POLITECNICO DI MILANO
DIPARTIMENTO DI ELETTRONICA, INFORMAZIONE E BIOINGEGNERIA
DOCTORAL PROGRAM IN INFORMATION TECHNOLOGY

# LEARNING AND ADAPTATION TO DETECT CHANGES AND ANOMALIES IN HIGH-DIMENSIONAL DATA

Doctoral Dissertation of:
**Diego Carrera**

Supervisor:
**Prof. Giacomo Boracchi**

Tutor:
**Prof. Cesare Alippi**

The Chair of the Doctoral Program:
**Prof. Barbara Pernici**

2018 – Cycle XXXI

This thesis is the result of a PhD course that has been funded by STMicroelectronics.

# Abstract

Monitoring a datastream to detect whether the incoming data departs from normal conditions is a problem encountered in many important applications, ranging from quality control in industrial process to health monitoring. Many solutions in the literature adopt a model that describes normal data and check whether this model is not able to describe new data, detecting anomalous instances or permanent changes affecting the data-generating process. Pursuing this approach is challenging when data have high dimensions or feature complex structures (as in case of images or signals), and these are the settings we consider in this thesis.

We address this problem from two different perspectives. At first, we model data as realization of a random vector, i.e., we assume that data can be described by a smooth probability density function. In these settings we focus on the change-detection problem, where the goal is to detect permanent changes affecting the data-generating process. In particular, we prove the detectability loss phenomenon, namely that performance of a popular change-detection algorithm that monitors the likelihood decreases when the data dimension increases. We also propose QuantTree, a novel algorithm to define histograms as density models for high dimensional data that are perfectly suit for change detection purposes. In fact, we prove that adopting QuantTree leads to an important property, i.e., that the distribution of any statistic computed over histograms generated by QuantTree does not depend on the distribution of the data-generating process. This enables non-parametric monitoring of multivariate datastreams with any statistic. Our experiments also show that combining several histograms computed by QuantTree into an ensemble can effectively mitigate the detectability loss phenomenon.

In the second part we focus on data that feature complex structures, that cannot be described by a smooth probability density function. We adopt dictionaries yielding sparse representations to characterize normal data and propose a novel anomaly-detection algorithm that detects as anomalous any data that do not conform to the learned dictionary. To make our anomaly-detection algorithm effective in practical application, we propose two domain adaptation algorithms that adapt the anomaly detector when the process generating normal data changes. The proposed algorithms have been successfully tested in two real world applications: a quality inspection system monitoring the production of nanofibrous materials through the analysis of Scanning Electron Microscope (SEM) images, and ECG monitoring using wearable devices. Finally, we investigate convolutional sparse representations, translation invariant extensions of traditional sparse representations that are gaining much attention in the last few years. In particular our analysis focuses on image denoising and show that convolutional sparse representations outperform their traditional counterparts only when the image admits an extremely sparse representation.

# Contents

CHAPTER 1

# Introduction

A primary concern in many applications is to detect whether a process departs from its normal conditions, since this could indicate a problem that has to be promptly alarmed and solved. Most often, changes have to be detected by monitoring a datastream generated by the process of interest. In this thesis, we investigate change and anomaly-detection problems from both theoretical and practical perspectives. In particular, we formally study the intrinsic difficulty of monitoring a datastream when the data dimension increases and propose novel algorithms to perform monitoring in two real world scenarios: industrial quality inspection and ECG monitoring. In what follows we describe these scenarios to better illustrate the challenges we have addressed along with our contributions.

Figure 1.1 shows some Scanning Electron Microscope (SEM) images acquired by a quality inspection system monitoring a prototype machine that produces nanofibrous materials. In normal conditions (Figure 1.1(a)), the produced materials are composed by tiny filaments randomly disposed, and problems affecting the production process might impair the normal structure, by creating defects such as films and beads, as those shown in Figure 1.1(b). Automatic detection of these defects allows to raise alerts as soon as the amount/size of defects exceeds a given tolerance level and

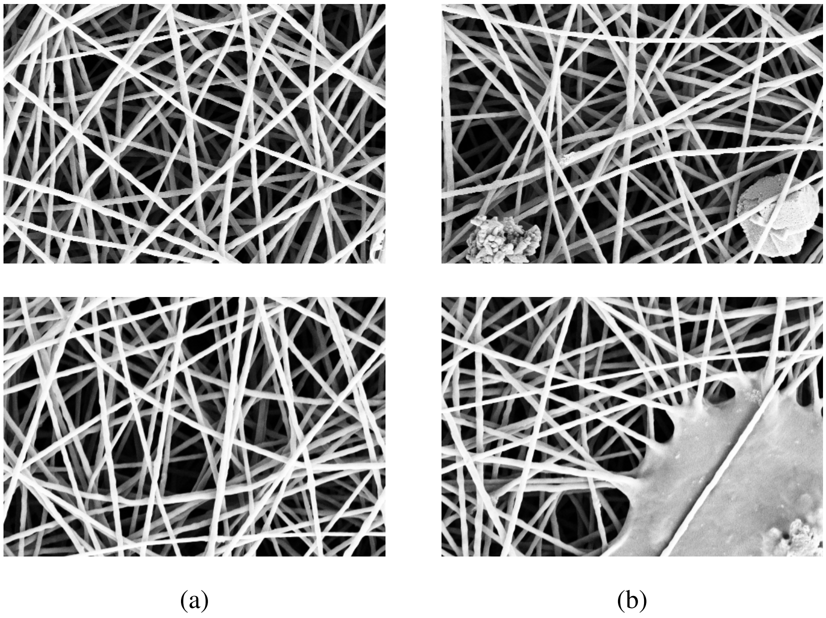


**Figure 1.1:** *Four details from SEM images depicting nanofibrous material acquired by the considered quality inspection system. (a) Two samples of good quality material. (b) The top image contains two sorts of localized defects: a small speck of dust at the bottom right and a bead, namely a fiber clot, at the bottom left. The bottom image contains a film, which is a thin layer of material among the fibers.*

guarantee a satisfactory production quality. Moreover, the quantitative assessment of the defects can be used to accurately design/tune the production process to both optimize the physical properties and control the defectiveness of the produced material.

While humans can easily identify the defects in images like that in Figure 1.1, thus distinguish between normal and defective regions, this is not a simple task for a machine. In fact, both normal and defective regions are far from being regular: fibers follow different orientations and randomly overlap, and defects can be very different in appearance and shape.

The other scenario considered is the monitoring of ECG signals using wearable devices. Wearable devices have a huge potential in health and fitness scenario, since they ease the transitioning from hospital to home/-mobile health monitoring. However, performing anomaly detection directly on these devices is far from being trivial. Figure 1.2 shows two segments of few seconds of ECG signals acquired from different users, containing both normal and anomalous heartbeats. Normal heartbeats feature a specific morphology, while anomalous ones, that can be due to acquisition errors, device displacements or dangerous arrhythmias, can have very different shapes. Designing rules to describe normal heartbeats and detect anomalous one is challenging because the morphology of normal heartbeats depend on the user (see Figure 1.2) and the position of the sensing device. Therefore, to be effective in online monitoring, an anomaly detector has to be customized on the specific user. This customization is even more important when the anomaly detector is embedded in a wearable device, since the user places the device by himself.

When monitoring this kind of datastreams we have to face several challenges. At first, data feature complex structures and are characterized by a high dimension (to locate defects in SEM images we analyze $15 \times 15$ patch, thus containing 225 pixel, while an heartbeat contains about 150 samples), and there does not exist any analytical model to describe them. Therefore, to perform successful monitoring, we have to resort to data-driven models, that are learned directly from data. However, we can acquired only data in normal conditions, since collecting data in alternative conditions is difficult, when not impossible. For example, in case of ECG monitoring it is not possible to collect a large number of anomalous heartbeats of the same user, since these might be due to potentially dangerous arrhythmias. This limitation prevents the use of supervised models, such as classifiers to address this kind of problem, since we do not have training data for both normal and anomalous classes. The only viable solution is to resort to unsupervised learning techniques to learn a data-driven model that describes only normal data.

The second challenge to be addressed is the design of specific indicators and decision rules that allow to successfully detect whether data are generated in alternative conditions. Moreover, the amount of false detections has to be controlled and kept under a given tolerance level to make the detector effective in practical applications.

Finally, the third challenge to be addressed is domain adaptation since normal conditions might change overt time. Thus, a model learned during the training phase, i.e., on data in the source domain, might not be able to describe new incoming data in the target domain. For example, in case of ECG monitoring, the heartbeats of a user get transformed when the heart rate increases, thus the morphology of heartbeats acquired during everyday activity is not the same of training ones, acquired in resting conditions.

In this thesis, we investigate these challenges from two different perspectives, which corresponds to two different modeling assumptions on the data-generating process. At first, we model data as a realizations of random vectors, thus we assume that there exists a sufficiently smooth probability

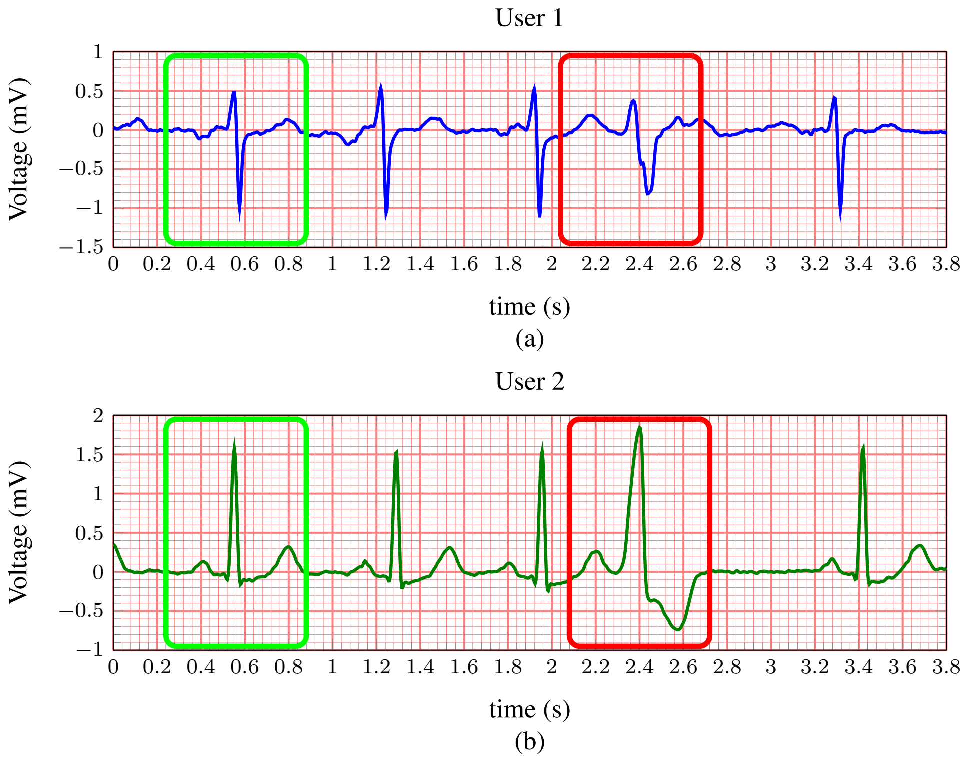


**Figure 1.2:** *Example of few seconds of ECG signals acquired by two different users. Both segments contain 4 normal heartbeats and an anomalous one (highlighted in the figure, with green and red boxes, respectively). From these plots, we also observe that the morphology of normal heartbeats depend on the user.*

density function describing data, as it is customary in the statistical process control and machine learning literature. Then, we consider the case when such probability density functions would not be easy to describe and we adopt models that provide meaningful representations to data. Such models have been firstly proposed in the signal and image processing literature, and are now the main object of the research on deep learning.

## 1.1 Original contribution

In the first part of the thesis, we model data as realization of a random vector and we focus on the change-detection problem, where the goal is to determined whether data-generating process permanently departs from normal to alternative conditions.

At first, we investigate the intrinsic difficulty of performing change detection when the data dimension increases. To this purpose, we formally define two measures to asses the change-magnitude, that depends only on the data distributions before and after the change, and the change-detectability, namely how perceivable the change is by a specific change-detection algorithm. In particular, we consider a popular change-detection algorithm based on the monitoring of the log-likelihood w.r.t. a learned model. We show the *detectability loss* problem [1], namely a decay in the change-detectability when the data dimension increases as the change-magnitude is kept fixed. We analytically prove this result in case of Gaussian datastreams, and empirically in case of real world data.

To perform our empirical analysis on the detectability loss, we develop CCM (Controlling Change Magnitude) [2, 3], a framework to manipulate real world datasets and introduce changes having a controlled magnitude. The framework can operate both in parametric settings, by approximating the stationary distribution as a Gaussian mixture, and in non-parametric ones. The developed framework enables a fair comparison between different change-detection algorithms on different datasets, and has been made publicly available for download.

Finally, we propose QuantTree [4], a novel algorithm to learn histograms for change-detection purposes. Peculiarity of the adaptive splitting scheme adopted by QuantTree is that it enables the non-parametric monitoring of datastreams. In particular, we theoretically prove that any statistic defined over such histograms does not depend on the data-generating distributions. This allows to control the false positive rate disregarding the data generating distribution. Moreover, to mitigate the effect of detectability loss, which affects also QuantTree, we propose an ensemble method that combines multiple histograms. Our experiments show that the non-deterministic nature of QuantTree increases the diversity of the computed histograms, improving the detection capability of the ensemble.

In the second part of the thesis we adopt a different modeling assumption: we assume that normal data can be well described by a dictionary yielding sparse representations, which is a model that have been successfully used in several signal and image processing applications. Moreover, we consider the anomaly detection problem, where the alternative conditions are not persistent, but affect only sporadic samples in the datastream.

We propose a novel anomaly-detection algorithm, which has at the core a data-driven dictionary that is learned on a training set of normal data. This dictionary provides sparse representation only to normal data, and we design low-dimensional indicators upon this representation to assess whether new data conform or not to normal ones, thus detect anomalies. Moreover, we propose two domain adaptation techniques to adapt these dictionaries when the process generating normal data changes. This general approach has been applied to ECG monitoring and the quality inspection system presented in the previous section.

We develop an anomaly detector for ECG signals that can be embedded in a wearable device to perform online and long-term monitoring. Our solution leverages a user-specific dictionary describing the morphology of normal heartbeats of the user [5]. This dictionary is learned during a user-configuration phase and on heartbeats acquired in resting conditions. Since during everyday activities the heart rate changes and thus normal heartbeats get transformed, we address the domain adaptation challenge by user-independent transformations that depends only on the heart rate [6]. These transformations can adapt both the user-specific dictionaries and the decision rule adopted to detect anomalous heartbeats [7], and are learned from public datasets containing long ECG signals of several users. We optimize our solution to be compatible with the limited resources available on a wearable device, and implement it on the Bio2Bit Dongle, a prototype wearable device produced by STMicroelectronics. We develop a demo implementing our solution [8] and fill a patent application [9, 10].

We customize the proposed anomaly detector for the quality inspection system described above [11]. Also in this scenario we have to face the domain adaptation problem, since novel images to be analyzed might be acquired at a magnification level that is different from training ones. In this case the structure of normal images would not be the same of that featured by training ones affecting the performance of the anomaly detector. To avoid the expensive, and often infeasible, acquisition of training images at different magnification levels, we propose a procedure to learn scale-invariant dictionaries from artificially generated normal training images at different scale. This is not sufficient to successfully detect anomalies, thus we design a specific sparse coding procedure by enforcing group sparsity in the representations [12]. By doing so, we can compute more powerful indicators to describe normal data and make our anomaly detector scale invariant.

Finally, we investigate the use of convolutional sparse representations as a powerful and very promising image model. These representations are gaining much attention in the literature and can be interpreted as the synthesis dual of popular convolutional neural networks (the corresponding analysis models). Our study [13] focuses on natural image denoising, which is the reference task to quantitatively assess the effectiveness of imaging models. In particular, we compare convolutional sparse representations with cycle spinning, a traditional image processing algorithm to compute translation-invariant representations. We show that the two approaches attain comparable performance in case of natural images and convolutional sparse representations outperform the cycle spinning only when the input image admits an extremely sparse representation.

The results presented in this thesis are partially presented in the following papers:

- Cesare Alippi, Giacomo Boracchi, Diego Carrera, Manuel Roveri. Change Detection in Multivariate Datastreams: Likelihood and Detectability Loss. In Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI), 2016, New York, New York, USA, July 9-15.

- Diego Carrera, Beatrice Rossi, Daniele Zambon, Pasqualina Fragneto, Giacomo Boracchi. ECG Monitoring in Wearable Devices by Sparse Models. In Proceedings of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery (ECML-PKDD), 2016, Riva del Garda, Italy, September 19-23.

- Diego Carrera, Giacomo Boracchi, Alessandro Foi, Brendt Wohlberg. Scale-invariant anomaly detection with multiscale group-sparse models. In Proceedings of the IEEE International Conference on Image Processing (ICIP), 2016, Phoenix, Arizona, USA, September 25-28.

