = \mathbf{f}(\mathbf{y}[k]) = \mathbf{W}^T\mathbf{y}[k]$, $\mathbf{W} \in \mathbb{R}^{M \times P}$), the problem simplifies significantly. An algorithm for solving the optimization problem under this constraint is given in this section.

We define $\mathbf{y}_j = [y_j[0], y_j[1], ..., y_j[N-1]]^T$ for every $j \in \{1, ..., M\}$ and the observation matrix $\mathbf{Y} = [\mathbf{y}_1, ..., \mathbf{y}_M]$, where $\mathbf{Y} \in \mathbb{R}^{N\times M}$. Since $\mathbf{x}[k] = \mathbf{f}(\mathbf{y}[k]) = \mathbf{W}^T\mathbf{y}[k]$, we can write

$$\mathbf{x}_j = \mathbf{Y}\mathbf{w}_j \tag{2.4}$$

where $\mathbf{w}_j$ is the j^{th} column of $\mathbf{W}$. We define an auxiliary variable $\mathbf{z}_j$ such that

$$\begin{aligned} \mathbf{z}_j &= \mathbf{G}\mathbf{x}_j & (2.5)\\ &= (\mathbf{G}\mathbf{Y})\mathbf{w}_j & (2.6)\\ &= \mathbf{Y}_f\mathbf{w}_j & (2.7) \end{aligned}$$

The filter $\mathbf{G}$ acts on the observation matrix to get the filtered observation matrix $\mathbf{Y}_f$. The objective function in equation (2.1) can be written in terms of $\mathbf{z}_j$.

$$\begin{aligned} \psi_j &= \frac{1}{N}\mathbf{z}_j^T\mathbf{z}_j & (2.8)\\ &= \mathbf{w}_j^T(\frac{1}{N}\mathbf{Y}_f^T\mathbf{Y}_f)\mathbf{w}_j & (2.9) \end{aligned}$$

Assuming $\frac{1}{N}\mathbf{Y}^T\mathbf{Y} = \mathbf{I}$ and $< y_j > = 0$ for all $j \in \{1, ..., P\}$, the constraints (2.2) and (2.3) are satisfied if and only if,

$$\mathbf{w}_i^T\mathbf{w}_j = \delta_{ij} \tag{2.10}$$

for all $i, j \in \{1, ..., P\}$. Thus, the overall optimization problem is,

$$\max_{\mathbf{w}_j} \mathbf{w}_j^T(\frac{1}{N}\mathbf{Y}_f^T\mathbf{Y}_f)\mathbf{w}_j \tag{2.11}$$

under the constraint (2.10). The solution to this optimization problem is obtained by solving the eigenvalue problem,

$$(\frac{1}{N}\mathbf{Y}_f^T\mathbf{Y}_f)\mathbf{w} = \lambda\mathbf{w} \tag{2.12}$$

The eigenvectors corresponding to the P largest eigenvalues form the solution $\{\mathbf{w}_{j=1,...,P}\}$. From this formulation, we can see that $P \leq M$ i.e., the number of latent variables is constrained by the number of observed variables.

2.3 Special Remarks on the Filter, G

The filter **G** has the following structure,

$$\mathbf{G} = \begin{pmatrix} g[0] & g[N-1] & ... & g[1] \\ g[1] & g[0] & ... & g[2] \\ . & . & . & . \\ . & . & . & . \\ . & . & . & . \\ g[N-1] & g[N-2] & ... & g[0] \end{pmatrix} \tag{2.13}$$

Here, $g[k]$ is the impulse response coefficients of the transfer function $G(q^{-1})$, specified by the user. Given the way the matrix has been constructed, multiplying **G** with an arbitrary vector $\mathbf{x} = [x[0], x[1], ..., x[N-1]]^T$ is equivalent to performing circular convolution between the sequences $g[k]$ and $x[k]$. Mathematically, for every $k \in \{0, 1, ..., N-1\}$,

$$\mathbf{Gx}_k = \sum_{l=0}^{N-1} g[l]x[(k-l)\, mod\, N] \tag{2.14}$$

$\mathbf{Gx}_k$ refers to the k^{th} element of the vector **Gx**. Moreover, the Discrete Fourier Transform (DFT) of a circular convolution between two signals is given by the element-wise product of the DFT coefficients of the individual signals. Using these properties, it can be shown using Parseval's theorem that the objective function ψ_j in equation (2.1) follows,

$$\psi_j \propto \sum_{n=0}^{N-1} |G[n]X_j[n]|^2 \tag{2.15}$$

where $G[n]$ and $X_j[n]$ are the DFT of $g[k]$ and $x_j[k]$, respectively. If the matrix **G** is arbitrary, we cannot write ψ_j in the above form. By choosing the filter **G**, we are implicitly shaping the power spectrum of the extracted features. Thus, if we want slow features (low-frequency signals), we can construct **G** from a low-pass filter $G(q^{-1})$ and maximize the low-frequency contents of the extracted signals. This approach seems straightforward compared to that of SFA where the high-frequency contents of the extracted features are minimized to achieve "slowness". The main advantage of this approach is that we have the choice of the matrix **G**, which can be tuned accordingly for each task. Particularly, when $G(q^{-1}) = 1+q^{-1}$, standard linear SFA becomes a special case of linear GSFA (refer Appendix). Thus, GSFA subsumes a large class of features which can be obtained by simply changing the filter matrix, **G**.

CHAPTER 3

PROBABILISTIC GSFA

In the previous sections, the signals were assumed to be deterministic. However, this assumption may not be valid in all applications, especially in areas like process monitoring. We need to take into account the stochastic nature of the signals. In this section, we come up with a probabilistic model for GSFA that includes measurement and process noise in the formulation while retaining flexibility in choosing the filter matrix.

3.1 Prior: Latent Signals

Let us assume that all signals are zero at negative time instants.
For $j \in \{1, ..., P\}$, the j^{th} latent signal $x_j[k]$ follows,

$$\begin{aligned} x_j[k] &= -\Lambda_j(q^{-1})x_j[k] + e_j[k] && (3.1) \\ &= -\sum_{l=0}^{k} \lambda_j[l]x_j[k-l] + e_j[k] && (3.2) \end{aligned}$$

Here, $e_j[k] \sim GWN(0, \sigma_j^2)$ and $\lambda_j[k]$ are the impulse response coefficients of a causal filter, $\Lambda_j(q^{-1})$. Note that $\Lambda_j(q^{-1})$ should have a delay of at least one (ie. $\lambda_j[0] = 0$). We can also write,

$$\begin{aligned} x_j[k] &= H_j(q^{-1})e_j[k] && (3.3) \\ H_j(q^{-1}) &= \frac{1}{1 + \Lambda_j(q^{-1})} && (3.4) \end{aligned}$$

The marginal and conditional distributions are given by,

$$f(x_j[k]) = \frac{1}{\sqrt{2\pi\sigma_j^2 \sum_{l=0}^{k} h_j^2[l]}} \exp\left\{-\frac{1}{2\sigma_j^2 \sum_{l=0}^{k} h_j^2[l]} x_j^2[k]\right\} \tag{3.5}$$

$$f(x_j[k]|k-1) = \frac{1}{\sqrt{2\pi\sigma_j^2}} \exp\left\{-\frac{1}{2\sigma_j^2}\left(x_j[k] + \sum_{l=0}^{k} \lambda_j[l] x_j[k-l]\right)^2\right\} \tag{3.6}$$

Here, $h_j[k]$ are the impulse response coefficients of $H_j(q^{-1})$.
Note that $h_j[0] = 1$. We would require absolute convergence of $h_j[k]$ for $x_j[k]$ to be second-order stationary at large k. At large k, for $Var(x_j[k]) = 1$, we have the following constraint, for $j \in \{1, ..., P\}$

$$\sigma_j^2 \sum_{l=0}^{\infty} h_j^2[l] = 1 \tag{3.7}$$

The observation signal $\mathbf{y}[k] \in \mathbb{R}^{M\times 1}$ follows,

$$\mathbf{y}[k] = f_\phi(\mathbf{x}[k]) + \boldsymbol{\varepsilon}[k] \tag{3.8}$$

Here, $\boldsymbol{\varepsilon}[k] \sim GWN(\mathbf{0}, \sigma_\varepsilon^2 \mathbf{I}_{M\times M})$ and $f_\phi(.)$ is a function parameterized by ϕ. When $f_\phi(\mathbf{x}[k]) = \mathbf{W}\mathbf{x}[k]$, $\mathbf{W} \in \mathbb{R}^{M\times P}$, we get a linear generative model that does not use past latent variables to generate the observation signal (linear GSFA). The conditional distribution is given by,

$$f(\mathbf{y}[k]|k-1) = \frac{1}{\sqrt{2\pi\sigma_\varepsilon^2}} \exp\left\{-\frac{1}{2\sigma_\varepsilon^2} \|\mathbf{y}[k] - f_\phi(\mathbf{x}[k])\|^2\right\} \tag{3.9}$$