- Cesare Alippi, Giacomo Boracchi, Diego Carrera. CCM: Controlling the Change Magnitude in High Dimensional Data. In Proceedings of the INNS Conference on Big Data, 2016, Thessaloniki, Greece, October 23-25.

- Diego Carrera, Fabio Manganini, Giacomo Boracchi, Ettore Lanzarone. Defect Detection in SEM Images of Nanofibrous Materials. IEEE Transactions on Industrial Informatics, Volume 13, Issue 2, April 2017.

- Diego Carrera, Giacomo Boracchi, Alessandro Foi, Brendt Wohlberg. Sparse overcomplete denoising: aggregation versus global optimization. IEEE Signal Processing Letters, Volume 24, Issue 10, October 2017.

- Diego Carrera, Beatrice Rossi, Pasqualina Fragneto, Giacomo Boracchi. Domain Adaptation for Online ECG Monitoring. In Proceedings of the IEEE International Conference on Data Mining (ICDM), 2017, New Orleans, Louisiana, USA, November 18-21.

- Diego Carrera, Giacomo Boracchi. Generating High-Dimensional Datastreams for Change detection. Big Data Research, Elsevier, Volume 11, March 2018.
- Giacomo Boracchi, Diego Carrera, Cristiano Cervellera, Danilo Macciò. QuantTree: Histograms for Change Detection in Multivariate Data Streams. In Proceedings of the International Conference Machine Learning (ICML), 2018, Stockholm, Sweden, July 10-15.
- Marco Longoni, Diego Carrera, Beatrice Rossi, Pasqualina Fragneto, Marco Pessione, Giacomo Boracchi. A Wearable Device for Online and Long-Term ECG Monitoring. In Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI) - Demo Track, 2018, Stockholm, Sweden, July 13-19.
- Diego Carrera, Beatrice Rossi, Pasqualina Fragneto, Giacomo Boracchi. Online anomaly detection for long-term ECG monitoring using wearable devices. Pattern Recognition, Volume 88, April 2019.

## 1.2 Structure of the thesis

The thesis is structured as follows. Chapter 2 presents a formal formulation to main problems addressed in the thesis. Then, we start the first part of the thesis, composed by Chapters 3-6, where data are modeled as realization of random vectors. Chapter 3 presents the related literature, while Chapter 4 is devoted to prove the detectability loss phenomenon. Then, Chapter 5 presents CCM, our framework to generate datastreams having changes of a given magnitude at a known location. Finally, Chapter 6 is devoted to QuantTree: here we introduce the algorithm and prove its theoretical properties. Moreover, we show how it can be combined in ensembles to mitigate the detectability loss.

In the second part of the thesis (Chapters 7-11) we focus on modeling data using sparse representations. The related literature is summarized Chapter 7, while 8 introduce our base anomaly detector and its applications to the considered application scenario. The two domain-adaptation algorithms are proposed in Chapter 9, and our complete monitoring solution is detailed in Chapter 10. Finally, Chapter 11 is devoted to investigate the convolutional sparse representations model, while Chapter 12 report our concluding remarks and future research directions.

# CHAPTER 2

# Problem Formulation

Let us consider a multivariate datastream $\{\mathrm{s}_t\}_{t=1,\ldots}$, where $\mathrm{s}_t \in \mathbb{R}^d$. We assume that in normal condition, each $\mathrm{s}_t$ is an i.i.d. realization of an unknown stochastic process $\mathcal{P}_N$. We are interested in monitoring the datastream and determine whether new data $\mathrm{s}_t$ are drawn from $\mathcal{P}_N$ or from an alternative stochastic process $\mathcal{P}_A$, that is completely unknown. As described in the previous chapters, we consider two different modeling assumptions on $\mathrm{s}_t$. At first, we model $\mathrm{s}_t$ as a realization of a random vector and address the change-detection problem (Section 2.1). Then, we consider the case when $\mathrm{s}_t$ features a complex structure thus $\mathcal{P}_N$ is not conveniently modeled as a random vector, and address the anomaly-detection and domain adaptation-problems, introduced in Sections 2.2 and 2.3, respectively. In both cases, we assume that a training set of normal data is available to learn a model that describes $\mathcal{P}_N$.

## 2.1 Change Detection

In this section we focus on the case when $\mathrm{s}_t$ is modeled as the realization of a continuous random vector S. In particular, we assume that stochastic processes $\mathcal{P}_N$ and $\mathcal{P}_A$ admit smooth probability density functions $\phi_0$ and

$\phi_1$, respectively. This assumption might seem too strict, but it is usually met in the statistical process control literature and in traditional supervised and unsupervised machine learning settings. In particular, when the datastream undergoes a feature extraction process, the extracted features can be modeled as realization of a continuous random vector.

Here we address the problem of detecting abrupt change in the data-generating process. More precisely, our goal is to detect if there is a change point $\tau \in \mathbb{N}$ in the datastreams such that

$$\mathrm{s}_t \sim \begin{cases} \phi_0 & t \leq \tau, \\ \phi_1 & t > \tau. \end{cases} \tag{2.1}$$

For sake of simplicity, we analyze the datastream in batches $W = \{\mathrm{s}_1, \dots, \mathrm{s}_\nu\}$ of $\nu$ samples. We detect changes by performing the following hypothesis test:

$$\begin{aligned} H_0 &: \ W \sim \phi_0 \\ H_1 &: \ W \nsim \phi_0 \end{aligned} \tag{2.2}$$

As customary in statistical hypothesis testing, a rejection region for (2.2) is defined by means of a test statistic $\mathcal{T}$ defined over $W$. We reject the null hypothesis $H_0$ if $\mathcal{T}(W) > \gamma$, where $\gamma$ is defined to guarantee a desired probability of type I error, also called False Positive Rate (FPR), namely

$$P_{\phi_0}(\mathcal{T}(W) > \gamma) \leq \alpha. \tag{2.3}$$

The statistic $\mathcal{T}$ is typically defined upon the model that approximate the density $\phi_0$.

Analyzing data in a batch-wise manner is not a genuine monitoring scheme. However, this mechanism is at the core of several online change-detection methods [14–17]. Moreover, the statistical power of test (2.2), namely the probability of rejecting the null hypothesis when the alternative holds, indicates the effectiveness of the test statistic $\mathcal{T}$ when the same is used in sequential-monitoring techniques.

Our goal is twofold: *i*) propose a new change-detection algorithm performing hypothesis test (2.2) and *ii*) investigate the impact of the dimension $d$ on change detection performance.

## 2.2 Anomaly Detection

In the second part of the thesis we focus on data featuring a specific structure, such as signals and images. In particular, we consider the anomaly detection problem, namely we analyze data $\mathrm{s}_t \in \mathbb{R}^d$ independently, and detect whether they are normal or anomalous, namely they are drawn from $\mathcal{P}_N$ or $\mathcal{P}_A$, respectively. Since there is no strict correlation with time, we will omit the subscript $_t$.

The assumption that s is drawn from a random vector having a smooth probability density function $\phi_0$ is not met. In fact, complex data $\mathrm{s} \in \mathbb{R}^d$ do not cover the entire high-dimensional space $\mathbb{R}^d$, but live in a low-dimensional manifold embedded in $\mathbb{R}^d$. In practice, this manifold is the support of the probability density function $\phi_0$ generating such data, thus $\phi_0$ cannot be smooth.

Here, we consider dictionary yielding sparse representations to approximate $\mathcal{P}_N$, i.e., we model $\mathrm{s} \sim \mathcal{P}_N$ as

$$\mathrm{s} \approx D\mathbf{x}, \tag{2.4}$$

where $D \in \mathbb{R}^{d\times n}$ is a matrix called *dictionary* and the coefficient vector $\mathbf{x} \in \mathbb{R}^n$ is *sparse* [18], namely it has few of nonzero components. This is equivalent to assume that the manifold where normal data live is a union of low-dimensional subspaces.

The anomaly detection problem boils down to defining a decision rule to determine whether s conforms or not to the learned dictionary. Our goal is to define a function $\mathcal{T}\colon \mathbb{R}^d \to \mathbb{R}$ and a threshold $\gamma$ such that

$$\text{s is anomalous} \quad \iff \quad \mathcal{T}(\mathrm{s}) > \gamma, \tag{2.5}$$

where $\mathcal{T}(\mathrm{s})$ is defined using the sparse representation $\mathbf{x}$ of s.

In what follows, we detail the anomaly detection problem in both application scenarios described in Chapter 1, i.e., the defect detection in SEM images, and the monitoring of ECG signals.

### 2.2.1 Defect Detection in SEM images

Let us denote by $s\colon \mathcal{X} \to \mathbb{R}^+$ the SEM image depicting the nanostructure to be analyzed, where $\mathcal{X} \subset \mathbb{Z}^2$ is the regular pixel grid corresponding to the image domain. The object our analysis are small squared patches defined as

$$\mathrm{s} = \{s(c+v),\ v \in \mathcal{V}\}, \tag{2.6}$$

where $c$ is the center of patch and $\mathcal{V}$ is a 2-D squared neighborhood of $\sqrt{d} \times \sqrt{d}$ pixels, where $d$ is the cardinality of $\mathcal{V}$. Some examples of patches extracted by normal images are shown in Figure 2.1(a). Although we address the anomaly detection problem in a patch-wise manner, our ultimate goal is to locate all the anomalous region in $s$; as such, the problem can be

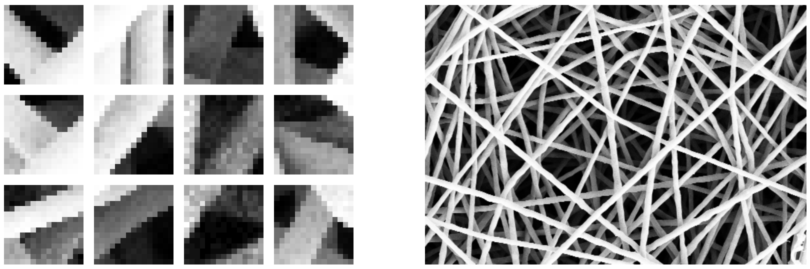

(a) (b)

**Figure 2.1:** *(a) Examples of* $15 \times 15$ *patches extracted from the normal images in (b).*

formulated as estimating an unknown anomaly mask

$$\Omega(c) = \begin{cases} 0 & \text{if } c \text{ falls inside a normal region} \\ 1 & \text{if } c \text{ falls inside a defective region} \end{cases}, \tag{2.7}$$

where $c \in \mathcal{X}$ denotes a pixel in the image $s$. Therefore, once we have detected all the anomalous patches in an image, we have to aggregate all these detections in single estimate of the anomaly mask $\Omega$.

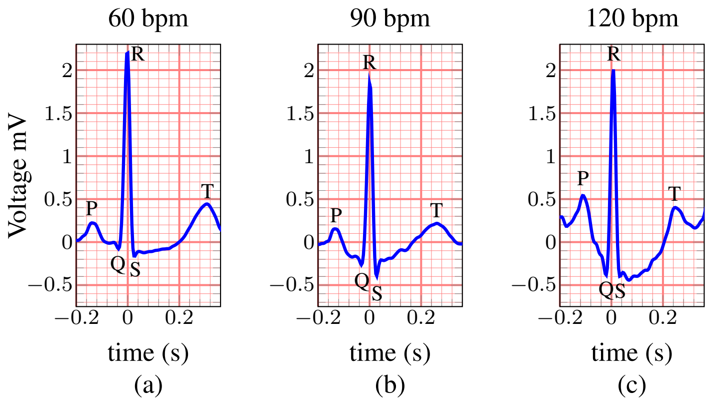
### 2.2.2 Anomaly detection in ECG signals

We denote by $s\colon \mathbb{N} \to \mathbb{R}$ the ECG signal which has been uniformly sampled in time and preprocessed to remove the baseline wander due to respiration and attenuate high-frequency noise, as in [19]. An heartbeat is a small segment of ECG signal that contains all the waveforms related to a single contraction of the heart, i.e., the P wave, the QRS complex and the T wave, as shown in Figure 2.2. More formally, we define an heartbeat s as

$$\mathrm{s} = \{s(t+v),\ u \in \mathcal{V}\}, \tag{2.8}$$

where $t$ is the location of the R-peak in the ECG signal, and $\mathcal{V}$ is a 1-D neighborhood of the origin that containing $d$ samples. The location of the R-peaks in ECG signals can be found using standard methods, e.g., the Pan-Tompkins algorithm [20]. Our goal is to perform online ECG monitoring by analyzing each heartbeat s as soon as it is acquired to detect whether it is normal or anomalous.

As shown in Figure 2.2, the heartbeats of each user feature a very specific morphology, which also depends on the sensing apparatus and the electrodes position [21]. As such, to successfully classify or detect anomalies in ECG signals, a single data-driven model is not able to accurately



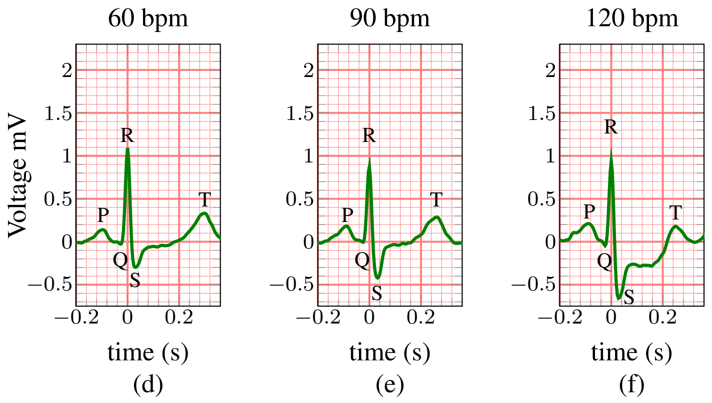


**Figure 2.2:** *Examples of heartbeats acquired at different heart rates measured in beats per minute (bpm) from two users. Letters indicate the waveforms in an ECG [24], namely the P wave, the QRS complex and the T wave, while the heart rates are reported over each plot. The heartbeats get transformed when the heart rate increases: the T-waves approach the QRS complexes, the QT intervals narrow down and the support of each heartbeat contracts (Figures (b), (c), (e), (f)). These heartbeats from different users seem to undergo a similar transformation, which does not seem to be simple dilation / contraction, since peaks change their intensities and shapes.*

describe all the users (even when trained on large datasets), and has to be directly learned / customized from the heartbeats of the specific user [22,23]. Therefore, we will denote the user-specific data generating process as $\mathcal{P}_{N_u}$ and learn a user-specific dictionary $D_u$ to approximate it.

## 2.3 Domain Adaptation

As described in Chapter 1, the process generating normal data $\mathcal{P}_N$ may change over time. Therefore, a dictionary learned on training data (i.e., in the source domain), might not be able to describe normal data during test (i.e., in the target domain). To avoid a degradation in the anomaly-detection performance we have to adapt our dictionary to track variations in $\mathcal{P}_N$. Our main assumption is that training data are available only in the source domain, as these are the settings that we faced in the considered scenarios, as described in what follows.

### 2.3.1 Defect Detection in SEM Images

Consider, for example, the images in Figure 2.3, acquired from different specimens of a nanofibrous material. While patches in these images are perceptually similar, their content is very different because the images are acquired at different magnification levels (i.e. scales). This means that an anomaly detector trained on an image like Figure 2.3(a) would possibly detect as anomalous patches from images like Figure 2.3(d), which should instead be considered as normal since only the scale has changed.

We assume that training images contain only patches generated by $\mathcal{P}_N$ and that these have been acquired at the maximum scale available. Thus, test images are acquired at the same or at a lower magnification. This is not a restrictive assumption in applications where the maximum magnification level is known beforehand.

Our goal is to increase the capabilities of model (2.4) by designing *i*) multiscale dictionaries $D$ that can describe data acquired at multiple magnification levels, *ii*) a specific sparse coding procedure to compute representations that exploit the multiscale property of the dictionary, and *iii*) more powerful indicators computed over these representations that make the anomaly detector scale invariant.

### 2.3.2 ECG Monitoring

As shown in Figure 2.2, heartbeats get transformed when the heart rate changes, thus the stochastic process generating normal heartbeats actually depends on the heart rate $r$, thus will be indicated by $\mathcal{P}_{N_{u,r}}$. We cannot learn a user-specific dictionary for each heart rate, since this would require to collect training heartbeats for every possible heart rate, that is impractical and potentially dangerous. Therefore, we have to modify the anomaly-detection algorithm to correctly operate at different heart rates at a controlled FPR.