3.2 Estimation of Parameters

Let $\mathbf{Y} \in \mathbb{R}^{N\times M}$ be the observation matrix. Given $\mathbf{Y}$, the task is to estimate the parameters $\mathbf{\Phi} = \left\{\sigma_{j=1:P}^2,\ (\lambda_{j=1:P}[k]|k>0),\ \sigma_\varepsilon^2,\ \phi\right\}$. We follow the variational EM algorithm for estimating the parameters. Let us define a variational distribution $q(\mathbf{x}[k]|\boldsymbol{\theta}) \sim N(\boldsymbol{\mu}_{\theta_1}(\mathbf{y}[k]), \mathbf{\Sigma}_{\theta_2})$, where $\boldsymbol{\mu}_{\theta_1}(.)$ and $\mathbf{\Sigma}_{\theta_2}$ are functions parameterized by $\boldsymbol{\theta} = (\theta_1, \theta_2)$. For the latent signal matrix $\mathbf{X} \in \mathbb{R}^{N\times P}$, we assume the following factorization,

$$q(\mathbf{X}|\boldsymbol{\theta}) \;=\; \prod_{k=0}^{N-1} q(\mathbf{x}[k]|\boldsymbol{\theta}) \tag{3.10}$$

Let us define $L(q(\mathbf{X}|\boldsymbol{\theta}), \boldsymbol{\Phi})$,

$$L(q(\mathbf{X}|\boldsymbol{\theta}), \boldsymbol{\Phi}) \;=\; \mathbb{E}_q(\log(f(\mathbf{Y}, \mathbf{X})|\boldsymbol{\Phi})) \;-\; \mathbb{E}_q(q(\mathbf{X}|\boldsymbol{\theta})) \tag{3.11}$$

The variational EM algorithm (Bishop (2007)) is a two-stage iterative optimization technique. Suppose that the current value of the parameter vector is $\boldsymbol{\Phi}_{old}$ and $\boldsymbol{\theta}_{old}$. In the E step, $L(q(\mathbf{X}|\boldsymbol{\theta}), \boldsymbol{\Phi}_{old})$ is maximized with respect to $\boldsymbol{\theta}$ while holding $\boldsymbol{\Phi}_{old}$ fixed, to give $\boldsymbol{\theta}_{new}$. In the subsequent M step, the distribution $q(\mathbf{X}|\boldsymbol{\theta}_{new})$ is held fixed and $L(q(\mathbf{X}|\boldsymbol{\theta}_{new}), \boldsymbol{\Phi})$ is maximized with respect to $\boldsymbol{\Phi}$ to give $\boldsymbol{\Phi}_{new}$. These steps are repeated until convergence (or a fixed number of iterations). After the completion of the algorithm, the distribution $q(\mathbf{X}|\boldsymbol{\theta}_{final})$ is an approximation to the true posterior distribution $f(\mathbf{X}|\mathbf{Y}, \boldsymbol{\Phi}_0)$, where $\boldsymbol{\Phi}_0$ is the true parameter values of the data generating process.

Equation (3.11) can be split as,

$$\begin{aligned} L(q(\mathbf{X}|\boldsymbol{\theta}), \boldsymbol{\Phi}) \;=\; & \mathbb{E}_q[\log f(\mathbf{x}[0])] \;+\; \sum_{k=1}^{N-1} \mathbb{E}_q[\log(f(\mathbf{x}[k]|k-1))] \\ & +\; \sum_{k=0}^{N-1} \mathbb{E}_q[\log(f(\mathbf{y}[k]|\mathbf{x}[k]))] - N\,\mathbb{E}_q[\log q(\mathbf{x}[k]|\boldsymbol{\theta})] \end{aligned} \tag{3.12}$$

The individual terms in equation (3.12) are,

$$\mathbb{E}_q[\log f(\mathbf{x}[0])] \;=\; -\frac{1}{2}\log[(2\pi)^P|\boldsymbol{\Sigma}_e|] - \frac{1}{2}\mathrm{tr}[\boldsymbol{\Sigma}_e^{-1}\,\mathbb{E}_q(\mathbf{x}[0]\mathbf{x}[0]^T)] \tag{3.13}$$

$$\mathbb{E}_q[\log(f(\mathbf{x}[k]|k-1))] \;=\; -\frac{1}{2}\log[(2\pi)^P|\boldsymbol{\Sigma}_e|] - \frac{1}{2}\mathrm{tr}[\boldsymbol{\Sigma}_e^{-1}\,\mathbb{E}_q(\mathbf{z}[k]\mathbf{z}[k]^T)] \tag{3.14}$$

$$\mathbb{E}_q[\log(f(\mathbf{y}[k]|\mathbf{x}[k]))] \;=\; -\frac{M}{2}\log(2\pi\sigma_\varepsilon^2) - \frac{1}{2\sigma_\varepsilon^2}\,\mathbb{E}_q(\|\mathbf{y}[k] - f_\phi(\mathbf{x}[k])\|^2) \tag{3.15}$$

$$\mathbb{E}_q[\log q(\mathbf{x}[k]|\boldsymbol{\theta})] \;=\; -\frac{1}{2}\log|\boldsymbol{\Sigma}_{\theta_2}| \tag{3.16}$$

$\odot$ - Element-wise multiplication
tr - Trace of a matrix
diag(.) - Turns a vector into a diagonal matrix

Here, $\boldsymbol{\Sigma}_e = \text{diag}(\sigma_1^2, ..., \sigma_P^2)$

$$\mathbb{E}_q(\mathbf{x}[k]\mathbf{x}[k]^T) = \boldsymbol{\Sigma}_{\theta_2} + \boldsymbol{\mu}_{\theta_1}(\mathbf{y}[k])^T \boldsymbol{\mu}_{\theta_1}(\mathbf{y}[k]) \tag{3.17}$$

$$\mathbb{E}_q(\mathbf{z}[k]\mathbf{z}[k]^T) = \boldsymbol{\Sigma}_{\theta_2} \odot \mathbf{S} + \mathbb{E}_q(\mathbf{z}[k])^T \, \mathbb{E}_q(\mathbf{z}[k]) \tag{3.18}$$

We have,

$$\mathbb{E}_q(\mathbf{z}[k]) = \mathbb{E}_q(\mathbf{x}[k]) + [\Lambda_1(q^{-1}) \, \mathbb{E}_q(x_1[k]), ..., \Lambda_P(q^{-1}) \, \mathbb{E}_q(x_P[k])]^T \tag{3.19}$$

$$\mathbf{S} = \mathbf{1}_{P \times P} + \begin{pmatrix} \sum_{l=1}^{\infty} \lambda_1^2[l] & \sum_{l=1}^{\infty} \lambda_1[l]\lambda_2[l] & ... & \sum_{l=1}^{\infty} \lambda_1[l]\lambda_P[l] \\ \sum_{l=1}^{\infty} \lambda_2[l]\lambda_1[l] & \sum_{l=1}^{\infty} \lambda_2^2[l] & ... & \sum_{l=1}^{\infty} \lambda_2[l]\lambda_P[l] \\ . & . & . & . \\ . & . & . & . \\ . & . & . & . \\ \sum_{l=1}^{\infty} \lambda_P[l]\lambda_1[l] & \sum_{l=1}^{\infty} \lambda_P[l]\lambda_2[l] & ... & \sum_{l=1}^{\infty} \lambda_P^2[l] \end{pmatrix} \tag{3.20}$$

The expectation in equation (3.15) can be approximated using Monte-Carlo simulations because a closed-form expression is not possible for a general $f_\phi(.)$. However, when $f_\phi(.)$ is linear (linear probabilistic GSFA), the computation greatly reduces. The above expressions assume that the process has been generating data for a long period of time so that the latent signals form a stationary process. The overall algorithm is shown in the algorithm (1). Since this is an iterative algorithm, it is sensitive to initialization. We need to run the algorithm for several initializations and choose the one that produces the maximum value of the objective function in equation (3.11).