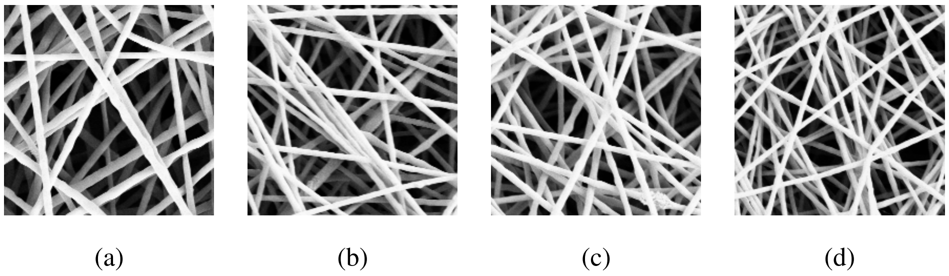


**Figure 2.3:** *Example of normal images acquired at different magnification levels. Patches extracted from images are perceptually similar, but their content is very different.*

In our case this requires adapting both the user-specific dictionary and the decision rule.

Our modeling assumption is that normal heartbeats acquired at heart rate $r_0$ admit a sparse representation w.r.t. to a dictionary $D_{u,r_0} \in \mathbb{R}^{d(r_0)\times n}$, where the number of samples in each heartbeat is $d(r_0)$ which also depends on the heart rate. We formulate dictionary adaptation as the problem of learning, for each target heart rate $r$, a user-independent transformation $\mathcal{F}_{r,r_0} \colon \mathbb{R}^{d(r_0)\times n} \to \mathbb{R}^{d(r)\times n}$ that operates as follows

$$D_{u,r} = \mathcal{F}_{r,r_0}(D_{u,r_0}) \quad \forall u, \tag{2.9}$$

and such that $D_{u,r}$ can properly approximate $\mathcal{P}_{N_{u,r}}$.

To adapt the decision rule (2.5) we tackle the problem of learning a set of user-independent transformations $f_{r,r_0} \colon \mathbb{R} \to \mathbb{R}$ to control the FPR. Considering (2.5), this consists in transforming the threshold $\gamma_{u,r_0}$:

$$\gamma_{u,r} = f_{r,r_0}(\gamma_{u,r_0}). \tag{2.10}$$

The user-independent transformations $\{\mathcal{F}_{r,r_0}\}$ and $\{f_{r,r_0}\}$ have to be learned from a collection $\{S_{u,r}\}$ of sets of normal heartbeats acquired at several heart rates from $L \gg 1$ users. We extract this collection from large publicly available datasets containing long ECG signals.

# Part I

# Random Vectors

# CHAPTER 3

# Related Literature

In this chapter we present an overview of the change-detection literature. We start by considering algorithms designed for univariate datastreams (Section 3.1), then we move to the multivariate case (Section 3.2), that is the main object of our analysis. Finally, we present the main methods that are strictly related to QuantTree, i.e., histograms in Section 3.3 and ensembles in Section 3.4.

## 3.1 Univariate Datastreams

The change-detection problem was firstly introduced in the statistical process control (SPC) literature [25], where data are modeled as i.i.d. realizations of a random variable. In SPC this problem was addressed in two phases, called Phase I and Phase II. In the Phase I, a fixed-size batch is analyzed to determine whether it contains a change point. In Phase II, the batch has no fixed size, but new samples are collected over time and a hypothesis test is performed at each time instant.

The first univariate change detection algorithms, such as CUSUM [26], EWMA [27], and Shewhart [28] charts, are designed to address the Phase II problem and detect shifts in the mean of the process $\mathcal{P}_N$. Although these algorithms are simple and well studied, their main drawback is that they are parametric, i.e., they assume that the pre-change distribution $\phi_0$ belongs to a known family, typically Gaussian distributions. There have been some works addressing the problem to detect shift in the variance of the monitored process [29,30] as well as changes in non-Gaussian distributions [31], but they all require the knowledge of $\phi_0$.

A meaningful scenario of known $\phi_0$ is classification in non-stationary environments, where changes are typically known as concept drift [32]. In this case, the monitored datastream is the sequence of classification error of a classifier, thus $\phi_0$ is known to be a Bernoulli distribution. In these settings many change-detection tests [16, 33–35] have been proposed in combination with an adaptation phase to avoid the performance degradation of the classifier.

There are two major approaches to make a change-detection test distribution free, and control the false positive rate without knowing $\phi_0$. The first one is to perform a data transformation that guarantees that data follows a known distribution. This approach is adopt in [36], where the Box-Cox transformation [37] is pursued to make data approximately Gaussian distributed. Another approach directly uses non-parametric statistics [17,38], such as the Kolmogorov-Smirnov [38], Mann-Whitney [39], Lepage [40], and order statistics [41].

## 3.2 Multivariate Datastreams

Several extensions of univariate change-detection tests to multivariate case have been presented in the literature, such as [42], [43] and [44], that extend the Shewart, the CUSUM and the EWMA, respectively. As in the univariate case, the main drawbacks of these tests is that they require the full knowledge of the pre-change distribution $\phi_0$. This assumption is relaxed in [45,46], where it is required that $\phi_0$ is Gaussian, but with unknown parameters.

The general approach to monitor multivariate datastreams when $\phi_0$ is completely unknown is to learn a model that approximate $\phi_0$ from a training set of data drawn from $\phi_0$, and then monitoring a test statistic defined upon the learned model, such as the likelihood (or the loglikelihood to avoid numerical issues). Therefore, the problem boils down to monitoring univariate datastreams. Some works use a parametric model such as Hidden Markov Models [47] and Gaussian Processes [48] that are very powerful model able to approximate a broad range of distributions. One of the most popular non-parametric approximation of $\phi_0$ is given by Kernel Density Estimation (KDE) [15, 49], that however becomes intractable when the data dimension increases.

In [14] $\phi_0$ is approximated by a Gaussian mixture Model, since these are flexible density models able to approximate even skewed and multi-modal distributions. Then, [14] monitors an upper bound of the log-likelihood computed w.r.t. the Gaussian mixture. Using this upper bound in place of the exact log-likelihood has two advantages: it avoids numerical problems, and makes the monitored statistic approximately distributed as a $\chi^2$ random variable. For these reasons, we also adopt Gaussian mixture in our experiments on detectability loss (see Chapter 4).

Other change-detection tests directly comnpute a multivariate statistics, such as [50, 51], that however depend on $\phi_0$, and threshold has to be estimated through bootstrap or permutation tests to properly control the FPR. Unfortunately, extending non-parametric change-detection tests to the multivariate case is far from being trivial. In fact, non-parametric statistics are all based on the natural order defined over real numbers, that is not well defined over $\mathbb{R}^d$. There are a few works in this direction: in [52] a bivariate Kolmogorov-Smirnov test is presented, but the extension to larger dimension is not trivial, as reported by the authors. An extension of the Mann-Whitney statistic based on multirank is presented in [53], but the computational complexity of this test is prohibitive in large dimension. Finally, [54] uses $k$-nearest neighbor distance to introduce an order in $\mathbb{R}^d$ and defines a test statistic, but it is not distribution free and thresholds have to be set using an asymptotic approximation.

## 3.3 Histograms for Change Detection

In this section we focus on histograms, which are perhaps the most natural candidates for describing densities, and, differently from likelihood based monitoring solutions, allow a comparison among distributions [55, 56]. However, they are often implemented over regular grids and require a number of bins that grows exponentially with the data dimension. As a consequence, many change-detection algorithms based on histograms are limited to the univariate setting [57], or resort to dimensionality-reduction techniques, such as PCA [58].

An alternative scheme that better scales in larger dimensions is presented in [56], where it is shown that histograms built on uniform-density partitions rather than regular grids provide superior detection performance. In [59] a recursive partitioning scheme based on tree (called $kqd$-tree) shows good change-detection performance also in high dimension. In particular, $kqd$-trees are introduced as a variant of $kd$-trees [60] to guarantee that all the leaves contain a minimum number of training samples and have a minimum size. The use of trees mitigates dimensionality issues, and offer efficient solutions to compute test statistics, which is a crucial aspect in high-throughput applications. However, trees have received very little attention in the change-detection literature, even though they have been widely used in data mining and machine learning. In particular, trees provide very efficient solutions to classical problems such as nearest-neighbor search [60, 61], classification and regression [62] as well as density estimation [63], even in high-dimensional data. The proposed QuantTree algorithm brings these successful models in the change-detection framework, providing solid theoretical foundations.

## 3.4 Ensembles of Change-Detection Algorithms

Ensembles of histograms can be referred to the ensemble literature, which concerns techniques to combine different models to address a specific task. From a purely statistical point of view, the use of ensembles of histograms for change detection can be framed in the context of multiple hypotheses testing, where a collection of statistical tests is simultaneously performed to answer a given question. Here a key issue is to control the Family-Wise Error Rate, namely the probability of one or more false positives in the individual tests. This is typically performed by adequately adjusting the thresholds (or the confidence) of every individual test, e.g., by the Bonferroni correction or the Holm method [64, Chapter 9]. From the machine-learning perspective, this approach can be generally ascribed to the family of ensemble learning algorithms. These techniques generate a predictive model through the combination of a collection of different base models (individuals). The generality of this approach has led to many different methods proposed in the literature (for a comprehensive description refer to [65] and references therein). Two major categories in the context of ensemble approaches are bagging [66] and boosting [67].

Thanks to their efficiency, trees have been widely used in ensemble methods. For instance, random forests [68] can be seen as an evolution of bagging trees that operates by constructing a collection of de-correlated trees with a random selection of features at every iteration. Ensemble models has been successfully employed in many classical learning problems, such as regression, classification and density estimation. The $kd$-trees [60] have been widely employed in ensemble methods for data mining and computer vision [69], where a collection of $kd$-trees is built exploiting various forms of randomization, to enable an approximate but very efficient implementation of $k$-nearest neighbor ($k$-NN) search.

It is important to remark that, even if ensembles of histograms and of trees in particular, have shown impressive success in the aforementioned learning problems, none of the examples reported above address the change-detection problem. The only few papers adopting ensemble methods for change-detection purposes [70, 71] do not concern histograms, as they are based on parametric models, such as Hidden Markov Models [70], or they are limited to aggregating multiple decisions from univariate change detectors monitoring each component of the input independently [71].

# CHAPTER 4

# Detectability Loss

In this chapter we investigate the intrinsic challenges of change-detection problem in high-dimensional datastreams. In particular, we show that, when the change magnitude is kept fixed, the change detectability decreases as the data dimension increases. We focus on a rather general change model and a popular change-detection approach, described in Sections 4.1 and 4.2, respectively. In these settings, we formally define the change magnitude, which measures the difference between the pre- and the post-change distributions, and the change detectability, which assesses how the change is perceivable by the considered change-detection algorithm. Our theoretical analysis (Section 4.3) shows that, when the change magnitude is kept fixed, the change detectability decreases as the data dimension increases. The same phenomenon affects also real world datasets, as we show in our experimental analysis in Section 4.4.

## 4.1 The Change Model

We assume that the random vector $\mathbf{S}$ in normal conditions admits a probability density function $\phi_0 \colon \mathbb{R}^d \to \mathbb{R}$ which is continuous, strictly positive over all $\mathbb{R}^d$ and bounded. This assumption it is necessary to ensure the existence of all the integrals in the chapter. We consider changes $\phi_0 \to \phi_1$ affecting the expectation and/or the correlation of S. In particular, $\phi_1 \colon \mathbb{R}^d \to \mathbb{R}$ has the following form:

$$\phi_1(\mathbf{s}) = \phi_0(Q\mathbf{s} + \mathbf{v})\,, \tag{4.1}$$

$\mathbf{v} \in \mathbb{R}^d$ changes the location of $\phi_0$, and $Q \in O(d) \subset \mathbb{R}^{d\times d}$ is an orthogonal matrix, i.e., $QQ^T = Q^TQ = I_d$, that modifies the correlation among the components of S. This rather general change-model requires a truly multivariate monitoring scheme: changes affecting only the correlation among components of S cannot be perceived by analyzing each component individually, or by extracting straightforward features (such as the norm) out of $\mathbf{s}_t$. Here, we do not consider changes affecting data dispersion as these can be easily detected by monitoring the Euclidean norm of $\mathbf{s}_t$.

## 4.2 The Considered Change-Detection Approach

We consider the popular change-detection approach that consists in monitoring the log-likelihood of $\mathbf{s}_t$ with respect to $\phi_0$ [14, 15, 72]:

$$\mathcal{L}(\mathbf{s}_t) = \log(\phi_0(\mathbf{s}_t)). \tag{4.2}$$

We denote by $L = \{\mathcal{L}(\mathbf{s}_t), t = 1, \dots\}$ the sequence of log-likelihood values, and observe that in stationary conditions, $L$ contains i.i.d. data drawn from a scalar random variable. When S undergoes a change, the distribution of $\mathcal{L}(\cdot)$ is also expected to change. Thus, changes $\phi_0 \to \phi_1$ can be detected by comparing the distribution of $\mathcal{L}(\cdot)$ over $W_P$ and $W_R$, two non-overlapping batches of $L$, where $W_P$ refers to past data (that we assume are generated from $\phi_0$), and $W_R$ refers to most recent ones (that are possibly generated from $\phi_1$). In practice, a suitable test statistic $\mathcal{T}(W_P, W_R)$ is computed to compare $W_P$ and $W_R$. When $\mathcal{T}(W_P, W_R) > \gamma$ we can safely consider that the log-likelihood values over $W_P$ and $W_R$ are from two different distributions, indicating indeed a change in S. The threshold $\gamma > 0$ controls the probability of type I error, i.e., the FPR.

As described in Chapter 2, $\phi_0$ in (4.2) is unknown and has to be preliminarily estimated from a training set of stationary data. Then, $\phi_0$ is simply replaced by its estimate $\widehat{\phi}_0$. We consider this quite simple change-detection approach since it makes easy to analyze how its performance is affected by the data dimension $d$.

## 4.3 Theoretical Analysis

This section sheds light on the relationship between change detectability and $d$. To this purpose, we introduce: *i*) a measure of the *change magnitude*, and *ii*) an indicator that quantitatively assesses *change detectability*, namely how difficult is to detect a change when monitoring $\mathcal{L}(\cdot)$ as described in Section 4.2. Afterward, we can study the influence of $d$ on the change detectability provided that changes $\phi_0 \to \phi_1$ have a constant magnitude.

### 4.3.1 Change Magnitude

The magnitude of $\phi_0 \to \phi_1$ can be naturally measured by the symmetric Kullback-Leibler divergence between $\phi_0$ and $\phi_1$ (also known as Jeffreys divergence):

$$\begin{aligned}\text{sKL}(\phi_0, \phi_1) &:= \text{KL}(\phi_0, \phi_1) + \text{KL}(\phi_1, \phi_0) \\ &= \int_{\mathbb{R}^d} \log \frac{\phi_0(\mathrm{s})}{\phi_1(\mathrm{s})} \phi_0(\mathrm{s}) d\mathrm{s} + \int_{\mathbb{R}^d} \log \frac{\phi_1(\mathrm{s})}{\phi_0(\mathrm{s})} \phi_1(\mathrm{s}) d\mathrm{s}\,.\end{aligned} \tag{4.3}$$

This choice is supported by the Stein's Lemma [73], which states that $\text{KL}(\phi_0, \phi_1)$ yields an upper-bound for the power of parametric hypothesis tests that determine whether a given sample population is generated from $\phi_0$ (null hypothesis) or $\phi_1$ (alternative hypothesis). In practice, large values of $\text{sKL}(\phi_0, \phi_1)$ indicate changes that are very apparent, since hypothesis tests designed to detect either $\phi_0 \to \phi_1$ or $\phi_1 \to \phi_0$ can be very powerful.

### 4.3.2 Change Detectability

We define the following indicator to quantitatively assess the detectability of a change when monitoring $\mathcal{L}(\cdot)$.

**Definition 4.1.** *The* signal-to-noise ratio *(SNR) of the change $\phi_0 \to \phi_1$ is defined as:*

$$\text{SNR}(\phi_0, \phi_1) := \frac{\left( \underset{\mathrm{s}\sim\phi_0}{E}[\mathcal{L}(\mathrm{s})] - \underset{\mathrm{s}\sim\phi_1}{E}[\mathcal{L}(\mathrm{s})] \right)^2}{\underset{\mathrm{s}\sim\phi_0}{\text{var}}[\mathcal{L}(\mathrm{s})] + \underset{\mathrm{s}\sim\phi_0}{\text{var}}[\mathcal{L}(\mathrm{s})]}, \tag{4.4}$$

*where* $\text{var}[\cdot]$ *denotes the variance of a random variable.*

In particular, $\text{SNR}(\phi_0, \phi_1)$ measures the extent to which $\phi_0 \to \phi_1$ is detectable by monitoring the expectation of $\mathcal{L}(\cdot)$. In fact, the numerator of (4.4) corresponds to the shift introduced by $\phi_0 \to \phi_1$ in the expectation of $\mathcal{L}(\cdot)$ (i.e., the relevant information, the *signal*) which is easy/difficult to detect relatively to its random fluctuations (i.e., the *noise*), which are assessed in the denominator of (4.4). Note that, if we replace the expectations and the variances in (4.4) by their sample estimators, we obtain that $\text{SNR}(\phi_0, \phi_1)$ corresponds – up to a scaling factor – to the squared statistic of a Welch's $t$-test [74], that detects changes in the expectation of two sample populations. This is another argument supporting the use of $\text{SNR}(\phi_0, \phi_1)$ as a measure of change detectability.

The following proposition relates the change magnitude $\text{sKL}(\phi_0, \phi_1)$ with the numerator of (4.4).

**Proposition 4.1.** *Let us consider a change $\phi_0 \to \phi_1$ such that*

$$\phi_1(\mathbf{s}) = \phi_0(Q\mathbf{s} + \mathbf{v}) \tag{4.5}$$

*where $Q \in \mathbb{R}^{d\times d}$ is orthogonal and $\mathbf{v} \in \mathbb{R}^d$. Then, it holds:*

$$\text{sKL}(\phi_0, \phi_1) \geq \underset{\mathbf{s}\sim\phi_0}{E}[\mathcal{L}(\mathbf{s})] - \underset{\mathbf{s}\sim\phi_1}{E}[\mathcal{L}(\mathbf{s})] \tag{4.6}$$

*Proof.* From the definition of $\text{sKL}(\phi_0, \phi_1)$ in (4.3) it follows

$$\begin{aligned} \text{sKL}(\phi_0, \phi_1) = \underset{\mathbf{s}\sim\phi_0}{E}[\log(\phi_0(\mathbf{s}))] - \underset{\mathbf{s}\sim\phi_0}{E}[\log(\phi_1(\mathbf{s}))]+ \\ + \underset{\mathbf{s}\sim\phi_1}{E}[\log(\phi_1(\mathbf{s}))] - \underset{\mathbf{s}\sim\phi_1}{E}[\log(\phi_0(\mathbf{s}))]\,. \end{aligned} \tag{4.7}$$

Since $\mathcal{L}(\cdot) = \log(\phi_0(\cdot))$, (4.6) holds if and only if

$$\underset{\mathbf{s}\sim\phi_1}{E}[\log(\phi_1(\mathbf{s}))] - \underset{\mathbf{s}\sim\phi_0}{E}[\log(\phi_1(\mathbf{s}))] \geq 0. \tag{4.8}$$

From (4.5) it follows that $\phi_0(\mathbf{s}) = \phi_1(Q^T(\mathbf{s} - \mathbf{v}))$, thus, by replacing the mathematical expectations with their integral expressions, (4.8) becomes

$$\int \log(\phi_1(\mathbf{s}))\phi_1(\mathbf{s})d\mathbf{s} - \int \log(\phi_1(\mathbf{s}))\,\phi_1(Q^T(\mathbf{s} - \mathbf{v}))d\mathbf{s} \geq 0 \tag{4.9}$$

Let us define $\mathbf{y} = Q^T(\mathbf{s} - \mathbf{v})$, then $\mathbf{s} = Q\mathbf{y} + \mathbf{v}$ and $d\mathbf{s} = |\det(Q)|d\mathbf{y} = d\mathbf{y}$, since $Q$ is orthogonal. Using this change of variables in the second summand of (4.9) we obtain

$$\int \log(\phi_1(\mathbf{s}))\phi_1(\mathbf{s})d\mathbf{s} - \int \log(\phi_1(Q\mathbf{y} + \mathbf{v}))\,\phi_1(\mathbf{y})d\mathbf{y} \geq 0. \tag{4.10}$$

Finally, defining $\phi_2(\mathbf{y}) := \phi_1(Q\mathbf{y} + \mathbf{v})$ turns (4.10) into

$$\int \log(\phi_1(\mathbf{s}))\phi_1(\mathbf{s})d\mathbf{s} - \int \log\left(\phi_2(\mathbf{y})\right)\phi_1(\mathbf{y}))d\mathbf{y} \geq 0, \tag{4.11}$$

which holds since the left-hand-side of (4.11) is $\mathrm{KL}(\phi_1, \phi_2)$, that is always non-negative.

□

### 4.3.3 Detectability Loss

It is now possible to investigate the intrinsic challenge of change-detection problems when data dimension increases. In particular, we study how the change detectability (i.e., $\mathrm{SNR}(\phi_0, \phi_1)$) varies when $d$ increases and changes $\phi_0 \to \phi_1$ preserve constant magnitude (i.e., $\mathrm{sKL}(\phi_0, \phi_1) = const$). Unfortunately, since there are no general expressions for the variance of $\mathcal{L}(\cdot)$, we have to assume a specific distribution for $\phi_0$ to carry out any analytical development. As a relevant example, we consider Gaussian random variables, which enable a simple expression of $\mathcal{L}(\cdot)$. The following theorem demonstrates the *detectability loss* for Gaussian distributions, namely that $\mathrm{SNR}(\phi_0, \phi_1)$ decays as $d$ increases.

**Theorem 4.1.** *Let $\phi_0 = \mathcal{N}(\mu_0, \Sigma_0)$ be a $d$-dimensional Gaussian pdf and $\phi_1 = \phi_0(Q\mathbf{s} + \mathbf{v})$, where $Q \in \mathbb{R}^{d\times d}$ is orthogonal and $\mathbf{v} \in \mathbb{R}^d$. Then, it holds*

$$\mathrm{SNR}(\phi_0, \phi_1) \leq \frac{C}{d} \tag{4.12}$$

*where the constant $C$ depends only on $\mathrm{sKL}(\phi_0, \phi_1)$.*

*Proof.* Basic algebra leads to the following expression for $\mathcal{L}(\mathbf{s})$ when $\phi_0 = \mathcal{N}(\mu_0, \Sigma_0)$:

$$\mathcal{L}(\mathbf{s}) = -\frac{1}{2}\log\left((2\pi)^d \mathrm{det}(\Sigma_0)\right) - \frac{1}{2}(\mathbf{s} - \mu_0)^T \Sigma_0^{-1}(\mathbf{s} - \mu_0)\,. \tag{4.13}$$

The first term in the right-hand-side of (4.13) is constant, while the second term is distributed as a chi-squared having $d$ degrees of freedom. Therefore,

$$\underset{\mathbf{s}\sim\phi_0}{\mathrm{var}}\left[\mathcal{L}(\mathbf{s})\right] = \mathrm{var}\left[-\frac{1}{2}\chi^2(d)\right] = \frac{d}{2}\,. \tag{4.14}$$

Then, from the definition of $\mathrm{SNR}(\phi_0, \phi_1)$ in (4.4) and Proposition 4.1, it follows that

$$\begin{aligned}\mathrm{SNR}(\phi_0, \phi_1) &\leq \frac{\mathrm{sKL}(\phi_0, \phi_1)^2}{\underset{\mathrm{s}\sim\phi_0}{\mathrm{var}}[\mathcal{L}(\mathrm{s})] + \underset{\mathrm{s}\sim\phi_1}{\mathrm{var}}[\mathcal{L}(\mathrm{s})]} \leq \frac{\mathrm{sKL}(\phi_0, \phi_1)^2}{\underset{\mathrm{s}\sim\phi_0}{\mathrm{var}}[\mathcal{L}(\mathrm{s})]} = \\ &= \frac{\mathrm{sKL}(\phi_0, \phi_1)^2}{d/2} = \frac{C}{d}\,.\end{aligned}$$

□

Theorem 4.1 shows detectability loss for Gaussian distributions. In fact, when $d$ increases and $\mathrm{sKL}(\phi_0, \phi_1)$ remains constant, $\mathrm{SNR}(\phi_0, \phi_1)$ is upper-bounded by a function that monotonically decays as $1/d$. The decaying trend of $\mathrm{SNR}(\phi_0, \phi_1)$ indicates that detecting changes becomes more difficult when $d$ increases. Moreover, the decaying rate does not depend on $\mathrm{sKL}(\phi_0, \phi_1)$, thus this problem equally affects all possible changes $\phi_0 \to \phi_1$ defined as in (4.1), disregarding their magnitude.

Theorem 4.1 implicates detectability loss only when $\mathrm{sKL}(\phi_0, \phi_1)$ is kept constant. Assuming constant change magnitude is necessary to correctly investigate the influence of the sole data dimension $d$ on the change detectability. In fact, when the change magnitude increases with $d$, changes might become even easier to detect as $d$ grows. This is what experiments in [75, Section 2.1] show, where outliers become easier to detect when $d$ increases. However, in that experiment, the change-detection problem becomes easier as $d$ increases, since each component of s carries additional information about the change, thus increases $\mathrm{sKL}(\phi_0, \phi_1)$.

Detectability loss can be also proved when $\phi_0$ is non Gaussian, as far as its components are independent. In fact, if $\phi_0(\mathrm{s}) = \prod_{i=0}^{d} \phi_0^{(i)}(s^{(i)})$, where $(\cdot)^{(i)}$ denotes either the marginal of a pdf or the component of a vector, thus

$$\log(\phi_0(\mathrm{s})) = \log\left(\prod_{i=0}^{d} \phi_0^{(i)}(x^{(i)})\right) = \sum_{i=0}^{d} \log\left(\phi_0^{(i)}(x^{(i)})\right), \qquad (4.15)$$

from which we deduce the following:

$$\underset{\mathrm{s}\sim\phi_1}{\mathrm{var}}[\mathcal{L}(\mathrm{s})] = \underset{\mathrm{s}\sim\phi_0}{\mathrm{var}}\left[\sum_{i=0}^{d} \log\left(\phi_0^{(i)}(x^{(i)})\right)\right] = \sum_{i=0}^{d} \underset{\mathrm{s}\sim\phi_0}{\mathrm{var}}\left[\log\left(\phi_0^{(i)}(x^{(i)})\right)\right], \qquad (4.16)$$

since $\log(\phi_0^{(i)}(s^{(i)}))$ are independent. Clearly, (4.16) increases with $d$, since its summands are positive. Thus, also in this case, the upper bound of $\mathrm{SNR}(\phi_0, \phi_1)$ decays with $d$ when $\mathrm{sKL}(\phi_0, \phi_1)$ is kept constant.

Remarkably, detectability loss does not depend on how the change $\phi_0 \to \phi_1$ affects $\mathbf{S}$, as far as $\text{sKL}(\phi_0, \phi_1) = const$. Our results hold, for instance, when either $\phi_0 \to \phi_1$ affects all the components of $\mathbf{S}$ or some of them remain irrelevant for change-detection purposes. Moreover, detectability loss occurs independently of the specific change-detection method used on the log-likelihood (e.g. sequential analysis, or window comparison), as our results concern $\text{SNR}(\phi_0, \phi_1)$ only.

In the next section we show that detectability loss affects also real-world change-detection problems. To this purpose, we design a rigorous empirical analysis to show that the power of customary hypothesis tests actually decreases with $d$ when data are non Gaussian and possibly correlated.

## 4.4 Empirical Analysis

Our empirical analysis has been designed to show that:

- $\text{SNR}(\phi_0, \phi_1)$, which is the underpinning element of our theoretical result, is a suitable measure of change detectability. In particular, we show that the power of hypothesis tests able to detect both changes in mean and in variance of $\mathcal{L}(\cdot)$ also decays.
- Detectability loss is not due to density-estimation problems, but it becomes a more serious issue when $\phi_0$ is estimated from training data.
- Detectability loss occurs also in Gaussian mixtures, and also on high-dimensional real-world datasets, which are far from being Gaussian or having independent components.

We address the first two points in Section 4.4.1, while the third one in Sections 4.4.2 and 4.4.3.

In our experiments, the change-detection performance is assessed by numerically computing the power of two customary hypothesis tests, namely the Lepage [40] and the one-sided $t$-test on data batches $W_P$ and $W_R$ which contains 500 data each. The choice of a one-sided test over a two sided one is motivated by the fact that we can assume that $\phi_0 \to \phi_1$ decreases the expectation of $\mathcal{L}$ since

$$\underset{\mathbf{s}\sim\phi_0}{E}[\log(\phi_0(\mathbf{s}))] - \underset{\mathbf{s}\sim\phi_1}{E}[\log(\phi_0(\mathbf{s}))] \geq 0 \tag{4.17}$$

follows from (4.8). As we discussed in Section 4.3.2, the $t$-statistic on the log-likelihood is closely related to $\text{SNR}(\phi_0, \phi_1)$, while the Lepage is a nonparametric statistic that detects both location and scale changes, as it is defined as the sum of the squares of the Mann-Whitney and Mood statistics, see also [17]. To compute the power, we set $\gamma$ to guarantee a significance level $\alpha = 0.05$. In case of the Lepage statistic the value of $\gamma$ is given by the asymptotic approximation of the statistic in [40], as such approximations is very precise in our case, where we consider two batches of 500 samples each.

We synthetically introduce changes $\phi_0 \to \phi_1$ using CCM, a theoretically grounded framework to generate datastreams affected by changes that we will be presented in the next chapter. CCM allows to control sKL$(\phi_0, \phi_1)$, that is set to $1$ in all our experiments. In the univariate Gaussian case, this change magnitude corresponds to $\mathbf{v}$ equals to the standard deviation of $\phi_0$.

### 4.4.1 Gaussian Datastreams

We generate Gaussian datastreams having dimension $d \in \{1, 2, 4, \ldots, 128\}$ and, for each value of $d$, we prepare 10000 runs, with $\phi_0 = \mathcal{N}(\mu_0, \Sigma_0)$ and $\phi_1 = \mathcal{N}(\mu_1, \Sigma_1)$. The parameters $\mu_0 \in \mathbb{R}^d$ and $\Sigma_0 \in \mathbb{R}^{d\times d}$ have been randomly generated, while $\mu_1 \in \mathbb{R}^d$ and $\Sigma_1 \in \mathbb{R}^{d\times d}$ have been set to yield sKL$(\phi_0, \phi_1) = 1$. In each run we generate 1000 samples: $\{\mathrm{s}_t,\ t = 1, \ldots, 500\}$ from $\phi_0$, and $\{\mathrm{s}_t,\ t = 501, \ldots, 1000\}$ from $\phi_1$. Then, we compute the datastream $L = \{\mathcal{L}(\mathrm{s}_t),\ t = 1, \ldots, 1000\}$, and define $W_P = \{\mathcal{L}(\mathrm{s}_t), t = 1, \ldots, 500\}$ and $W_R = \{\mathcal{L}(\mathrm{s}_t), t = 501, \ldots, 1000\}$.

We repeat the same experiment replacing $\phi_0$ with its estimate $\widehat{\phi}_0(\mathrm{s})$, where $\widehat{\mu}_0$ and $\widehat{\Sigma}_0$ are computed using the sample estimators over an additional training set $S_0$ whose size grows linearly with $d$, i.e. $\#S_0 = 100 \cdot d$. We denote by $\widehat{L} = \{\widehat{\mathcal{L}}(\mathrm{s}(t)), t = 1, \ldots, 1000\}$ the sequence of estimated log-likelihood values. Finally, we repeat the whole experiments keeping $\#S_0 = 100$ for any value of $d$, and we denote by $\widehat{L}_{100}$ the corresponding sequence of log-likelihood values.

Figure 4.1(a) shows that the power of both the Lepage and one-sided $t$-test substantially decrease when $d$ increases. This result is coherent with our theoretical analysis of Section 4.3, and confirms that SNR$(\phi_0, \phi_1)$ is a suitable measure of change detectability. While it is not surprising that the power of the $t$-test decays, given its connection with the SNR$(\phi_0, \phi_1)$, it is remarkable that the power of the Lepage test also decays, as this fact indicates that it becomes more difficult to detect both changes in the mean and in the dispersion of $L$. The decaying power of both tests indicates that the corresponding test statistics decrease with $d$, which imply larger detection delays when using this statistics in sequential monitoring schemes.

Note that detectability loss is not due to density-estimation issues, but

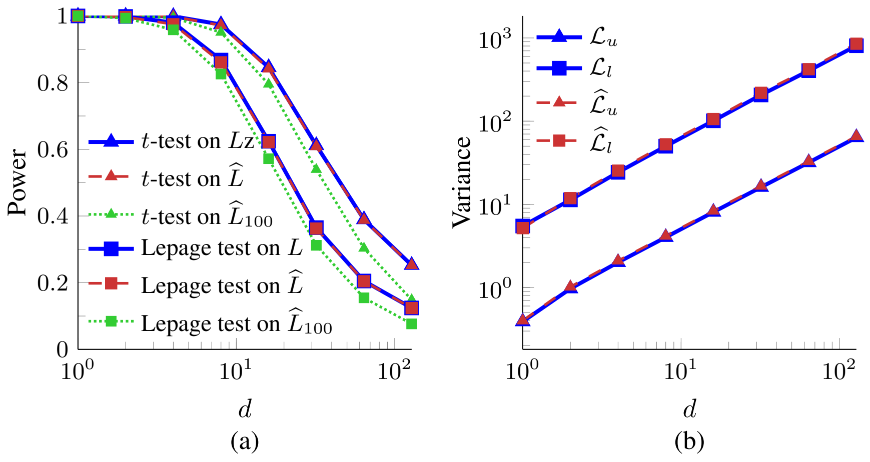


**Figure 4.1:** *(a) Power of the Lepage and one-sided $t$-test empirically computed on sequences generated as in Section 4.4.1. Detectability loss clearly emerges when the log-likelihood is computed using $\phi_0$ (denoted by $L$) or its estimates fitted on $100 \cdot d$ samples ($\widehat{L}$) or from 100 samples ($\widehat{L}_{100}$). (b) The sample variance of $\mathcal{L}_u(\cdot)$ (4.19) and $\mathcal{L}_l(\cdot)$ (4.20) computed as in Section 4.4.2. As in the Gaussian case, both these variances grow linearly with $d$ and similar results hold when using $\widehat{\phi}_0$, which is estimate from $100 \cdot d$ training data.*

rather to the fact that the change-detection problem becomes intrinsically more challenging, as it occurs in the ideal case where $\phi_0$ is known (solid lines). When $\mathcal{L}$ is computed from an estimated $\widehat{\phi}_0$ (dashed and dotted lines), the problem becomes even more severe, and worsens when the number of training data does not grow with $d$ (dotted lines).