Since there are an infinite number of impulse response coefficients $\lambda_{j=1:P}[k]$ to estimate,

Algorithm 1 Estimating Parameters in Probabilistic GSFA

```
Require: $\mathbf{Y} \in \mathbb{R}^{N\times M}$, $iter$ (Total no. of iterations)
  $i \leftarrow 1$
  $\boldsymbol{\theta}^{(1)} \leftarrow \boldsymbol{\theta}_{init}$
  $\boldsymbol{\Phi}^{(1)} \leftarrow \boldsymbol{\Phi}_{init}$
  while $i \leq iter$ do
    $\boldsymbol{\theta}^{(i+1)} \leftarrow \max_{\boldsymbol{\theta}} L(q(\mathbf{X}|\boldsymbol{\theta}), \boldsymbol{\Phi}^{(i)})$
    $\boldsymbol{\Phi}^{(i+1)} \leftarrow \max_{\boldsymbol{\Phi}} L(q(\mathbf{X}|\boldsymbol{\theta}^{(i+1)}), \boldsymbol{\Phi})$
    $i \leftarrow i+1$
  end while
  return $\boldsymbol{\theta}^{(iter)}$ and $\boldsymbol{\Phi}^{(iter)}$
Ensure: $\boldsymbol{\Phi}^{(iter)}$ satisfies second-order stationarity and constraint (3.7)
```

we parameterize $\Lambda_j(q^{-1})$. An effective parameterization would be,

$$\Lambda_j(q^{-1}) \;=\; \frac{b_j q^{-1}}{1 - d_j q^{-1}} \tag{3.21}$$

Note that when $d_j = 0$ for all j, we obtain the probabilistic model for standard SFA (refer Appendix). We should keep in mind that the estimated values for $\{b_{j=1:P}, d_{j=1:P}\}$ should result in a stable and invertible $H_{j=1:P}(q^{-1})$ (also satisfying constraint (3.7)). Hence, we should be solving a constrained (and non-linear) optimization problem. Since the distribution $q(\mathbf{X}|\boldsymbol{\theta}_{final})$ is an approximation to the true posterior distribution, we can use it for inference (estimation and construction of prediction intervals for latent signals).

The choice of parameterization of $H_j(q^{-1})$ shapes the power spectrum of the latent signals. For the above parameterization (3.21), the power spectral density of $x_j[k]$ is given by,

$$\gamma_j(\omega) \;=\; \frac{\sigma_j^2}{2\pi}\left[\frac{1 + d_j^2 - 2d_j\cos\omega}{1 + (d_j - b_j)^2 - 2(d_j - b_j)\cos\omega}\right] \tag{3.22}$$

Clearly, when d_j is very much smaller than b_j, $x_j[k]$ has predominantly low-frequency content and is a slow signal. An advantage of probabilistic GSFA, in addition to handling stochastic noise, is that we have the choice of parameterization depending on the task. Also, since we have assumed $f_{\Phi}(.)$ to be a general function, probabilistic GSFA can handle non-linear processes without relying on any non-linear expansion (which is the case with SFA and GSFA).

CHAPTER 4

APPLICATIONS/CASE STUDIES

In this section, we demonstrate the advantages of GSFA (and probabilistic GSFA) over standard SFA through three case studies: a) State estimation b) The German Traffic Sign Recognition Benchmark (GTSRB), and c) Fault classification in the Tennessee Eastman (TE) process. Through the case studies, the main advantages of GSFA (and probabilistic GSFA) that we wish to demonstrate are a) the choice of the filter matrix, b) the ease of handling non-linearities using probabilistic GSFA without relying on non-linear expansions, and c) the modeling of process and measurement noise (in probabilistic GSFA)

4.1 State Estimation

We generate observations $\mathbf{y}[k]$ through the following data-generating process (DGP),

$$\mathbf{y}[k] = \begin{bmatrix} 1.539 & -0.742 & 0.783 \\ 0.467 & -0.664 & -0.861 \\ 0.786 & -0.894 & -0.428 \end{bmatrix} \mathbf{x}[k] + \boldsymbol{\varepsilon}[k] \tag{4.1}$$

$$x_1[k] = \frac{1-0.5q^{-1}}{1+0.2q^{-1}} e_1[k] \tag{4.2}$$

$$x_2[k] = \frac{1-0.8q^{-1}}{1-0.6q^{-1}} e_2[k] \tag{4.3}$$

$$x_3[k] = \frac{1-0.1q^{-1}}{1+0.4q^{-1}} e_3[k] \tag{4.4}$$

Here, $\mathbf{e}[k] = [e_1[k], e_2[k], e_3[k]]^T \sim GWN(\mathbf{0}, \text{diag}(0.6621, 0.9412, 0.7706))$ are the endogenous driving forces and $\boldsymbol{\varepsilon}[k] \sim GWN(\mathbf{0}, 0.2802\mathbf{I}_{3\times3})$ is the measurement noise.

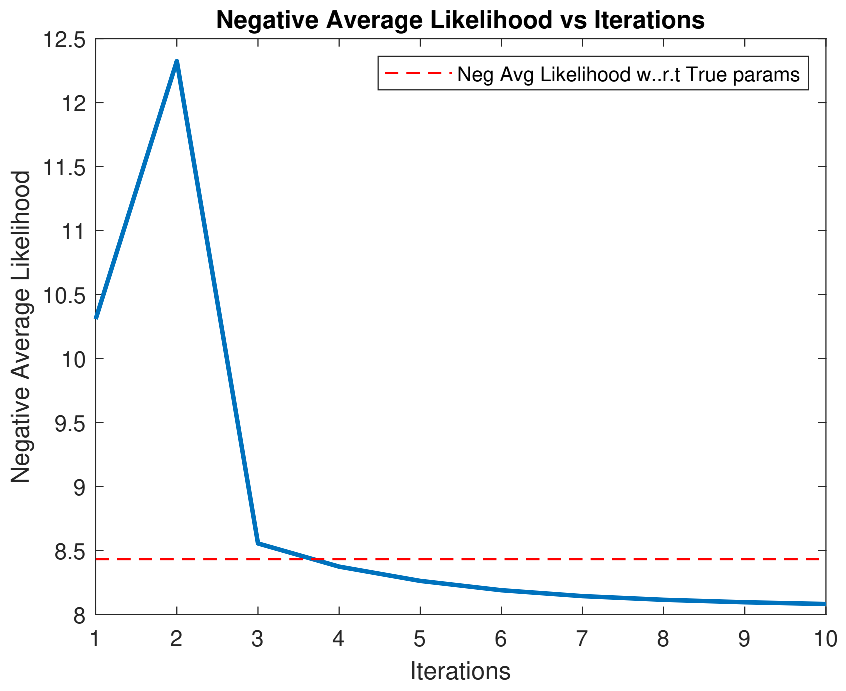

Figure 4.1: Training curve - Red dotted line is the negative average log-likelihood with respect to true parameters ($-\log f(\mathbf{Y}_{train}|\mathbf{\Phi}_0)$)

With this DGP, we generate $N = 200$ observation data points $\mathbf{y}[k]$ for training and testing. The minimum signal-to-noise ratio ($\frac{\min(\text{diag}(\text{Var}(\mathbf{Wx}[k])))}{\sigma_\varepsilon^2}$) is maintained at 5. The task is to estimate $\mathbf{x}[k]$ from the observations $\mathbf{y}[k]$. With the parameterization in (3.21), we run the algorithm (1) (linear probabilistic GSFA) on the training observation data points to estimate the parameters of our model. We run several initializations and choose the parameter $\mathbf{\Phi}_{est}$ that produces the lowest negative log-likelihood ($-\log f(\mathbf{Y}_{train}|\mathbf{\Phi})$) on the training set. The training curve is shown in figure (4.1). The red line, which indicates the negative average log-likelihood with respect to the true parameters, is used as a baseline to compare our estimated model. Eventually, we achieve a lower negative average log-likelihood (with respect to the estimated parameters) than the baseline. After training, we estimate the latent signals from the test data $\mathbf{Y}_{test}$, using the distribution $q(\mathbf{x}[k]|\boldsymbol{\theta}_{est})$.

	Avg. corr. coef. b/w true and estimated latents		
Method	Latent signal 1	Latent signal 2	Latent signal 3
Probabilistic GSFA	0.7627 ($\pm$0.03)	0.6447 ($\pm$0.04)	0.895 ($\pm$0.02)

Table 4.1: This table shows the average correlation coefficient between each of the true and estimated latent signals obtained from 100 realizations of the test data. The standard deviation is shown in brackets

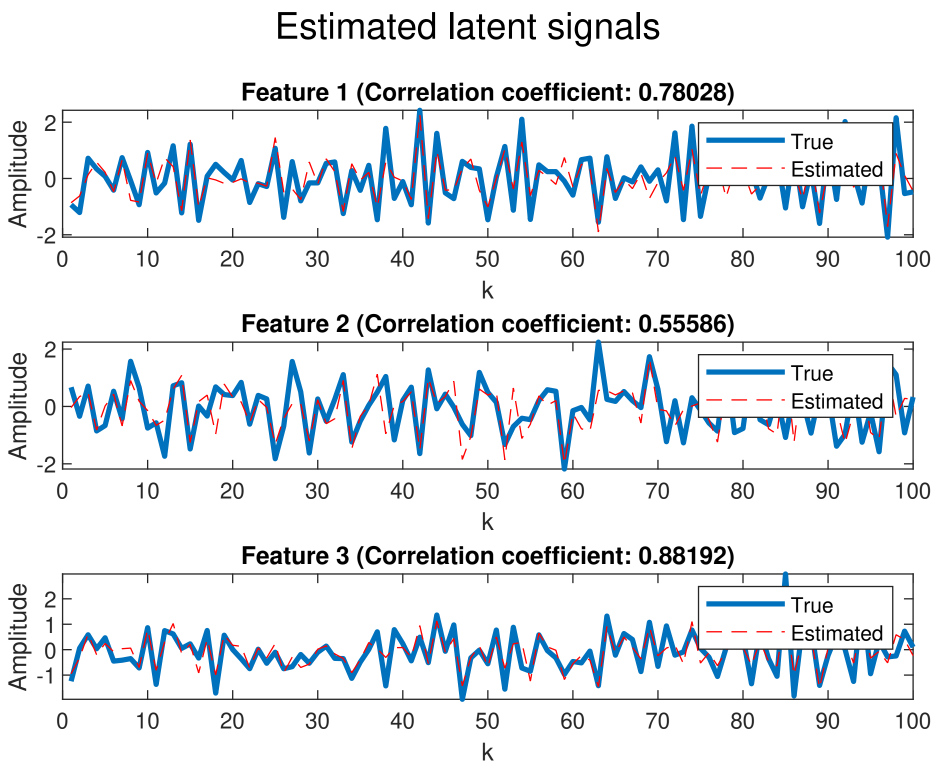

Figure 4.2: Correlation between true and estimated latent signals

From table (4.1), the estimated latent signals have a good correlation with the true latent signals across several realizations of the test data. From figure (4.2), which shows the true and estimated latent signals, we can qualitatively conclude that the estimated latent signals closely match the true latent signals. However, in practice, we do not have knowledge of the true latent signals. A practical evaluation metric would be to quantify the reconstruction error of $\mathbf{Y}_{test}$. We perform experiments by reconstructing $\mathbf{Y}_{test}$ from the estimated latent signals $\hat{\mathbf{X}}_{test}$ (using the estimated synthesis model - $f_{\phi_{est}}(.)$). The results are shown in figure (4.3) (top). The correlation coefficient is close to one for all the observation signals. Another way to assess the goodness of the model is to check the whiteness of the residuals ($= \mathbf{y}[k] - \mathbf{y}_{rec}[k]$) by examining the auto-correlation function (ACF). From the sample ACF of the residuals (figure (4.3) (bottom)), we can see that the residuals are white. This proves the goodness of our estimates.

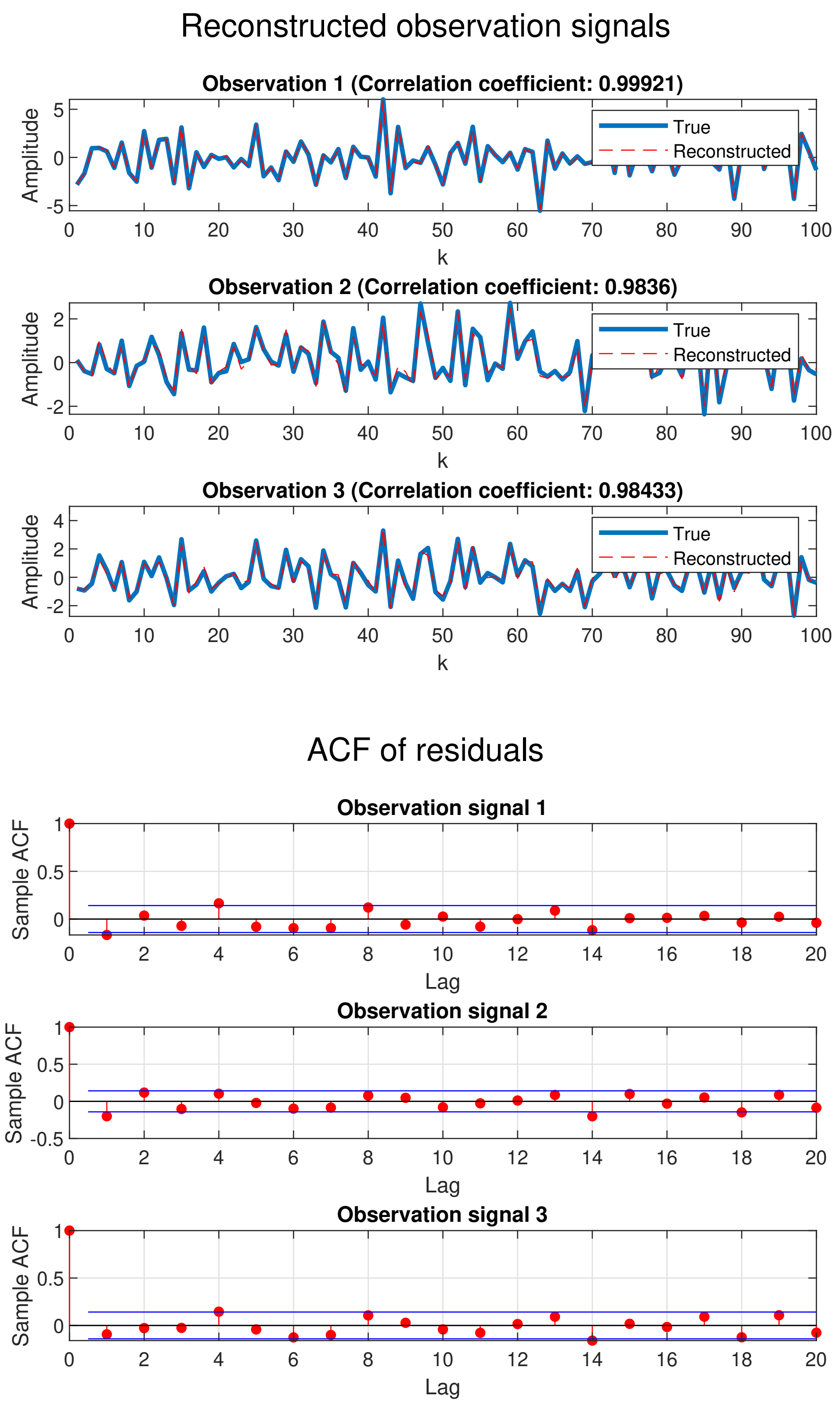

Figure 4.3: (Top) Correlation between true and reconstructed observation signals (correlation coefficient in brackets). (Bottom) The ACF plot of the residuals (= True - Reconstructed observation signals)

We plot the power spectral density (PSD) of the true and estimated latent signals in figure (4.4). We can see that the PSD of the estimated latent signals has a similar shape to that of the true latent signals (with predominantly high-frequency content). This shows that our formulation of probabilistic GSFA is not only useful for extracting slow signals (with predominantly low-frequency content) but also gives the freedom for extracting signals with a desired frequency content.

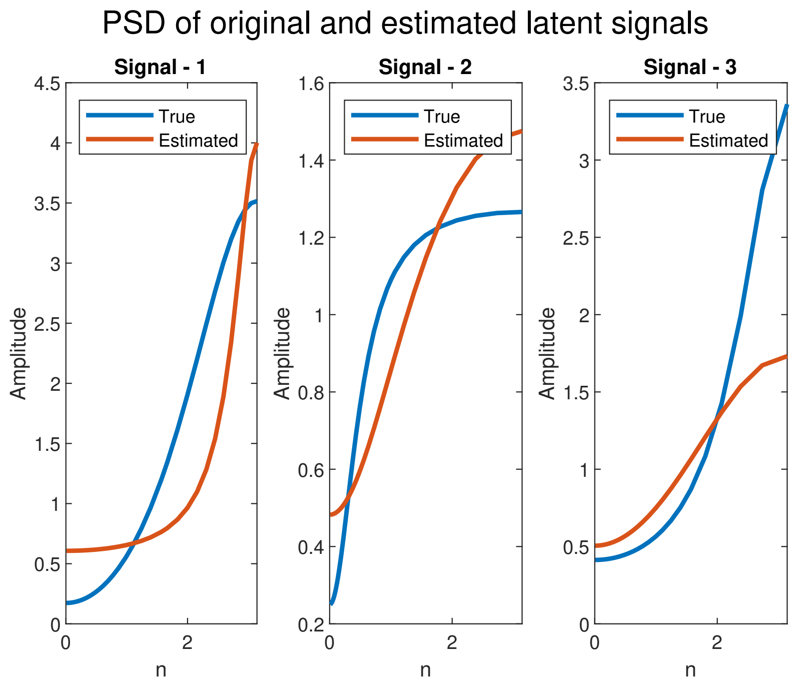

Figure 4.4: Power spectral density (PSD) of the true and estimated latent signals

Comparison between SFA and probabilistic GSFA

Standard SFA is the limiting case of probabilistic SFA. In probabilistic SFA, the latent signals evolve as an AR(1) process (Turner and Sahani (2007)). Hence, SFA (or probabilistic SFA) is not suited for a DGP whose latent signals evolve through a different process (MA/ARMA/higher order AR). To make a fair comparison between SFA and probabilistic GSFA, we repeat the above experiments for the following DGP (as mentioned in (Turner and Sahani (2007))), where we generate $N = 200$ observation data points $\mathbf{y}[k]$ for training and testing. The minimum signal-to-noise ratio is maintained at 5.

$$\mathbf{y}[k] = \begin{bmatrix} -0.216 & -0.910 & 0.353 \\ 0.002 & -0.362 & -0.932 \\ 0.976 & -0.201 & 0.080 \end{bmatrix} \mathbf{x}[k] + \varepsilon[k] \tag{4.5}$$

$$x_1[k] = \frac{1}{1-0.1q^{-1}} e_1[k] \tag{4.6}$$

$$x_2[k] = \frac{1}{1-0.2q^{-1}} e_2[k] \tag{4.7}$$

$$x_3[k] = \frac{1}{1-0.3q^{-1}} e_3[k] \tag{4.8}$$

Here, $\mathbf{e}[k] = [e_1[k], e_2[k], e_3[k]]^T \sim GWN(\mathbf{0}, \text{diag}(0.99, 0.96, 0.91))$ are the endogenous driving forces and $\varepsilon[k] \sim GWN(\mathbf{0}, 0.20\mathbf{I}_{3\times 3})$ is the measurement noise.