We interpret this result as a consequence of the intrinsic worsening of change detectability when $d$ increases. Moreover, detectability loss is not due to density-estimation issues, as it occurs also in the ideal case where $\phi_0$ is known (solid lines). In the more realistic cases where $\mathcal{L}$ is computed from an estimated $\widehat{\phi}_0$ (dashed and dotted lines), the problem become more severe, in particular when the number of training data does not increase with $d$ (dotted lines).

### 4.4.2 Gaussian mixtures

We now consider $\phi_0$ and $\phi_1$ as Gaussian mixtures, to prove that detectability loss occurs also when datastreams are generated/approximated by more general distribution models. Mimicking the proof of Theorem 4.1, we show that when $d$ increases and $\text{sKL}(\phi_0, \phi_1)$ is kept constant, the upper-bound of $\text{SNR}(\phi_0, \phi_1)$ decreases. To this purpose, it is enough to show that

$\underset{\mathbf{s}\sim\phi_0}{\text{var}}\left[\mathcal{L}(\mathbf{s})\right]$ increases with $d$.

The pdf of a mixture of $k$ Gaussians is

$$\begin{aligned}\phi_0(\mathbf{s}) &= \sum_{i=1}^{k} \lambda_{0,i}\mathcal{N}(\mu_{0,i},\Sigma_{0,i})(\mathbf{s}) = \\ &= \sum_{i=1}^{k} \frac{\lambda_{0,i}}{(2\pi)^{d/2}\text{det}(\Sigma_{0,i})^{1/2}} e^{-\frac{1}{2}(\mathbf{s}-\mu_{0,i})^T\Sigma_{0,i}^{-1}(\mathbf{s}-\mu_{0,i})},\end{aligned} \tag{4.18}$$

where $\lambda_{0,i} > 0$ is the weight of the $i$-th Gaussian $\mathcal{N}\left(\mu_{0,i},\Sigma_{0,i}\right)$. Unfortunately, the log-likelihood $\mathcal{L}(\mathbf{s})$ of a Gaussian mixture does not admit an expression similar to (4.13) and two approximations are typically used to avoid severe numerical issues when $d \gg 1$.

The first approximation consists in considering only the Gaussian of the mixture yielding the largest likelihood, as in [14] i.e.,

$$\begin{aligned}\mathcal{L}_u(\mathbf{s}) = -\frac{k\lambda_{0,i^*}}{2}\Big(&\log\left((2\pi)^d\text{det}(\Sigma_{0,i^*})\right) + \\ &+ (\mathbf{s}-\mu_{0,i^*})^T\Sigma_{0,i^*}^{-1}(\mathbf{s}-\mu_{0,i^*})\Big)\end{aligned} \tag{4.19}$$

where $i^*$ is defined as

$$i^* = \underset{i=1,\dots,k}{\text{argmax}}\left(\frac{\lambda_{0,i}}{(2\pi)^{d/2}\text{det}(\Sigma_{0,i})^{1/2}} e^{-\frac{1}{2}(\mathbf{s}-\mu_{0,i})^T\Sigma_{0,i}^{-1}(\mathbf{s}-\mu_{0,i})}\right).$$

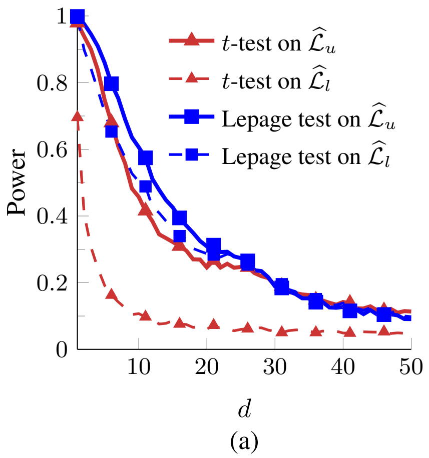
The second approximation we consider is:

$$\mathcal{L}_l(\mathbf{s}) = -\frac{1}{2}\sum_{i=1}^{k}\lambda_{0,i}\Big(\log\left((2\pi)^d\text{det}(\Sigma_{0,i})\right) + (\mathbf{s}-\mu_{0,i})^T\Sigma_{0,i}^{-1}(\mathbf{s}-\mu_{0,i})\Big), \tag{4.20}$$

that is a lower bound of $\mathcal{L}(\cdot)$ due to the Jensen inequality:

$$\begin{aligned}\mathcal{L}(\mathbf{s}) = \log\left(\sum_{i=1}^{k}\lambda_{0,i}\mathcal{N}(\mu_{0,i},\Sigma_{0,i})(\mathbf{s})\right) \geq \\ \geq \sum_{i=1}^{k}\lambda_{0,i}\log\left(\mathcal{N}(\mu_{0,i},\Sigma_{0,i})(\mathbf{s})\right) = \mathcal{L}_l(\mathbf{s}).\end{aligned}$$

We consider the same values of $d$ as in Section 4.4.1 and, for each of these, we generate 1000 datastreams each containing 500 data drawn from the Gaussian mixture $\phi_0$. We assume $k = 2$ and $\lambda_{0,1} = \lambda_{0,2} = 0.5$, while



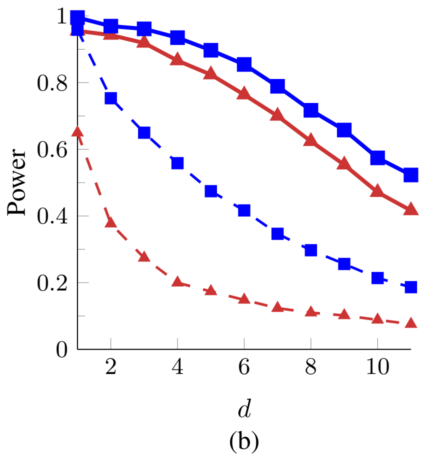


**Figure 4.2:** *Detectability loss on the Particle Dataset (a) and Wine Dataset (b), approximated by a mixture of 2 and 4 Gaussians, respectively, using both $\widehat{\mathcal{L}}_u$ (4.19) and $\widehat{\mathcal{L}}_l$ (4.20). The powers of both Lepage and $t$-test decay, confirming the detectability loss. The tests based on $\widehat{\mathcal{L}}_u$ outperform the corresponding ones based on $\widehat{\mathcal{L}}_l$ because this latter approximation yields a larger variance, as can be seen in Fig 4.1(b).*

the parameters $\mu_{0,1}$, $\mu_{0,2}$, $\Sigma_{0,1}$, $\Sigma_{0,2}$ are randomly generated. We then compute the sample variance of both $\mathcal{L}_u$ and $\mathcal{L}_l$ over each datastream and report their average in Figure 4.1(b). As in Section 4.4.1, we repeat this experiment estimating $\widehat{\phi}_0$ from a training set containing $200 \cdot d$ additional samples, then computing $\widehat{\mathcal{L}}_u$ and $\widehat{\mathcal{L}}_l$.

Figure 4.2(b) shows that the variances of $\mathcal{L}_u$ and $\mathcal{L}_l$ grow linearly with respect to $d$, as in the Gaussian case (4.14). This result indicates that also in this case, where datastreams are generated by correlated and bimodal distributions, detectability loss occurs, since the $\text{SNR}(\phi_0, \phi_1)$ decreases when $d$ increases. As in Section 4.4.1, we experienced the same trend when the log-likelihoods $\widehat{\mathcal{L}}_u$ and $\widehat{\mathcal{L}}_l$ are computed with respect to fitted models $\widehat{\phi}_0$. We further observe that $\mathcal{L}_l$ exhibits a much larger variance than $\mathcal{L}_u$, thus we expect this to achieve lower change-detection performance than $\mathcal{L}_u$. Probably, this is why $\mathcal{L}_u$ was used in [14] instead of $\mathcal{L}_l$.

### 4.4.3 Real-World Data

To investigate detectability loss in real-world datasets, we design a change-detection problem on the *Wine Quality Dataset* and the *MiniBooNE Particle Identification Dataset* from the UCI repository [76]. The Wine dataset has 12 dimensions: 11 corresponding to numerical results of laboratory analysis (such as density, Ph, residual sugar), and one corresponding to a final grade (from 0 to 10) for each different wine. We consider the vectors of laboratory analysis of all white wines having a grade above 6, resulting in a 11-dimensional dataset containing 3258 data. The Particle dataset contains numerical measurements from a physical experiment designed to distinguish electron from muon neutrinos. Each sample has 50-dimensions and we considered only data from muon class, yielding 93108 data.

Since in either datasets $\phi_0$ is completely unknown, we need to estimate it for both introducing changes having constant magnitude and computing the log-likelihood. We adopt Gaussian mixtures and estimate $k$ by 5-fold cross validation over the whole datasets, obtaining $k = 4$ and $k = 2$ for Wine and Particle dataset respectively.

We process each dataset as follows. Let us denote by $D$ the dataset dimension and for each value of $d = 1, \dots, D$ we consider only $d$ components of our dataset that are randomly selected. We then generate a $d$-dimensional training set of $200 \cdot d$ samples and a test set of 1000 samples (datastream), which are extracted by a bootstrap procedure without replacement. The second half of the datastream is perturbed by the change $\phi_0 \to \phi_1$, which is defined using CCM. Then, we estimate $\widehat{\phi}_0$ from the training set and we compute $\mathcal{T}(\widehat{W}_P, \widehat{W}_R)$, where $\widehat{W}_P$, $\widehat{W}_R$ are defined as in Section 4.4.1. This procedure is repeated 5000 times to numerically compute the test power. Note that the number of Gaussians in both $\widetilde{\phi}_0$ and $\widehat{\phi}_0$ is the value of $k$ estimated from whole $d$-dimensional dataset, and that $\widetilde{\phi}_0$ is by no means used for change-detection purposes.

Figure 4.2 reports the power of both Lepage and one-sided $t$-tests on the Particle dataset and Wine dataset, considering $\widehat{\mathcal{L}}_u$ (4.19) and $\widehat{\mathcal{L}}_l$ (4.20) as approximated expressions of the likelihoods. The power of both tests is monotonically decreasing, indicating an increasing difficulty in detecting a change among $\widehat{W}_P$ and $\widehat{W}_R$ when $d$ grows. This result is in agreement with the claim of Theorem 4.1 and the results shown in the previous sections. In contrast of Gaussian datastreams, the Lepage here turns to be more powerful than the $t$-test. This fact indicates that it is important to monitor also the dispersion of $\mathcal{L}(\cdot)$ in case of Gaussian mixture, where $\mathcal{L}(\cdot)$ can be multimodal. The decay of the power of the Lepage test also indicate that monitoring both expectation and dispersion of $\mathcal{L}(\cdot)$ does not prevent the detectability loss. Figure 4.2 indicates that $\widehat{\mathcal{L}}_u(\cdot)$ guarantees superior performance than $\widehat{\mathcal{L}}_l(\cdot)$ and this is a consequence of the lower variance of $\widehat{\mathcal{L}}_u(\cdot)$. This fact also underlines the importance of considering the variance of $\mathcal{L}(\cdot)$ in measures of change detectability, as in (4.4).

# CHAPTER 5

# Controlling the Change Magnitude

As shown in the previous chapter, assessing the performance of a change-detection algorithms requires datastreams affected by changes at known locations. Unfortunately, there are not many real-world datasets satisfying this requirement, and when the exact change-location is unknown, change-detection performance has to be interpreted by visualizing the analyzed datastream [17]. While visual inspection can indicate whether the algorithm was successful or not, it does not provide a statistically sound assessment, and it is definitively not a viable option when data dimension increases. Therefore, experiments are more conveniently performed on real-world datasets that have been manipulated to contain a change at a known location.

Most of the papers in the literature resort to introducing arbitrary shifts, scaling or swapping few components of the original dataset [14,77]. Changes can be also introduced by rotations and other linear transformations in multivariate data [55, 77], by mixing different datasets [78] or by swapping the classes of labeled data [15, 79]. Moreover, there are a few frameworks designed for generating datastreams containing changes [80–82]. Unfortunately, none of these approaches control the magnitude of the introduced changes, namely to which extent $\phi_1$ differs from $\phi_0$.

In this chapter we present "Controlling Change Magnitude" (CCM), a rigorous method to generate real-world datastreams affected by changes at known locations. We illustrate our design choices in Section 5.1, while CCM is presented in Section 5.2 together with two Theorems that guarantee its convergence. An extension to NP-CCM that does not require parametric assumptions on the pre-change distribution $\phi_0$ is presented in Section 5.3. We discuss the computational complexity of CCM and NP-CCM in Section 5.4, while experiments are shown in Section 5.5, and the MATLAB framework implementing CCM (including NP-CCM) is described in Section 5.6.

## 5.1 Settings

Let us consider a dataset $S_0$ containing stationary data, which we assume are i.i.d. realizations of a $d$-dimensional random vectors $\mathbf{S} \in \mathbb{R}^d$ having probability density function $\phi_0$. We manipulate $S_0$ to generate a datastream $\mathcal{S} = \{\mathrm{s}_t,\ t = 1, \dots, T\}$ affected by an abrupt change $\phi_0 \to \phi_1$ at location $t = \tau$ as in (2.1). We consider the change model introduced in Chapter 4, i.e., we assume that

$$\phi_1(\mathrm{s}) = \phi_0(Q\mathrm{s} + \mathbf{v}), \tag{5.1}$$

where $Q \in O(d) \subset \mathbb{R}^{d\times d}$ is an orthogonal matrix, i.e. $Q^TQ = QQ^T = I_d$, and $\mathbf{v}$ is a shift in the location of $\phi_0$.

Our goal is to identify suitable $Q$ and $\mathbf{v}$ such that the change $\phi_0 \to \phi_1$ features a specific magnitude, measured by the symmetric Kullback-Leibler divergence (sKL) between $\phi_0$ and $\phi_1$ (see Section 4.3.1). In practice, given $\phi_0$ and the desired magnitude $\kappa$, our goal is to compute $Q \in \mathbb{R}^{d\times d}$ and $\mathbf{v} \in \mathbb{R}^d$ such that

$$\text{sKL}\left(\phi_0, \phi_1\right) = \text{sKL}\left(\phi_0, \phi_0(Q\,\cdot + \mathbf{v})\right) = \kappa\,. \tag{5.2}$$

Choosing roto-translations in our change-model is particularly advantageous first of all because the datastream $\mathcal{S}$ can be *assembled* without having to synthetically *generate* any sample from $\phi_1$. In fact, stationary data (i.e. $\mathrm{s}_t,\ t < \tau$) can be randomly selected from $S_0$, while changed data can be obtained by directly roto-translating other randomly selected samples $\mathrm{s} \in S_0$, i.e. $\mathrm{s}_t = Q^T(\mathrm{s} - \mathbf{v})$ for $t > \tau$. Second, as discussed in Chapter 4, assuming $\phi_1(\cdot) = \phi_0(Q\ \cdot + \mathbf{v})$ is quite a general change model, which encompasses changes in the mean as well in the correlation among the data components. As we will show in Section 5.5, changes obtained by "component swap" or "adding an offset", which are typically encountered in the change-detection literature, correspond to particular roto-translations.