Figure (4.5) shows the true and estimated latent signals from standard linear SFA and linear probabilistic GSFA (with the parameterization in (3.21)) on the test data. By visual comparison, the latent signals estimated using linear probabilistic GSFA closely match the true latent signals compared to that of standard SFA, except for the latent signal 3. In the presence of measurement noise, $\varepsilon[k]$, probabilistic GSFA outperforms SFA, which is evident by comparing the correlation coefficients between the true and estimated latent signals (refer table (4.2)). This is because standard SFA assumes that all signals are deterministic, unlike probabilistic GSFA.

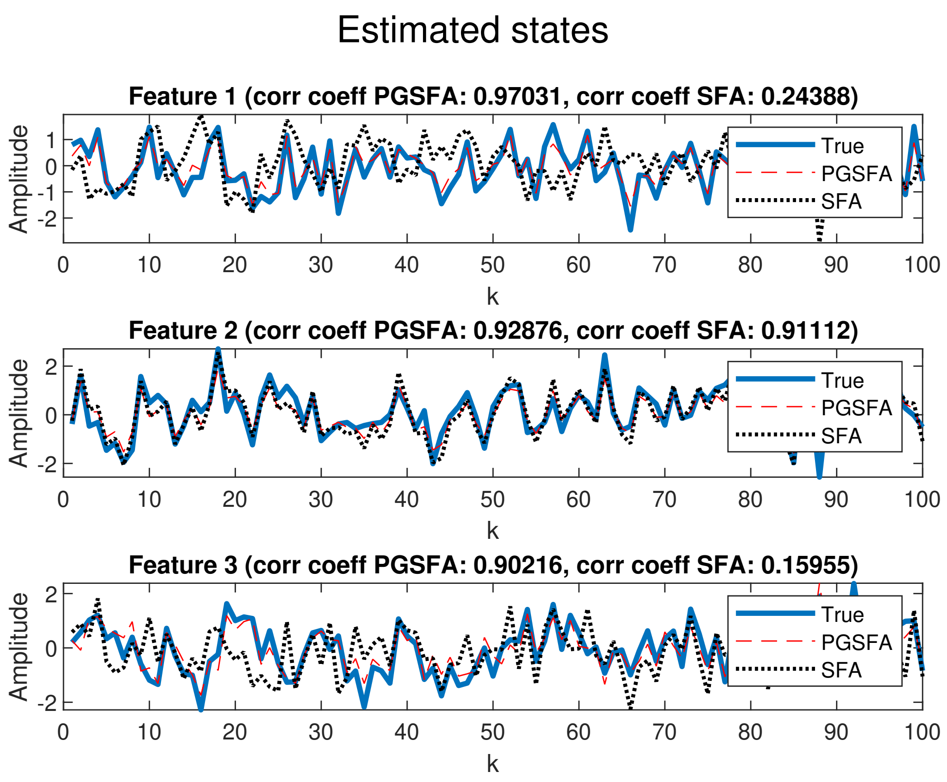

Figure 4.5: Correlation between true and estimated latent signals - linear SFA and probabilistic GSFA

	Avg. corr. coef. b/w true and estimated latents		
Method	Latent signal 1	Latent signal 2	Latent signal 3
Probabilistic GSFA	0.9715 ($\pm$0.004)	0.9214 ($\pm$0.01)	0.9057 ($\pm$0.01)
Standard SFA	0.2113 ($\pm$0.07)	0.9006 ($\pm$0.01)	0.1382 ($\pm$0.07)

Table 4.2: Comparison b/w SFA and probabilistic GSFA on state estimation

4.2 German Traffic Sign Recognition Benchmark (GTSRB)

The German Traffic Sign Recognition Benchmark (GTSRB) (Houben *et al.* (2013)) consists of photographs of 43 different traffic signs taken on German roads under uncontrolled conditions with variations in lighting, sign size, and distance. It has 39,209 labeled images for training and 12,630 images without labels for evaluation. In this section, we compare standard SFA with our proposed methods GSFA and probabilistic GSFA on the GTSRB and highlight the advantages of GSFA over standard SFA. For a fair comparison, all the stages of the pipeline, except feature extraction, are kept the same so that the difference in the final evaluation metrics can be attributed only to the feature extraction algorithm.

Preprocessing

The images were resized to 64×128 dimensions and the HOG (Histogram of Oriented Gradients) features were computed (which resulted in a vector of size 3780). We used PCA to further reduce the dimensionality to 200 (as it explained $> 75\%$ of the total variance of the training data). To employ methods like SFA, GSFA, and probabilistic GSFA, we sort the training data (and not the test data) according to their class labels.

Feature extraction method 1: SFA

We use a two-layer, cascaded, SFA with non-linear expansion (a compact expansion that only doubles the data dimension was employed, $\mathbf{x}^T \rightarrow [\mathbf{x}^T, |\mathbf{x}^T|^{0.8}]$ where the absolute value and exponent 0.8 are computed component-wise). At each layer, we retain only the top 400 slow features. Once the slow features are computed, the classification head is trained on it.

Feature extraction method 2: GSFA

We use a two-layer, cascaded, GSFA with non-linear expansion (a compact expansion that only doubles the data dimension was employed, $\mathbf{x}^T \rightarrow [\mathbf{x}^T, |\mathbf{x}^T|^{0.8}]$ where the absolute value and exponent 0.8 are computed component-wise). At each layer, we retain only the top 400 slow features. Once the slow features are computed, the classification head is trained on it. To show the flexibility in shaping the frequency content of the extracted features according to the task, we try out filter matrices constructed from the following transfer function operators $G(q^{-1})$:

1. $G(q^{-1}) \ = \ 1 + q^{-1} + q^{-2}$
2. $G(q^{-1}) \ = \ 1 + \sum_{i=1}^{79} q^{-i}$
3. $G(q^{-1}) \ = \ \sum_{i=1}^{5} c_i F_i(q^{-1}, p, d)$

Here,

$$F_i(q^{-1}, p, d) \;=\; q^{-(1+d)} \frac{\sqrt{1-p^2}}{1-pq^{-1}} \left(\frac{q^{-1}-p}{1-pq^{-1}} \right)^{i-1} \tag{4.9}$$

The parameters, $\{c_{i=1:5}, p, d\}$ are tuned using Bayesian hyper-parameter optimization. The above filters are low-pass filters that enable us to extract features with low-frequency content (slowly varying signals) (refer Appendix).

Feature extraction method 3: Probabilistic GSFA

Imparting non-linearity is straightforward in probabilistic GSFA without the need for non-linear expansions. With the dimension of the observation signal to be 200 ($M = 200$), we try the following different functions for $f_\phi(.)$ and $\boldsymbol{\mu}_{\theta_1}(.)$:

1. $f_\phi(.)$ and $\boldsymbol{\mu}_{\theta_1}(.)$ are linear with $P = M = 200$
2. $f_\phi(.)$ and $\boldsymbol{\mu}_{\theta_1}(.)$ are linear with $P = 400$
3. $f_\phi(.)$ and $\boldsymbol{\mu}_{\theta_1}(.)$ are two-layered neural networks (Tanh activation in between) with $P = 800$
4. $f_\phi(.)$ and $\boldsymbol{\mu}_{\theta_1}(.)$ are two-layered neural networks (Tanh activation in between) with $P = 1200$

Similar to GSFA, we try different filters:

1. $\Lambda_j(q^{-1}) = b_j q^{-1} + d_j q^{-2}$
2. $\Lambda_j(q^{-1}) = \sum_{i=1}^{79} b_{ij} q^{-i}$
3. $\Lambda_j(q^{-1}) = \frac{b_j q^{-1}}{1-d_j q^{-1}}$

The above filters are chosen to facilitate the extraction of latent signals with a high degree of auto-correlation (refer Appendix). For a fair comparison, we use the same classification head for all the methods. We use the Gaussian Naïve Bayes (GNB) classifier as the classification head. Taking into account the fact that the dataset is imbalanced, we report the following classification metrics: Accuracy and weighted F1 score on the test set in the table (4.3). From the table, we can see that GSFA outperforms all the methods with a weighted F1 score of 86.34% when the filter matrix is constructed from $G(q^{-1}) = 1 + \sum_{i=1}^{79} q^{-i}$. On the other hand, SFA has a weighted F1 score of 83.96%. Thus, there exist filters that can outperform the traditional SFA filter ($G(q^{-1}) = 1 - q^{-1}$) on this particular task. The flexibility provided by GSFA in choosing the filter matrix has proved advantageous. Please note that the accuracy of the pipeline can be improved further by choosing a better classifier. Since the motivation of this section is to highlight the advantages of GSFA over SFA, we do not focus on improving the performance through the inclusion of a better classification head. Though probabilistic GSFA has fallen behind, we can observe that adding non-linearity (in terms of additional layers) has improved the performance. The best weighted F1 score of 83.32% is obtained when $f_{\phi}(.)$ and $\boldsymbol{\mu}_{\theta_1}(.)$ are two-layered neural networks (Tanh activation in between) with $P = 1200$ and $\Lambda_j(q^{-1}) = \frac{b_j q^{-1}}{1-d_j q^{-1}}$. Experimenting with more filters and non-linear functions seems to be a promising direction for improving performance. Unlike SFA and GSFA, the parameters of probabilistic GSFA are estimated using numerical optimization techniques that are sensitive to initialization and are prone to get stuck in local minima. This could be one of the reasons for probabilistic GSFA falling behind SFA.

Probabilistic GSFA ($M = 200$)								
$\Lambda_j(q^{-1})$	Linear ($P = M$)		Linear ($P = 400$)		2-layer ($P = 800$)		2-layer ($P = 1200$)	
	Acc	F1	Acc	F1	Acc	F1	Acc	F1
$b_j q^{-1} + d_j q^{-2}$	80.06	80.21	80.94	81.17	82.69	82.91	82.55	82.82
$\sum_{i=1}^{79} b_{ij} q^{-i}$	79.40	79.54	80.14	80.30	82.79	83.01	82.86	83.12
$b_j q^{-1}/(1 - d_j q^{-1})$	79.78	79.92	80.77	80.96	82.59	82.79	83.16	83.32
GSFA								
$G(q^{-1})$	Acc				F1			
$1 + q^{-1} + q^{-2}$	83.53				84.04			
$1 + \sum_{i=1}^{79} q^{-i}$	85.79				86.34			
$\sum_{i=1}^{5} c_i F_i(q^{-1}, p, d)$	85.10				85.61			
SFA								
$G(q^{-1})$	Acc				F1			
$1 - q^{-1}$	83.57				83.96			

Table 4.3: GTSRB: Results on the test dataset

4.3 Fault Classification in Tennessee Eastman Process

The TE process was introduced by Downs and Vogel. There are 41 measured variables and 11 manipulated variables for the TE process. To enhance the TE process, 21 different faults are simulated. For the training dataset, there are 480 faulty samples for each fault (except faults 3, 9, and 15). For comparing our results with those presented in (Huang *et al.* (2020)), we do not include faults 3, 9, and 15. For the testing dataset, we have 800 samples for each fault. The measurements contain random noise which is carried out using a random seed. The sampling frequency for the TE process is 3 minutes. Before the fault classification procedure, the training set and the testing set are standardized. The samples in the training set are sorted according to the class label (similar to the section (4.2)). For a fair comparison, all the stages of the pipeline, except feature extraction, are kept the same so that the difference in the final evaluation metrics can be attributed only to the feature extraction algorithm.

Feature extraction method 1: SFA

We use a single-layered SFA with non-linear expansion (a compact expansion that only doubles the data dimension was employed, $\mathbf{x}^T \rightarrow [\mathbf{x}^T, |\mathbf{x}^T|^{0.8}]$ where the absolute value and exponent 0.8 are computed component-wise). Once the slow features are computed, the classification head is trained on it.

Feature extraction method 1: GSFA

We use a single-layered GSFA with non-linear expansion (a compact expansion that only doubles the data dimension was employed, $\mathbf{x}^T \rightarrow [\mathbf{x}^T, |\mathbf{x}^T|^{0.8}]$ where the absolute value and exponent 0.8 are computed component-wise). Once the slow features are computed, the classification head is trained on it. We try out filter matrices constructed from the transfer function operator:

$$G_d(q^{-1}) = 1 + \sum_{i=1}^{d} q^{-i} \tag{4.10}$$

Here, $d \in \{99, 199, 299, 399, 499, 599\}$. All the above filters are low-pass filters that enable us to extract features with low-frequency content (slow signals) (refer section

The simulation dataset of the TE process is provided here

A)).

Feature extraction method 1: Probabilistic GSFA

With $M = 52$, we try the following different functions for $f_\phi(.)$ and $\boldsymbol{\mu}_{\theta_1}(.)$:

1. $f_\phi(.)$ and $\boldsymbol{\mu}_{\theta_1}(.)$ are linear with $P = M = 52$
2. $f_\phi(.)$ and $\boldsymbol{\mu}_{\theta_1}(.)$ are linear with $P = 100$
3. $f_\phi(.)$ and $\boldsymbol{\mu}_{\theta_1}(.)$ are linear with $P = 200$
4. $f_\phi(.)$ and $\boldsymbol{\mu}_{\theta_1}(.)$ are linear with $P = 300$

Similar to GSFA, we try different filters:

1. $\Lambda_j(q^{-1}) = \sum_{i=1}^{50} b_{ij} q^{-i}$
2. $\Lambda_j(q^{-1}) = \sum_{i=1}^{100} b_{ij} q^{-i}$
3. $\Lambda_j(q^{-1}) = \frac{b_j q^{-1}}{1 - d_j q^{-1}}$

The above filters are chosen to facilitate the extraction of latent signals with a high degree of autocorrelation (refer Appendix). For a fair comparison, we use the same classification head for all the methods. We use the Support Vector Machine (SVM) classifier as the classification head (one vs one multiclass classification). Taking into account the fact that the dataset is imbalanced, we report the following classification metrics: Accuracy and weighted F1 score on the test set (refer table (4.4)). From table (4.4), we can see that probabilistic GSFA outperforms all the methods with a weighted F1 score of 62.77% when $\Lambda_j(q^{-1}) = \sum_{i=1}^{100} b_{ij} q^{-i}$. Though GSFA has a lower weighted F1 score, it still outperforms SFA. Once again, the flexibility provided by GSFA in choosing the filter matrix has proved advantageous. Since this task deals with process data that contains measurement and process noise, probabilistic GSFA is able to do a better job in extracting the latent signals compared to GSFA. This explains its improved performance. Please note that the accuracy of the pipeline can be improved further by choosing a better classifier. Since the motivation of this section is to highlight the advantages of GSFA (and probabilistic GSFA) over SFA, we refrain from improving the performance through the inclusion of a better classification head. Also, we do not have sufficient details to compare our results with those presented in (Huang *et al.* (2020)).

Probabilistic GSFA ($M = 52$)								
$\Lambda_j(q^{-1})$	Linear ($P = M$)		Linear ($P = 100$)		Linear ($P = 200$)		Linear ($P = 300$)	
	Acc	F1	Acc	F1	Acc	F1	Acc	F1
$\sum_{i=1}^{50} b_{ij} q^{-i}$	58.54	59.15	60.84	61.14	62.13	62.48	61.54	62.09
$\sum_{i=1}^{100} b_{ij} q^{-i}$	59.30	59.74	61.74	62.13	61.79	62.20	62.31	62.77
$b_j q^{-1}/(1 - d_j q^{-1})$	59.72	59.83	61.16	61.69	62.02	62.39	61.49	61.74
GSFA								
$G_d(q^{-1}) = 1 + \sum_{i=1}^{d} q^{-i}$	Acc				F1			
$d = 99$	59.29				56.97			
$d = 199$	58.34				56.38			
$d = 299$	58.63				56.51			
$d = 399$	58.75				56.08			
$d = 499$	58.75				56.49			
$d = 599$	56.75				55.18			
SFA								
$G(q^{-1})$	Acc				F1			
$1 - q^{-1}$	57.85				56.36			

Table 4.4: TE process: Results on the test dataset

CHAPTER 5

CONCLUSIONS

In this paper, we proposed Generalized Slow Feature Analysis (GSFA), which is a generalization of the SFA objective function in the frequency domain. By introducing a tunable filter, GSFA provides the flexibility to shape the frequency content of the extracted features, unlike SFA. Since GSFA and SFA do not take into account the stochastic nature of the observations, we introduced a probabilistic version of GSFA (probabilistic GSFA). In addition to handling noise, probabilistic GSFA can handle non-linear processes as well. The proposed methods were applied to tasks like state estimation and classification. GSFA outperformed SFA in all the above-mentioned tasks. Some interesting areas that need attention are: 1) The estimation algorithm for probabilistic GSFA relies on numerical optimization methods which do not guarantee convergence to the global minima and are sensitive to initialization. Because of this, we obtain sub-optimal estimates that degrade the performance. We should come up with better optimization strategies to solve this problem. 2) Though GSFA gives the freedom to work with different filters, more work is required to come up with suitable filters for particular tasks. One promising idea is to use Bayesian optimization for tuning the filter parameters. 3) Our idea of probabilistic GSFA closely resembles (or encompasses) the Kalman filter and its variants. Studying and comparing both methods in several numerical simulations/applications would be interesting.