**Algorithm 5.1** CCM: an overview

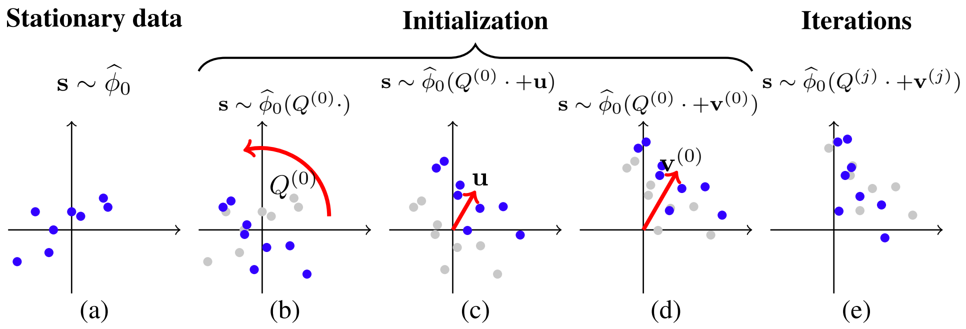
```
Input: $S_0$ the stationary dataset, $\kappa$ the desired change magnitude, $\varepsilon$ the tolerance for the change magnitude, $N$ the number of datastreams to generate, $T$ the length of the generated datastreams, $\tau$ the change location.
Output: $\mathcal{S}_1, \dots, \mathcal{S}_N$, generated datastreams.
1:  Fit a Gaussian mixture $\widetilde{\phi}_0$ to $S_0$
2:  for $i = 1, \dots, N$ do
    // Estimation of roto-translation parameters
3:      Run Algorithm 5.2 to initialize $Q^{(0)}$ and $\mathbf{v}^{(0)}$, yielding sKL$(\widetilde{\phi}_0, \widetilde{\phi}_0(Q^{(0)} \cdot +\mathbf{v}^{(0)})) > \kappa$
4:      Run Algorithm 5.3 to compute $Q$ and $\mathbf{v}$ yielding $|$sKL$(\widetilde{\phi}_0, \widetilde{\phi}_0(Q \cdot +\mathbf{v})) - \kappa| < \varepsilon$
    // Generation of the datastream
5:      for $t = 1, \dots, T$ do
6:          Randomly draw $\mathbf{w}$ from $S_0$.
7:          if $t < \tau$ then
8:              Set $\mathbf{s}_t = \mathbf{w}$.
9:          else
10:             Set $\mathbf{s}_t = Q^T(\mathbf{w} - \mathbf{v})$.
11:         end if
12:     end for
13:     Set $\mathcal{S}_i = \{\mathbf{s}_t, \, t = 1, \dots, T\}$.
14: end for
```

Third, the matrix $Q$ and the vector $\mathbf{v}$ themselves can be easily parametrized with respect to planes / angles of rotation, and to vector length and direction, respectively.

## 5.2 The CCM Method

In this section we first provide an overview of CCM and then describe in detail the algorithms and techniques used to generate datastreams affected by changes $\phi_0 \to \phi_1$ having a desired magnitude $\kappa$. An important issue in CCM is that to compute sKL$(\phi_0, \phi_1)$ we need $\phi_0$, which is typically unknown when manipulating real-world datasets. We solve this issue by replacing $\phi_0$ with an empirical estimate, and in particular we adopt a Gaussian mixture $\widehat{\phi}_0$ to approximate $\phi_0$.

Figure 5.1 illustrates the data manipulation performed by CCM, which is overviewed in Algorithm 5.1. At first, we start by fitting a Gaussian mixture $\widehat{\phi}_0$ to stationary data $S_0$ (line 1 and Section 5.2.1), which is used for computing our target function sKL$(\widetilde{\phi}_0, \widetilde{\phi}_0(Q \cdot +\mathbf{v}))$. The same Gaussian mixture $\widehat{\phi}_0$ is used to generate an arbitrary number $N$ of datastreams $\mathcal{S}$, each corresponding to a different roto-translation. The parametrization



Stationary data
Initialization
Iterations
$\mathbf{s} \sim \widehat{\phi}_0$
$\mathbf{s} \sim \widehat{\phi}_0(Q^{(0)}\cdot)$
$\mathbf{s} \sim \widehat{\phi}_0(Q^{(0)}\cdot + \mathbf{u})$
$\mathbf{s} \sim \widehat{\phi}_0(Q^{(0)}\cdot + \mathbf{v}^{(0)})$
$\mathbf{s} \sim \widehat{\phi}_0(Q^{(j)}\cdot + \mathbf{v}^{(j)})$
$Q^{(0)}$
$\mathbf{u}$
$\mathbf{v}^{(0)}$
(a)
(b)
(c)
(d)
(e)

**Figure 5.1:** *An illustration of the overall CCM procedure. Solid blue dots and shaded gray dots indicate the output and the input of each step, respectively. (a) At first, a Gaussian mixture $\widehat{\phi}_0$ is fitted on the stationary dataset $S_0$. Then, Algorithm 5.2 is used to initialize CCM by (b) randomly choosing a matrix $Q^{(0)}$ that defines an arbitrary rotation of $\mathbb{R}^d$, (c) computing random translation direction $\mathbf{u}$ and (d) doubling the length of the translation vector $\mathbf{v}^{(0)}$ until $\mathrm{sKL}(\phi_0, \phi_0(Q^{(0)}\cdot + \mathbf{v}^{(0)})) > \kappa$. The value of $Q$ and $\mathbf{v}$ are then adjusted by the bisection method in Algorithm 5.3 until the corresponding roto-translation yields a change having the desired magnitude.*

adopted to define the roto-translation is described in Section 5.2.3. For each datastream to be generated we initialize the values of $Q^{(0)}$ and $\mathbf{v}^{(0)}$ by Algorithm 5.2 (described in Section 5.2.4). Such initialization is meant to yield a change having a magnitude larger than the target value $\kappa$ and is illustrated in Figure 5.1(b-d). Then, the bisection method described in Algorithm 5.3 and in Section 5.2.5 is used to iteratively adjust $Q^{(0)}$ and $\mathbf{v}^{(0)}$ to achieve the desired change-magnitude $\kappa$ with an approximation error smaller than a given $\varepsilon > 0$. Finally, the computed $Q$ and $\mathbf{v}$ are used to generate the datastream $\mathcal{S}$ (lines 5-13) by randomly sampling data from $S_0$, and transforming those to be placed after the change location $\tau$. Since Algorithm 5.2 randomly initialize the change parameters, we can generate multiple datastreams $\mathcal{S}$ from the same $S_0$ by using the same $\widehat{\phi}_0$.

### 5.2.1 Fitting Pre-Change Distribution:

The pdf $\widetilde{\phi}_0$ of a Gaussian mixture is a convex combination of $k$ Gaussian functions, which we express as

$$\widetilde{\phi}_0(\mathbf{s}) = \sum_{i=1}^{k} \lambda_{0,i}\mathcal{N}(\mu_{0,i}, \Sigma_{0,i})(\mathbf{s}) = \sum_{i=1}^{k} \frac{\lambda_{0,i} \cdot e^{-\frac{1}{2}(\mathbf{s}-\mu_{0,i})^T \Sigma_{0,i}^{-1}(\mathbf{s}-\mu_{0,i})}}{(2\pi)^{d/2}\mathrm{det}(\Sigma_{0,i})^{1/2}}, \tag{5.3}$$

where $\sum_{i=1}^{k} \lambda_{0,i} = 1$ and $\lambda_{0,i} \geq 0$, $i \in 1, \ldots, k$ is the weight of the Gaussian having mean $\mu_{0,i}$ and covariance $\Sigma_{0,i}$.

We fit the Gaussian mixture to $S_0$ by traditional statistical techniques. In particular, we test different values of $k$ and select the best number of Gaussians by 5-fold cross-validation, by analyzing the distribution of the average likelihood over the test sets. Once having defined $k$, we estimate the parameters $\{\lambda_{0,i}\}$, $\{\mu_{0,i}\}$ and $\{\Sigma_{0,i}\}$ using the Expectation-Maximization (EM) algorithm [83].

We have adopted Gaussian mixtures since these are rather flexible models that can approximate skewed and multimodal distributions. Most importantly, $\text{sKL}(\widehat{\phi}_0, \widehat{\phi}_0(Q \cdot + \mathbf{v}))$ is always defined and finite when $\widehat{\phi}_0$ is a Gaussian mixture, and there exist approximated and fast expressions for computing the log-likelihood w.r.t. a Gaussian mixture, which we describe next.

### 5.2.2 Computing sKL

When running CCM, we have to repeatedly compute $\text{sKL}(\widehat{\phi}_0, \widetilde{\phi}_1)$, where both $\widehat{\phi}_0$ and $\widetilde{\phi}_1(\cdot) = \widehat{\phi}_0(Q \ \cdot + \mathbf{v})$ are Gaussian mixtures. Since there are no analytical expressions to compute the symmetric Kullback-Leibler divergence between two Gaussian mixtures, we approximate the integrals in (4.3) via Montecarlo simulations.

More precisely, we synthetically generate two sets $\widetilde{S}_0$ and $\widetilde{S}_1$ containing a sufficient number of i.i.d. realizations drawn from $\widehat{\phi}_0$ and $\widetilde{\phi}_1$, respectively, and compute the following approximation:

$$\text{sKL}(\widetilde{\phi}_0, \widetilde{\phi}_1) \approx \frac{1}{\#\widetilde{S}_0} \sum_{\mathbf{s} \in \widetilde{S}_0} \log \frac{\widetilde{\phi}_0(\mathbf{s})}{\widetilde{\phi}_1(\mathbf{s})} + \frac{1}{\#\widetilde{S}_1} \sum_{\mathbf{s} \in \widetilde{S}_1} \log \frac{\widetilde{\phi}_1(\mathbf{s})}{\widetilde{\phi}_0(\mathbf{s})}, \tag{5.4}$$

where $\#\widetilde{S}_0$ and $\#\widetilde{S}_1$ denote the cardinality of $\widetilde{S}_0$ and $\widetilde{S}_1$, respectively.

Since the computation of $\widetilde{\phi}_0(\mathbf{s})$ (resp. of $\widetilde{\phi}_1(\mathbf{s})$) and of its logarithm in (5.4) might rise severe numerical issues when $d$ is large, we further approximate the likelihood of a Gaussian mixture by considering only the Gaussian of the mixture yielding the largest likelihood, as in Section 4.4.2. In particular, this yield to the following upper bound of $\log \phi_0(\mathbf{s})$

$$\begin{aligned} \psi_0(\mathbf{s}) = -\frac{1}{2}\Big[ & \log\left((2\pi)^d \det(\Sigma_{0,i^*})\right) + \\ & + (\mathbf{s} - \mu_{0,i^*})^T \Sigma_{0,i^*}^{-1} (\mathbf{s} - \mu_{0,i^*}) \Big] + \log(k\lambda_{0,i^*}), \end{aligned} \tag{5.5}$$

where $i^*$ is defined as

$$i^* = \underset{i=1,\dots,k}{\mathrm{argmax}} \frac{\lambda_{0,i} \cdot e^{-\frac{1}{2}(\mathbf{s}-\mu_{0,i})^T \Sigma_{0,i}^{-1}(\mathbf{s}-\mu_{0,i})}}{(2\pi)^{d/2}\mathrm{det}(\Sigma_{0,i})^{1/2}}. \tag{5.6}$$

The upperbound of $\log \widetilde{\phi}_1(\mathbf{s})$, namely $\psi_1(\mathbf{s})$, is similarly defined. Then, the Montecarlo estimate of $\mathrm{sKL}(\widetilde{\phi}_0, \widetilde{\phi}_1)$ is obtained by computing

$$\mathrm{sKL}(\widetilde{\phi}_0, \widetilde{\phi}_1) \approx \frac{1}{\#\widetilde{S}_0} \sum_{\mathbf{s}\in\widetilde{S}_0} (\psi_0(\mathbf{s}) - \psi_1(\mathbf{s})) + \frac{1}{\#\widetilde{S}_1} \sum_{\mathbf{s}\in\widetilde{S}_1} (\psi_1(\mathbf{s}) - \psi_0(\mathbf{s})). \tag{5.7}$$

### 5.2.3 Parametrization

To ease calculations, we express $Q$ with respect to its planes and angles of rotation. We store $m := \lfloor d/2 \rfloor$ angles $\theta_1, \dots, \theta_m$ in a vector $\boldsymbol{\theta}$, and define the matrix $G(\boldsymbol{\theta}) \in \mathbb{R}^{d\times d}$ as

$$\begin{aligned} G(\boldsymbol{\theta}) &= \begin{bmatrix} R(\theta_1) & \cdots & 0 & 0 \\ \vdots & \ddots & \vdots & \vdots \\ 0 & \cdots & R(\theta_m) & 0 \\ 0 & \cdots & 0 & 1 \end{bmatrix}, \\ R(\theta_i) &= \begin{bmatrix} \cos\theta_i & -\sin\theta_i \\ \sin\theta_i & \cos\theta_i \end{bmatrix}. \end{aligned} \tag{5.8}$$

where $R(\theta_i)$ defines a counterclockwise rotation of angle $\theta_i$ around the origin in $\mathbb{R}^2$. When $d$ is even, the last column and the last row of $G(\boldsymbol{\theta})$ are dropped. The matrix $G(\boldsymbol{\theta})$ defines a rotation in $\mathbb{R}^d$ whose planes of rotation are generated by the coordinate axes: rotation matrices $Q$ referring to different coordinate axes can be obtained by multiplying $G(\boldsymbol{\theta})$ against an orthogonal matrix $P \in \mathbb{R}^{d\times d}$. Thus, we parametrize the rotation matrix $Q$ as

$$Q(\boldsymbol{\theta}, P) = PG(\boldsymbol{\theta})P^T. \tag{5.9}$$

The translation vector $\mathbf{v}$ is parameterized by its norm and direction:

$$\mathbf{v}(\rho, \mathbf{u}) = \rho\mathbf{u}, \quad \text{where } \|\mathbf{u}\|_2 = 1,\ \rho = \|\mathbf{v}\|_2. \tag{5.10}$$

### 5.2.4 Initialization

CCM initialization is described in Algorithm 5.2 and it is meant to set $Q^{(0)}$ and $\mathbf{v}^{(0)}$ yielding $\mathrm{sKL}(\phi_0, \phi_0(Q^{(0)} \cdot +\mathbf{v}^{(0)})) > \kappa$. This is a necessary condition for the bisection procedure described in Section 5.2.5 to

**Algorithm 5.2** CCM: initialization

```
Input: $\widetilde{\phi}_0$ and $\kappa$, the target value of sKL($\widetilde{\phi}_0, \widetilde{\phi}_1$).
Output: Initial roto-translation parameters $\boldsymbol{\theta}^{(0)}$, $P$, $\rho^{(0)}$, $\mathbf{u}$.
1: Set $\rho^{(0)} = 1$.
2: Randomly generate $m$ angles $\boldsymbol{\theta}^{(0)}$ in $[-\pi/2, \pi/2]^m$ and a unitary vector $\mathbf{u}$.
3: Generate a matrix $A \in \mathbb{R}^{d \times d}$ having Gaussian entries.
4: Set $P$ as the orthogonal matrix of the QR decomposition of $A$.
5: Set $Q^{(0)}(\boldsymbol{\theta}^{(0)}, P) = PG(\boldsymbol{\theta}^{(0)})P^T$ and $\mathbf{v}(\rho^{(0)}, \mathbf{u}) = \rho^{(0)}\mathbf{u}$.
6: repeat
7:     Compute $s^{(0)} = \text{sKL}(\widetilde{\phi}_0, \widetilde{\phi}_1)$, where $\widetilde{\phi}_1 = \widetilde{\phi}_0(Q^{(0)} \cdot + \mathbf{v}^{(0)})$.
8:     $\rho^{(0)} = 2\rho^{(0)}$.
9: until $s^{(0)} > \kappa$
```

work. Given a stationary dataset (Figure 5.1(a)), we randomly define the angles of rotation $\boldsymbol{\theta}^{(0)}$ in $[-\pi/2, \pi/2]^m$ (line 2) and randomly select an orthogonal matrix $P$ (lines 3-4). The rotations angles are drawn from a multivariate uniform distribution, and the matrix $P$ is conveniently defined by computing the QR decomposition [84] of a matrix $A$ having random Gaussian entries. The rotation matrix $Q^{(0)}$ is defined according to (5.9) as $Q^{(0)} = Q^{(0)}(\boldsymbol{\theta}^{(0)}, P) = PG(\boldsymbol{\theta}^{(0)})P^T$ (line 5), being $G(\boldsymbol{\theta})$ as in (5.8). This is a very general procedure, that can generate any rotation preserving the orientation of the coordinate system. The matrix $Q^{(0)}$ defines the rotation around the origin of the dataset, as show in Figure 5.1(b).

Then, we randomly generate a vector $\mathbf{u}$ having unitary norm (line 2) as described in [85, Algorithm 11]. The vector $\mathbf{u}$ actually defines the translation direction, as illustrated in Figure 5.1(c). If the corresponding $\text{sKL}(\widetilde{\phi}_0, \widetilde{\phi}_0(Q^{(0)} \cdot + \mathbf{u}))$ is larger than $\kappa$, the algorithm terminates and returns $Q^{(0)}$ and $\mathbf{v}^{(0)} = \mathbf{u}$. Otherwise, the length of $\mathbf{v}^{(0)}$ is increased by doubling $\rho^{(0)}$ (Figure 5.1(d)), which controls the extent of the shift in the data (line 8). This procedure is repeated until the above condition is satisfied. Convergence of Algorithm 5.2 is guaranteed by the following theorem.