APPENDIX A

MATHEMATICAL DETAILS

A.1 Linear SFA as a special case of Linear GSFA

For a predefined filter **G**, the original optimization problem for GSFA is given in section (2.1). From section (2.3), the objective function ψ_j in equation (2.1) follows,

$$\psi_j \;\propto\; \sum_{n=0}^{N-1} |G[n]X_j[n]|^2 \tag{A.1}$$

where $G[n]$ and $X_j[n]$ are the DFT of $g[k]$ (impulse response of $G(q^{-1})$) and $x_j[k]$ (j^{th} extracted feature), respectively. Hence, for every $j \in \{1, ..., P\}$, the optimization problem becomes,

$$\max_{f_j} \psi_j \;=\; \max_{f_j} \sum_{n=0}^{N-1} |G[n]X_j[n]|^2 \tag{A.2}$$

and the constraints (2.2) and (2.3) get translated to (for every $i, j \in \{1, ..., P\}$):

$$X_j[0] \;=\; 0 \tag{A.3}$$

$$\sum_{n=0}^{N-1} X_i[n]X_j^*[n] \;=\; N\delta_{ij} \tag{A.4}$$

Here, $X_j^*[n]$ represents the complex conjugate of $X_j[n]$. If we choose $G(q^{-1}) = 1+q^{-1}$,

$$|G[n]|^2 \;=\; 2\left[1 + \cos\left(2\pi\frac{n}{N}\right)\right] \tag{A.5}$$

Substituting equation (A.5) in (A.2),

$$\max_{f_j} \sum_{n=0}^{N-1} |G[n]X_j[n]|^2 \;=\; \max_{f_j} \sum_{n=0}^{N-1} 2\left[1 + \cos\left(2\pi\frac{n}{N}\right)\right] |X_j[n]|^2 \tag{A.6}$$

From constraint (A.4), $\sum_{n=0}^{N-1} |X_i[n]|^2 = N$ for all j. Hence,

$$\max_{f_j} \sum_{n=0}^{N-1} |G[n]X_j[n]|^2 = \max_{f_j} \sum_{n=0}^{N-1} 2\left[1+\cos\left(2\pi\frac{n}{N}\right)\right] |X_j[n]|^2 \tag{A.7}$$

$$= \min_{f_j} \sum_{n=0}^{N-1} 2\left[1-\cos\left(2\pi\frac{n}{N}\right)\right] |X_j[n]|^2 \tag{A.8}$$

$$= \min_{f_j} \sum_{n=0}^{N-1} |G_{SFA}[n]X_j[n]|^2 \tag{A.9}$$

$$= \min_{f_j} \frac{1}{N} \|\mathbf{G}_{SFA}\mathbf{x}_j\|^2 \tag{A.10}$$

The matrix $\mathbf{G}_{SFA}$ constructed using $G_{SFA}(q^{-1}) = 1 - q^{-1}$ is used to approximate the time derivative of a signal using backward difference. Note that $|G_{SFA}[n]|^2 = 2\left[1-\cos\left(2\pi\frac{n}{N}\right)\right]$, where $G_{SFA}[n]$ is the unitary DFT of the impulse response coefficients of $G_{SFA}(q^{-1})$. Since $\frac{1}{N}\|\mathbf{G}_{SFA}\mathbf{x}_j\|^2$ is the temporal average of the squared time derivative of $x_j[k]$, the original optimization problem of GSFA reduces to that of standard linear SFA when the matrix $\mathbf{G}$ is constructed from $G(q^{-1}) = 1 + q^{-1}$.

A.2 Probabilistic SFA as a special case of Probabilistic GSFA

Probabilistic SFA has the same synthesis or generative model, as that of probabilistic GSFA. For probabilistic SFA, the prior for the j^{th} latent signal $x_j[k]$ follows, for $j \in \{1, ..., P\}$

$$x_j[k] = -b_j x_j[k-1] + e_j[k] \tag{A.11}$$

$$= -b_j q^{-1} x_j[k] + e_j[k] \tag{A.12}$$

Here, $e_j[k] \sim GWN(0, \sigma_j^2)$. We can also write,

$$x_j[k] = H_j(q^{-1})e_j[k] \tag{A.13}$$

$$H_j(q^{-1}) = \frac{1}{1 + b_j q^{-1}} \tag{A.14}$$

From equations (A.13) and (A.14), we can see that choosing a filter according to (A.14) in probabilistic GSFA reduces the problem to probabilistic SFA. Thus, probabilistic GSFA subsumes probabilistic SFA.

A.3 Guidelines for choosing the filter

For the task of classification, we want the extracted features to be slowly varying with low-frequency content.

GSFA

According to equation (2.15), we can see that the squared magnitude of the DFT coefficients of the impulse response of the filter $G(q^{-1})$ act as a weighting function in the GSFA objective function. Hence, if we want to extract slow features with low-frequency content, the weighting function ($|G[n]|^2$) should give more weightage to low frequencies (small values of n). We can see that when $G(q^{-1})$ is a low-pass filter, we obtain such a weighting function.

In all the case studies, the form of $G(q^{-1})$ is

$$G_d(q^{-1}) = 1 + \sum_{i=1}^{d} q^{-i} \tag{A.15}$$

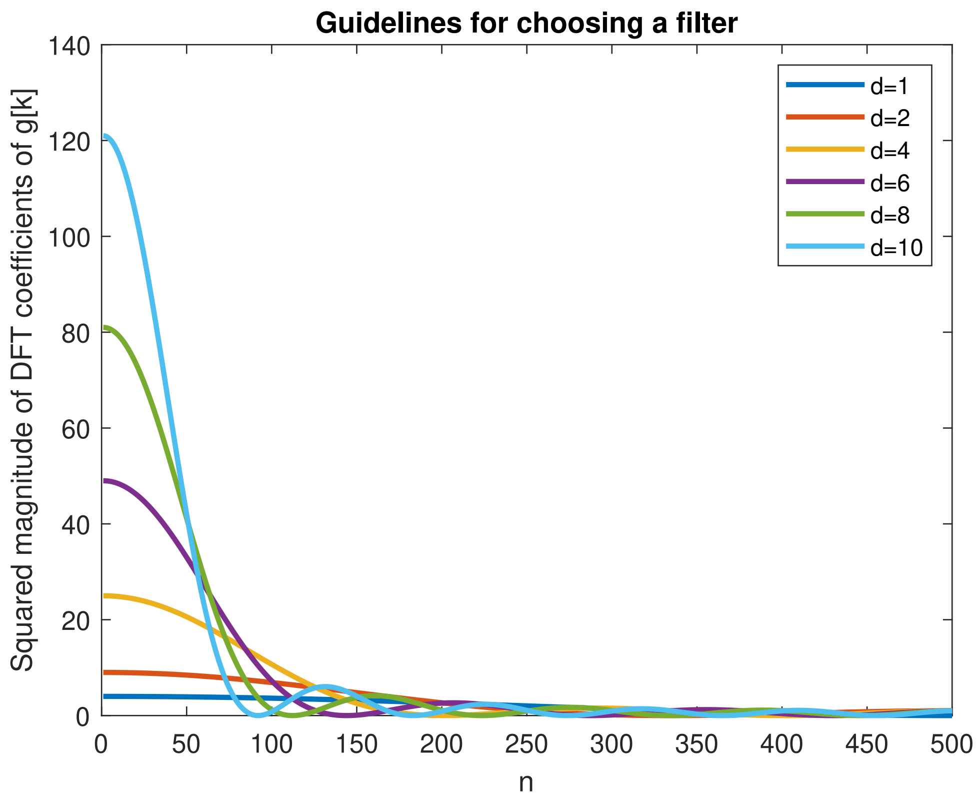

Figure A.1: Weighting function ($|G_d[n]|^2$) as a function of d

Figure (A.1) shows the weighting function for the above class of filters as a function of d. As the value of d increases, $|G_d[n]|^2$ becomes more concentrated at small values of n facilitating the extraction of features with low-frequency content. Motivated by this, we experimented with this class of filters and evaluated the final accuracy/F1 score in the case studies. In some applications, features with low-frequency content might not be suitable and we might not know a suitable filter for the task. In that case, we can resort to the following filter,

$$G(q^{-1}) = \sum_{i=1}^{l} c_i F_i(q^{-1}, p, d)$$

Here,

$$F_i(q^{-1}, p, d) = q^{-(1+d)} \frac{\sqrt{1-p^2}}{1-pq^{-1}} \left(\frac{q^{-1}-p}{1-pq^{-1}} \right)^{i-1} \tag{A.16}$$

The parameters, $\{c_{i=1:l}, p, d\}$ can be tuned using Bayesian hyper-parameter optimization. Depending on the parameter values, the shape of the weighting function changes.