**Theorem 5.1.** *Let $\widetilde{\phi}_0$ be a Gaussian mixture. Then, for any $\kappa > 0$ and tolerance $\varepsilon > 0$, Algorithm 5.2 converges in a finite number of iterations.*

*Proof.* It is enough to show that $\text{sKL}(\widetilde{\phi}_0, \widetilde{\phi}_0(Q^{(0)} \cdot + \mathbf{v})) \to \infty$ as $\|\mathbf{v}\|_2 \to \infty$, or that it admits a lower bound that diverges as $\|\mathbf{v}\|_2 \to \infty$. We here pursue the latter approach and define the lower bound as follows

$$\begin{aligned}
&\mathrm{sKL}(\widetilde{\phi}_0, \widetilde{\phi}_0(Q^{(0)} \cdot + \mathbf{v})) \geq \mathrm{KL}(\widetilde{\phi}_0, \widetilde{\phi}_0(Q^{(0)} \cdot + \mathbf{v})) \\
&= \int_{\mathbb{R}^d} \widetilde{\phi}_0(\mathbf{s}) \log \left( \frac{\widetilde{\phi}_0(\mathbf{s})}{\widetilde{\phi}_0(Q^{(0)}\mathbf{s} + \mathbf{v})} \right) d\mathbf{s} \\
&= \int_{\mathbb{R}^d} \widetilde{\phi}_0(\mathbf{s}) \log(\widetilde{\phi}_0(\mathbf{s})) d\mathbf{s} + \\
&\qquad - \int_{\mathbb{R}^d} \widetilde{\phi}_0(\mathbf{s}) \log(\widetilde{\phi}_0(Q^{(0)}\mathbf{s} + \mathbf{v})) d\mathbf{s} \\
&\geq \int_{\mathbb{R}^d} \widetilde{\phi}_0(\mathbf{s}) \log(\widetilde{\phi}_0(\mathbf{s})) d\mathbf{s} + \\
&\qquad - \log \left( \int_{\mathbb{R}^d} \widetilde{\phi}_0(\mathbf{s}) \widetilde{\phi}_0(Q^{(0)}\mathbf{s} + \mathbf{v}) d\mathbf{s} \right) .
\end{aligned} \tag{5.11}$$

which follows from the fact that the KL is nonnegative and from the Jensen's inequality.

The first term $\int_{\mathbb{R}^d} \widetilde{\phi}_0(\mathbf{s}) \log(\widetilde{\phi}_0(\mathbf{s})) d\mathbf{s}$ is finite, since the following upper bound[1]

$$|\log(\widetilde{\phi}_0(\mathbf{s}))| \leq c_1 + c_2 \|\mathbf{s}\|_2^2, \qquad \forall \mathbf{s} \text{ s.t. } \|\mathbf{s}\|_2 > r. \tag{5.12}$$

holds for suitable constants $c_1, c_2, r > 0$.

Then, we have to prove that the second term in the last row of (5.11) diverges, thus that the integral

$$\int_{\mathbb{R}^d} \widetilde{\phi}_0(\mathbf{s}) \widetilde{\phi}_0(Q^{(0)}\mathbf{s} + \mathbf{v}) d\mathbf{s} \tag{5.13}$$

converges to 0 as $\|\mathbf{v}\|_2 \to \infty$. To this purpose, we split the integral in in the sum of two integrals and show that each of these converges to 0.

Let us fix $\varepsilon > 0$. Since $\widetilde{\phi}_0$ is integrable, there exists $R_1 > 0$ such that

$$\int_{\{\mathbf{s}\,:\,\|\mathbf{s}\|_2 > R_1\}} \widetilde{\phi}_0(\mathbf{s}) d\mathbf{s} < \varepsilon. \tag{5.14}$$

Moreover, $\widetilde{\phi}(\mathbf{s}) \to 0$ as $\|\mathbf{s}\|_2 \to \infty$, therefore there exists $R_2 > 0$ such that

$$\widetilde{\phi}_0(\mathbf{s}) < \varepsilon \quad \forall \mathbf{s} \,:\, \|\mathbf{s}\|_2 > R_2. \tag{5.15}$$

[1]This bound is trivial for Gaussian pdfs and follows from basic algebra in case of Gaussian Mixtures.

Let us now define $R = 2\max(R_1, R_2)$ and pick $\mathbf{v}$ such that $\|\mathbf{v}\|_2 > R$. We split (5.13) as

$$\begin{aligned}\int_{\mathbb{R}^d} \widetilde{\phi}_0(\mathbf{s})\widetilde{\phi}_0(Q^{(0)}\mathbf{s}+\mathbf{v})d\mathbf{s} = &\int_{\{\mathbf{s}\,:\,\|\mathbf{s}\|_2>R\}} \widetilde{\phi}_0(\mathbf{s})\widetilde{\phi}_0(Q^{(0)}\mathbf{s}+\mathbf{v})d\mathbf{s} + \\ &+ \int_{\{\mathbf{s}\,:\,\|\mathbf{s}\|_2\le R\}} \widetilde{\phi}_0(\mathbf{s})\widetilde{\phi}_0(Q^{(0)}\mathbf{s}+\mathbf{v})d\mathbf{s}.\end{aligned} \tag{5.16}$$

Since $\widetilde{\phi}_0$ is bounded over $\mathbb{R}^d$, the first integral in the sum can be upper bounded as

$$\begin{aligned}\int_{\{\mathbf{s}\,:\,\|\mathbf{s}\|_2>R\}} \widetilde{\phi}_0(\mathbf{s})\widetilde{\phi}_0(Q^{(0)}\mathbf{s}+\mathbf{v})d\mathbf{s} \le &\int_{\{\mathbf{s}\,:\,\|\mathbf{s}\|_2>R\}} \widetilde{\phi}_0(\mathbf{s})\widetilde{\phi}_0(Q^{(0)}\mathbf{s}+\mathbf{v})d\mathbf{s}\,\cdot \\ &\cdot \sup_{\mathbf{s}\in\mathbb{R}^d} \widetilde{\phi}_0(\mathbf{s}) \le \\ &\le \varepsilon \cdot \sup_{\mathbf{s}\in\mathbb{R}^d} \widetilde{\phi}_0(\mathbf{s}),\end{aligned} \tag{5.17}$$

where the last inequality holds for (5.14).

To upper bound the second integral in (5.16), we note that $\forall\, \mathbf{s}$ such that $\|\mathbf{s}\|_2 \le R$, the inverse triangle inequality and the fact that $\|Q^{(0)}\mathbf{s}\|_2 = \|\mathbf{s}\|_2$ imply that

$$\begin{aligned}\|Q^{(0)}\mathbf{s}+\mathbf{v}\|_2 \ge \|\mathbf{v}\|_2 - \|Q^{(0)}\mathbf{s}\|_2 = \|\mathbf{v}\|_2 - \|\mathbf{s}\|_2 \ge \\ \ge 2R - R = R,\end{aligned} \tag{5.18}$$

where the last inequality holds since we pick $\|\mathbf{v}\|_2 > 2R$. We can derive the upper bound

$$\int_{\{\mathbf{s}\,:\,\|\mathbf{s}\|_2\le R\}} \widetilde{\phi}_0(\mathbf{s})\widetilde{\phi}_0(Q^{(0)}\mathbf{s}+\mathbf{v})d\mathbf{s} \le \varepsilon \int_{R^d} \widetilde{\phi}_0(\mathbf{s})d\mathbf{s} \le \varepsilon, \tag{5.19}$$

where the first inequality follows from (5.15) and (5.18). Combining (5.17) and (5.19) we obtain

$$\int_{\mathbb{R}^d} \widetilde{\phi}_0(\mathbf{s})\widetilde{\phi}_0(Q^{(0)}\mathbf{s}+\mathbf{v})d\mathbf{s} \le \left(1 + \sup_{\mathbf{s}\in\mathbb{R}^d} \widetilde{\phi}_0(\mathbf{s})\right) \cdot \varepsilon, \tag{5.20}$$

that completes the proof.

□

**Algorithm 5.3** CCM: iterations to compute the roto-translation yielding the desired change magnitude

```
Input: $\boldsymbol{\theta}^{(0)}$, $P$, $\rho^{(0)}$, $\mathbf{u}$ from Algorithm 5.2, $\widetilde{\phi}_0$, $\kappa$, and the tolerance $\varepsilon$.
Output: $Q$ and $\mathbf{v}$ defining the roto-translation yielding sKL$(\widetilde{\phi}_0, \widetilde{\phi}_0(Q \cdot + \mathbf{v})) \approx \kappa$.
 1: Set the lower bounds $\boldsymbol{\theta}_l^{(1)} = 0$, $\rho_l^{(1)} = 0$.
 2: Set the upper bounds $\boldsymbol{\theta}_u^{(1)} = \boldsymbol{\theta}^{(0)}$, $\rho_u^{(1)} = \rho^{(0)}$.
 3: Set $j = 1$.
 4: repeat
 5:     Compute $\boldsymbol{\theta}^{(j)} = (\boldsymbol{\theta}_l^{(j)} + \boldsymbol{\theta}_u^{(j)})/2$, and $Q^{(j)}(\boldsymbol{\theta}^{(j)}, P)$ as in (5.9).
 6:     Compute $\rho^{(j)} = (\rho_l^{(j)} + \rho_u^{(j)})/2$, and $\mathbf{v}^{(j)}(\rho^{(j)}, \mathbf{u})$ as in (5.10).
 7:     Compute $s^{(j)} = $ sKL$(\widetilde{\phi}_0, \widetilde{\phi}_1^{(j)})$, where $\phi_1^{(j)}(\cdot) = \widetilde{\phi}_0(Q^{(j)} \cdot + \mathbf{v}^{(j)})$.
 8:     if $s^{(j)} < \kappa$ then
 9:         $\boldsymbol{\theta}_l^{(j+1)} = \boldsymbol{\theta}^{(j)}$, $\rho_l^{(j+1)} = \rho^{(j)}$.
10:     else
11:         $\boldsymbol{\theta}_u^{(j+1)} = \boldsymbol{\theta}^{(j)}$, $\rho_u^{(j+1)} = \rho^{(j)}$.
12:     end if
13:     $j = j + 1$.
14: until $|s^{(j)} - \kappa| < \varepsilon$
```

### 5.2.5 Iterations

The initial values of $Q^{(0)}$ and $\mathbf{v}^{(0)}$ are iteratively adjusted in Algorithm 5.3 until the corresponding change achieves the target magnitude $\kappa$ with the desired tolerance $\varepsilon$ (Figure 5.1(e)). In particular Algorithm 5.3 implements a bisection procedure that starts from $Q^{(0)} = PG(\boldsymbol{\theta}^{(0)})P^T$ and $\mathbf{v}^{(0)} = \rho^{(0)}\mathbf{u}$, and that iteratively adjusts the rotation angles $\boldsymbol{\theta}$ and the translation extent $\rho$. The rotation planes defined by $P$ and the translation direction $\mathbf{u}$ are instead kept fixed.

In what follows, we use the subscripts $l$ and $u$ to denote the parameters yielding change magnitudes that are smaller and larger than $\kappa$, respectively. Initially, we set both $\boldsymbol{\theta}_l$ and $\rho_l$ to 0 (line 1), while $\boldsymbol{\theta}_u = \boldsymbol{\theta}^{(0)}$ and $\rho_u = \rho^{(0)}$ (line 2). As typical in bisection methods, we set $\boldsymbol{\theta}^{(j)}$ to the average of $\boldsymbol{\theta}_l^{(j)}$ and $\boldsymbol{\theta}_u^{(j)}$ (line 5), and $\rho^{(j)}$ to the average of $\rho_l^{(j)}$ and $\rho_u^{(j)}$ (line 6).

We denote by $s^{(j)}$ the change magnitude induced by a roto-tranlsation parametrized by $\boldsymbol{\theta}^{(j)}$ and $\rho^{(j)}$ (line 7), which we compute as described in Section 5.2.2. When $s^{(j)} < \kappa$, we replace both $\boldsymbol{\theta}_l$ and $\rho_l$ with $\boldsymbol{\theta}^{(j)}$ and $\rho^{(j)}$, respectively (line 9), otherwise we similarly update $\boldsymbol{\theta}_u$ and $\rho_u$ (line 11).

These steps are iterated until the sKL achieves the target value $\kappa$ up to the desired tolerance $\varepsilon$. Convergence of Algorithm 5.3 is guaranteed by the following theorem.

**Theorem 5.2.** *Let $\widetilde{\phi}_0$ be a Gaussian mixture. Then, for any $\kappa > 0$ and tolerance $\varepsilon > 0$, Algorithm 5.3 converges in a finite number of iterations.*

*Proof.* The thesis follows from the Intermediate Value Theorem [86] (Theorem 4.23) applied to the function used in the bisection procedure, i.e.

$$skl(a) := \text{sKL}(\widetilde{\phi}_0, \widetilde{\phi}_0(Q(\boldsymbol{\theta}(a), P) \cdot + \rho(a)\mathbf{u})), \tag{5.21}$$

where $\boldsymbol{\theta}(s) = (1-s)\boldsymbol{\theta}_l^{(1)} + s\boldsymbol{\theta}_u^{(1)}$ and $\rho(s) = (1-s)\rho_l^{(1)} + s\rho_u^{(1)}$ are convex combinations of the initialization parameters (defined in Algorithm 5.3, lines 1-2). We observe that $skl(0) = 0 < \kappa$ since $\boldsymbol{\theta}_l^{(1)} = 0$ and $\rho_l^{(1)} = 0$, thus the corresponding roto-translation is the identity transformation and $\text{sKL}(\widehat{\phi}_0, \widehat{\phi}_0) = 0$. Moreover, $skl(1) > \kappa$, thus it is enough to show that $skl(\cdot)$ is continuous in $[0,1]$ to prove that the bisection method converges (see [87], Theorem 2.1) to a value of $\bar{s}$ such that $skl(\bar{s}) = \kappa$. Thus, $\boldsymbol{\theta}(\bar{s})$ and $\rho(\bar{s})$ yield $\text{sKL}(\widetilde{\phi}_0, \widetilde{\phi}_1)$ equal to $\kappa$.

To show that $skl(\cdot)$ in (5.21) is continuous, we show that both summands of sKL are continuous, which can be proved by defining

$$kl(\cdot) = \int_{\mathbb{R}^d} g(\cdot, \mathbf{s}) d\mathbf{s} \tag{5.22}$$

where

$$g(a, \mathbf{s}) := \widetilde{\phi}_0(\mathbf{s}) \log\left( \frac{\widetilde{\phi}_0(\mathbf{s})}{\widetilde{\phi}_0(Q(\boldsymbol{\theta}(a), P)\mathbf{s} + \mathbf{v}(\rho(a), \mathbf{u}))} \right). \tag{5.23}$$

The function $g(\cdot,\cdot)$ in (5.22) is continuous in $[0,1] \times \mathbb{R}^d$, thus $g(\cdot, \mathbf{s})$ is continuous in $[0,1]$. Then, to prove that $kl(\cdot)$ is also continuous, we leverage Lemma 16.1 in [88] and prove that $|g(a,\mathbf{s})|$ admits a dominant integrable function that does not depend on $s$. To this purpose, we exploit (5.12) and, for a sufficiently large $r$,

$$\begin{aligned}
|g(a, \mathbf{s})| &= \widetilde{\phi}_0(\mathbf{s}) \Big| \log\left(\widetilde{\phi}_0(\mathbf{s})\right) + \\
&\quad - \log\left(\widetilde{\phi}_0(Q(\boldsymbol{\theta}(a), P)\mathbf{s} + \mathbf{v}(\rho(a), \mathbf{u}))\right)\Big| \\
&\leq \widetilde{\phi}_0(\mathbf{s}) \Big(c_1 + c_2\|\mathbf{s}\|_2^2 + c_1 + c_2\big\|Q(\boldsymbol{\theta}(a), P)\mathbf{s} + \\
&\quad + \mathbf{v}(\rho(a), \mathbf{u})\big\|_2^2\Big).
\end{aligned} \tag{5.24}$$

The exponential decay of Gaussian mixture implies that

$$\sqrt{\widetilde{\phi}_0(\mathrm{s})}\Bigg(2c_1 + c_2\Big(\|\mathrm{s}\|_2^2 + \\ + \|Q(s\boldsymbol{\theta}^{(0)}, P)\mathrm{s} + \mathbf{v}(s\rho^{(0)}, \mathbf{u})\|_2^2\Big)\Bigg) < c$$

for some $c > 0$. Thus, the function that dominates $|g(\cdot, \mathrm{s})|$ for $\|\mathrm{s}\|_2 > r$ is $c\sqrt{\widetilde{\phi}_0(\mathrm{s})}$ which obviously belongs[2] to $L^1(\mathbb{R}^d)$ . □

## 5.3 Non Parametric CCM (NP-CCM)

The main assumption in CCM is that $\phi_0$ can be well approximated by a Gaussian mixture $\widehat{\phi}_0$. Although this is quite a reasonable assumption, there are cases where, e.g. due to the lack of training data, it is not possible to accurately approximate $S_0$ using a Gaussian mixture. However, CCM uses Gaussian mixture only in Algorithms 5.2 and 5.3 to estimate the symmetric Kullback-Leibler divergence, and we can relax this technical assumption if an alternative method for computing $\mathrm{sKL}(\widetilde{\phi}_0, \widetilde{\phi}_1)$ is provided. The rationale behind Non Parametric CCM (NP-CCM) is that the parameters $Q$ and v yielding the desired value of $\mathrm{sKL}(\phi_0, \phi_0(Q \cdot + \mathrm{v}))$ can be directly computed by comparing stationary and transformed data, avoiding fitting any distribution $\widehat{\phi}_0$.