Bayesian optimization tunes the parameter values (and hence, shapes the weighting function) accordingly to suit the final objective of the task.

Probabilistic GSFA

Signals that are strongly correlated in the temporal dimension tend to be slowly varying. In all the case studies, we chose $\Lambda_j(q^{-1})$ to be of the form

$$\Lambda_j(q^{-1}) = \sum_{i=1}^{p} b_{ij} q^{-i} \tag{A.17}$$

From equation (3.4),

$$\begin{aligned} x_j &= H_j(q^{-1}) e_j[k] && \text{(A.18)} \\ &= \frac{1}{1 + \sum_{i=1}^{p} b_{ij} q^{-i}} e_j[k] && \text{(A.19)} \end{aligned}$$

This follows an AR(p) process. We examine the power spectral density of $x_j[k]$ as p changes, assuming $b_{ij} = -0.1$ for all j.

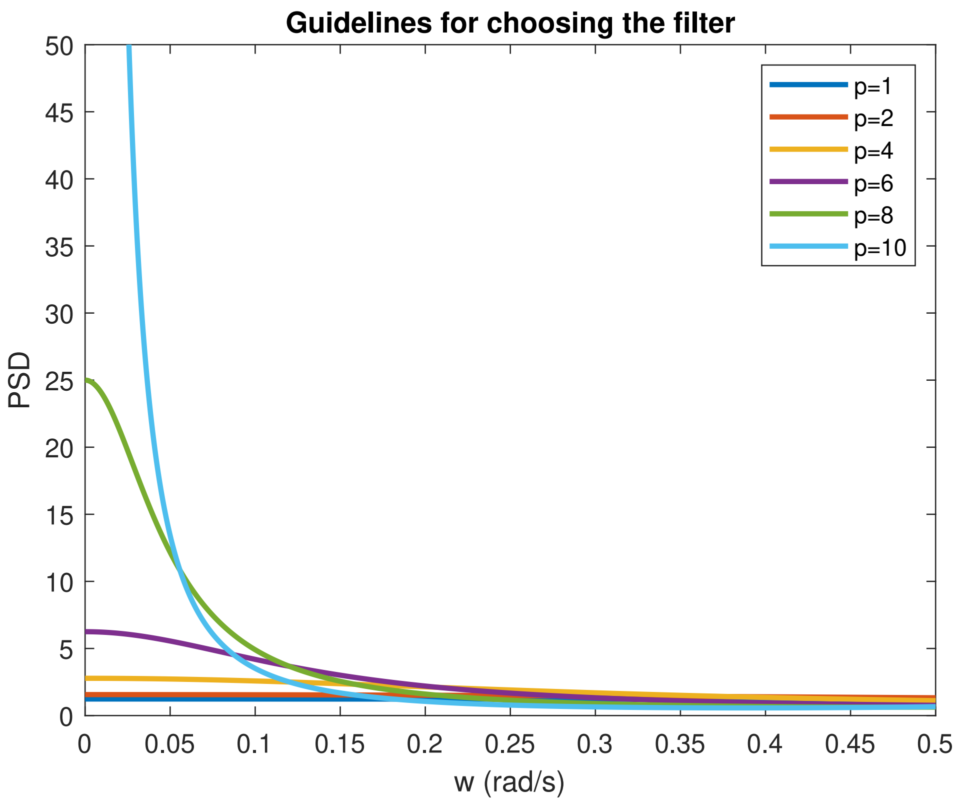

Figure A.2: Power spectral density (PSD) of $x_j[k]$ as a function of p

Figure (A.2) shows the PSD of $x_j[k]$ for the above class of filters as a function of p. As the value of p increases, the latent signals have predominantly low-frequency content and hence, they are slowly varying. Motivated by this, we experimented with this class of filters and evaluated the final accuracy/F1 score in the case studies.

REFERENCES

1. **Bishop, C. M.**, *Pattern Recognition and Machine Learning (Information Science and Statistics).* Springer, 2007, 1 edition. ISBN 0387310738. URL `http://www.amazon.com/Pattern-Recognition-Learning-Information-Statistics/dp/0387310738%3FSubscriptionId%3D13CT5CVB80YFWJEPWS02%26tag%3Dws%26linkCode%3Dxm2%26camp%3D2025%26creative%3D165953%26creativeASIN%3D0387310738`.

2. **Blaschke, T.**, **P. Berkes**, and **L. Wiskott** (2006). What is the relation between slow feature analysis and independent component analysis? *Neural computation*, **18**, 2495–508.

3. **Doumanoglou, A.**, **N. Vretos**, and **P. Daras** (2019). Frequency–based slow feature analysis. *Neurocomputing*, **368**.

4. **Escalante B., A. N.** and **L. Wiskott** (2012). Slow feature analysis: Perspectives for technical applications of a versatile learning algorithm. *KI - Künstliche Intelligenz*, **26**.

5. **Escalante B., A. N.** and **L. Wiskott** (2013). How to solve classification and regression problems on high-dimensional data with a supervised extension of slow feature analysis. *Journal of Machine Learning Research*, **14**, 3686–3719.

6. **Franzius, M.**, **N. Wilbert**, and **L. Wiskott**, Invariant object recognition with slow feature analysis. 2008. ISBN 978-3-540-87535-2.

7. **Houben, S.**, **J. Stallkamp**, **J. Salmen**, **M. Schlipsing**, and **C. Igel**, Detection of traffic signs in real-world images: The German Traffic Sign Detection Benchmark. *In International Joint Conference on Neural Networks*, 1288. 2013.

8. **Huang, J.**, **X. Yang**, **Y. A. W. Shardt**, and **X. Yan** (2020). Fault classification in dynamic processes using multiclass relevance vector machine and slow feature analysis. *IEEE Access*, **8**, 9115–9123.

9. **Jiang, C.**, **Y. Lu**, **W. Zhong**, **B. Huang**, **W. Song**, **D. Tan**, and **F. Qian** (2021). Deep bayesian slow feature extraction with application to industrial inferential modeling. *IEEE Transactions on Industrial Informatics*, **PP**, 1–1.

10. **Klampfl, S.** and **W. Maass**, Replacing supervised classification learning with slow feature analysis in spiking neural networks. *In Advances in Neural Information Processing Systems*. MIT Press, 2010. Annual Conference on Neural Information Processing Systems ; Conference date: 07-12-2009 Through 12-12-2009.

11. **Shang, C.**, **B. Huang**, **F. Yang**, and **D. Huang** (2016). Slow feature analysis for monitoring and diagnosis of control performance. *Journal of Process Control*, **39**, 21–34.

12. **Sprekeler, H.** (2011). On the relation of slow feature analysis and laplacian eigenmaps. *Neural Computation*, **23**(12), 3287–3302.

13. **Turner, R.** and **M. Sahani** (2007). A maximum-likelihood interpretation for slow feature analysis. *Neural Computation*, **19**(4), 1022–1038.

14. **Wiskott, L.** (2003*a*). Estimating driving forces of nonstationary time series with slow feature analysis.

15. **Wiskott, L.** (2003*b*). Slow Feature Analysis: A Theoretical Analysis of Optimal Free Responses. *Neural Computation*, **15**(9), 2147–2177. ISSN 0899-7667. URL `https://doi.org/10.1162/089976603322297331`.

16. **Wiskott, L.** and **T. J. Sejnowski** (2002). Slow feature analysis: Unsupervised learning of invariances. *Neural Comput.*, **14**(4), 715–770. URL `http://dblp.uni-trier.de/db/journals/neco/neco14.html#WiskottS02`.

17. **Zafeiriou, L.**, **M. A. Nicolaou**, **S. Zafeiriou**, **S. Nikitidis**, and **M. Pantic** (2016). Probabilistic slow features for behavior analysis. *IEEE Transactions on Neural Networks and Learning Systems*, **27**(5), 1034–1048.

18. **Zhang, Z.** and **D. Tao** (2019). Slow feature analysis for human action recognition.