The problem of estimating the Kullback-Leibler divergence from two populations have been recently investigated in the literature [89–92], and we adopt the method in [89] that is based on the computation of the $k$-nearest neighbor ($k$-nn) distance (as recommended in [89] we use 1-nn distance). In particular $\mathrm{KL}(\phi_0, \phi_1)$ can be estimated as

$$\widehat{\mathrm{KL}}(B_0, B_1) = \frac{d}{n_0}\sum_{\mathrm{s}\in B_0} \log\frac{r_0(\mathrm{s})}{r_1(\mathrm{s})} + \log\frac{n_1}{n_0 - 1}, \tag{5.25}$$

where $B_0$ and $B_1$ are two sets drawn from $\phi_0$ and $\phi_1$ that contain $n_0$ and $n_1$ samples, respectively. In (5.25), $r_0(\mathrm{s})$ denotes the distance between s and its nearest neighbor in $B_0\backslash\{\mathrm{s}\}$ and similarly $r_1(\mathrm{s})$ in $B_1$. It was shown [89] that $\widehat{\mathrm{KL}}(B_0, B_1)$ converges almost surely to $\mathrm{KL}(\phi_0, \phi_1)$ as $n_0, n_1 \to \infty$. Moreover, we have experienced that a better estimate can be obtained by setting $n_0 < n_1$.

[2]There is no need to find a dominant function for $\|\mathrm{s}\|_2 \leq r$ since there $g(\cdot, \mathrm{s})$ is bounded.

**Algorithm 5.4** NP-CCM: estimation of the symmetric Kullback-Leibler without fitting $\widehat{\phi_0}$

```
Input: Dataset $S_0$ sampled accordingly to $\phi_0$, roto-translation parameters $Q$ and $\mathbf{v}$.
Output: Estimate of $\text{sKL}(\phi_0, \phi_0(Q \cdot +\mathbf{v}))$.
1: Randomly extracts $B_0 \subset S_0$ and $B_1 \subset \mathcal{S}$ such that $B_0 \cap B_1 = \emptyset$ and $\#B_0 = n_0$, $\#B_1 = n_1$.
2: Apply the roto-translation given by $Q$ and $\mathbf{v}$ to each element of $B_1$, obtaining $\widetilde{B}_1$.
3: Estimate $\widehat{\text{KL}}(\phi_0, \phi_0(Q \cdot +\mathbf{v})) = \widehat{\text{KL}}(B_0, \widetilde{B}_1)$ as in (5.25).
4: Repeat 1-3 to estimate $\widehat{\text{KL}}(\phi_0(Q \cdot +\mathbf{v}), \phi_0) = \widehat{\text{KL}}(\widetilde{B}_1, B_0)$.
5: Set the estimate of $\text{sKL}(\phi_0, \phi_0(Q\cdot+\mathbf{v}))$ to $\widehat{\text{KL}}(\phi_0, \phi_0(Q\cdot+\mathbf{v})) + \widehat{\text{KL}}(\phi_0(Q\cdot+\mathbf{v}), \phi_0)$.
```

Thus, NP-CCM develops as CCM, provided that the change-magnitude is estimated by approximating sKL as in (5.25) rather than (5.7). In Algorithm 5.4 we precisely describe how to estimate $\text{sKL}(\phi_0, \phi_0(Q \cdot +\mathbf{v}))$ from a stationary dataset $S_0$, provided $Q$ and $\mathbf{v}$. At first, we randomly extract two subsets $B_0$ and $B_1$ from $S_0$ (line 1) and we roto-translate samples in $B_1$ using $Q$ and $\mathbf{v}$ (line 2). These sets are used to estimate the Kullback-Leibler divergence between $\phi_0$ and $\phi_0(Q \cdot +\mathbf{v})$ using (5.25) (line 3). The same procedure is repeated to estimate the Kullback-Leibler divergence between $\phi_0(Q \cdot +\mathbf{v})$ and $\phi_0$ (line 4); then $\text{sKL}(\phi_0, \phi_0(Q \cdot +\mathbf{v}))$ is obtained by summing these two estimates (line 5).

Thus, the differences between CCP and NP-CCM are in Algorithm 5.2 line 7, and Algorithm 5.3 line 7, where sKL is estimated by using Algorithm 5.4. Obviously, in NP-CCM there is also no need to estimate the Gaussian mixture $\widehat{\phi_0}$ in Algorithm 5.1 line 1. We comment that removing the Gaussian-mixture assumption implies that Theorems 5.1 and 5.2 do not hold, thus the convergence of Algorithms 5.2 and 5.3 cannot be guaranteed. In fact, both theorems rely on the assumption that $\widehat{\phi_0}$ is a Gaussian mixture. When using (5.25), we need additional assumptions on $\phi_0$ to prove CCM convergence. More precisely, Theorem 5.1 holds if we assume that $\phi_0$ is bounded over $\mathbb{R}^d$ and vanish at infinity. In fact, in this case we can prove that the integral in (5.13) converges to 0 as $\|\mathbf{v}\|_2 \to \infty$. In case of Theorem 5.2, we have to assume the existence of a function $h \in L^1(\mathbb{R}^d)$ such that for every rotation matrix $Q$ and translation vector $\mathbf{v}$, the following condition holds:

$$\left| \phi_0(\mathbf{s}) \log \frac{\phi_0(\mathbf{s})}{\phi_0(Q\mathbf{s} + \mathbf{v})} \right| \leq h(\mathbf{s}). \tag{5.26}$$

In this case the $g(a, \mathbf{x})$ in (5.23) is dominated by $h(\mathbf{s})$ and the function $kl(\cdot)$ in (5.22) is continuous. The assumption in (5.26) holds when the data-generating distribution is a Gaussian mixture, but it is difficult to verify for a more general class of distributions.

## 5.4 Computational Complexity

The computational complexity of CCM is dominated by the cost of estimating the symmetric Kullback-Leibler divergence, that has to be performed at each iteration of Algorithms 5.2 and 5.3. In fact, fitting a Gaussian mixture or performing the QR decomposition (which are invoked in the initial steps of CCM), are not iterated in CCM. Estimating $\text{sKL}(\widehat{\phi}_0, \widetilde{\phi}_1)$ through (5.7) involves the computation of $\psi_0(\mathbf{s})$ and $\psi_1(\mathbf{s})$, which requires computing the Mahalanobis distance between $\mathbf{s}$ and the mean vector of each Gaussian in $\widehat{\phi}_0$ for each of the $k$ numerators in (5.6). Therefore, computing $\psi_0(\mathbf{s})$ and $\psi_1(\mathbf{s})$ requires $O(d^2k)$ operations that have to be computed for each $\mathbf{s} \in \widetilde{S}_0 \cup \widetilde{S}_1$. This yields the computational complexity of (5.7) to $O(d^2kn)$ for each iteration of Algorithms 5.2 and 5.3, being $2n$ the number of samples in $\widetilde{S}_0 \cup \widetilde{S}_1$.

The computational complexity of NP-CCM is determined by the cost of estimating sKL through (5.25). When the 1-nn distances $r_0$ and $r_1$ in (5.25) are computed by straightforward comparisons between all the elements of $\mathcal{S}_0$ and $\mathcal{S}_1$, the cost of estimating sKL is $O(d(n_0+n_1)^2)$. However, the complexity can be reduced by adopting advanced searching techniques, which rely for instance on the $k$-d tree structures [61], as we did in our MATLAB implementation.

## 5.5 Experiments

Experiments are meant to demonstrate the effectiveness of CCM and the limitations of the experimental practices adopted in the literature (Section 5.5.1). More precisely, in our experiments we inject changes in both synthetically generated and real world datasets, and then measure the magnitude of the changes introduced by all the different methods.

We first perform experiments on Gaussian datastreams (Section 5.5.2), where we can exactly measure the symmetric Kullback-Leibler divergence between the pre- and post-change distributions, then, we introduce changes in two real world datasets (Section 5.5.3). Section 5.5.4 is devoted to the empirical analysis of the convergence of CCM and NP-CCM. To avoid numerical issues, in all our experiments we standardize the components of each datasets by subtracting the sample mean and dividing by the sample standard deviation.

### 5.5.1 Considered Methods

We compare CCM and NP-CCM against other methods that used in the change-detection literature to introduce changes in real world datasets [14, 59, 77]:

- **CCM** is configured to identify roto-translation parameters yielding $\mathrm{sKL}(\widehat{\phi}_0, \widetilde{\phi}_1) = 1$.
- **NP-CCM**, where sKL is estimated using the method described in [89], configured to yield changes of magnitude 1.
- **Offset**: a 1-sigma offset is added to each component of the input data. This change model corresponds to roto-translations where $Q = I_d$ and $\mathbf{v} = [\sigma_1, \dots, \sigma_d]^T$, being $\sigma_i$ the sample standard deviation of the $i$-th component.
- **Normalized Offset**: an offset, having magnitude that decreases as $d$ increases, is added to each component of the input data. This change model corresponds to roto-translations having $\mathbf{v} = [\sigma_1/\sqrt{d}, \dots, \sigma_d\sqrt{d}]^T$ and $Q = I_d$. When $\phi_0$ is a Gaussian distribution having independent components, this change model yields $\mathrm{sKL}(\phi_0, \phi_1) = 1$.
- **Swap**: two randomly chosen components of the input data are swapped. This change model corresponds to a roto-translation where $Q$ is a permutation matrix and $\mathbf{v} = 0$.

While these are the most commonly adopted methods, the literature presents other solutions, such as mixing different datasets [78] or swapping classes of labeled datasets [15, 79]. These methods, however, depend on many degrees of freedom, which would yield a large variance in the results, and as such have not been considered in our analysis.

### 5.5.2 Gaussian Datastreams

We consider Gaussian datastreams generated by $\phi_0 = \mathcal{N}(\mu_0, \Sigma_0)$. The post-change distribution $\phi_1(\cdot) = \phi_0(Q\cdot + \mathbf{v})$ is still a Gaussian having mean $\mu_1 = Q^T(\mu_0 - \mathbf{v})$ and covariance matrix $\Sigma_1 = Q^T\Sigma_0 Q$. For each dimension $d \in \{1, 2, 4, \dots, 128\}$ we generate 1000 datasets containing 20000 i.i.d. realizations of $\phi_0$ each, where the values of $\mu_0$ and $\Sigma_0$ have been randomly defined in each dataset. From each dataset, we generate a datastream using the all the methods described above to introduce a change at $\tau = 1000$.

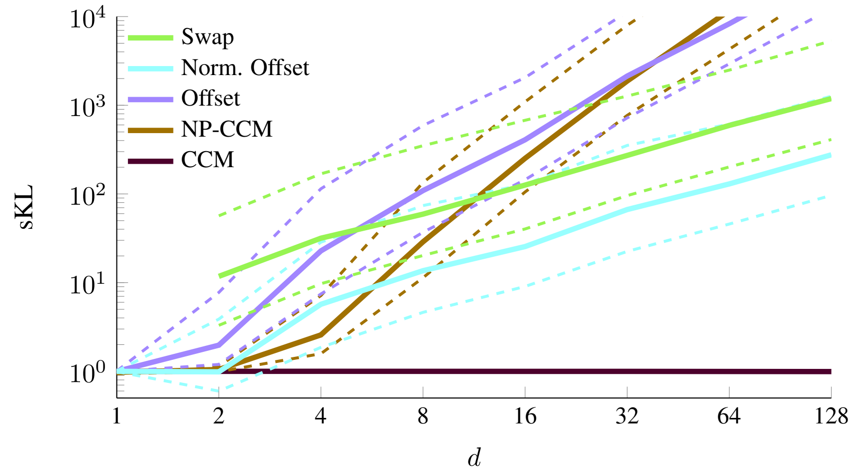


**Figure 5.2:** *Values of* $\mathrm{sKL}(\phi_0, \phi_1)$ *obtained on Gaussian datasets using different methods to inject changes at dimension* $d \in \{1, 2, 4, 8, 16, 32, 64, 128\}$. *Solid lines represent the median values, while dashed lines represent the first and the third quartiles.*

On Gaussian datastreams we can exactly compute the magnitude of the introduced changes, since the sKL is given by

$$\mathrm{sKL}(\phi_0, \phi_1) = \frac{1}{2}\Big[\mathrm{tr}(\Sigma_1^{-1}\Sigma_0) + \mathrm{tr}(\Sigma_0^{-1}\Sigma_1) + \\ + (\mu_1 - \mu_0)^T(\Sigma_1^{-1} + \Sigma_0^{-1})(\mu_1 - \mu_0) - 2d\Big]. \tag{5.27}$$

This expression is used as a reference to assess the magnitude of the introduced changes.

Figure 5.2 shows that CCM generates changes yielding $\mathrm{sKL}(\phi_0, \phi_1)$ equals to 1 (the target value) for each dimension $d$. This result is not surprising, since we set $k = 1$ and the assumption that $\phi_0$ can be well approximated by a Gaussian mixture is here perfectly met. In contrast, NP-CCM cannot successfully control the change magnitude, which reaches 2 when $d = 4$. This is due to the fact that 20000 samples are not sufficient to obtain an accurate estimate of sKL by (5.25) when $d$ increases, as noted in [89]. Other experimental practices can not preserve the change magnitude, since sKL steadily increases with $d$. Remarkably, also the magnitude of normalized offset increases, and this indicates that a coarse approximation of $\phi_0$, which in this case ignores the correlation among components, does not allow to control the change magnitude. Figure 5.2 shows that also the dispersion of the change magnitude increases (note the logarithmic scale in the axis) thus that the introduced changes have considerably different mag-

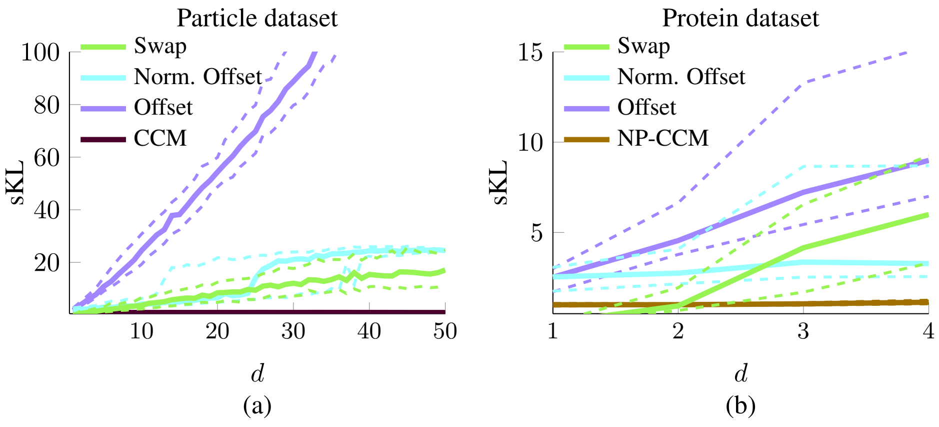


**Figure 5.3:** *Values of* $\mathrm{sKL}(\phi_0, \phi_1)$ *obtained on (a) Particle dataset and (b) Protein dataset using different methods to introduce changes in datastreams having different dimension d. Solid lines represent the median values, while the dashed lines represent the first and the third quartiles.*

nitudes when $d$ increases.

Such an increasing magnitude prevents the use of these techniques to correctly study the change-detection performance when data dimension scales. In practical applications, in fact, it is not expected that the change magnitude necessarily increases with $d$. For instance, when the dimension of the change-detection problem increases because of components that *i)* are not affected by the change, and *ii)* are independent from components that change, the change magnitude does not increase with $d$.

### 5.5.3 Real World Datastreams

In the second experiment we generate datastreams from two real world datasets from the UCI repository [76]. The *MiniBooNE Particle* dataset is the same one used in previous chapter and has dimension $d_{max} = 50$. We adopt a mixture of $k = 4$ Gaussians to approximate $\phi_0$. The second dataset is the *Physicochemical Properties of Protein Tertiary Structure* (Protein) dataset, which has dimension $d_{max} = 9$. By analyzing the marginal distribution over components and pairs of components, we conclude that a GM seems not to properly fit this latter dataset, and we have adopt NP-CCM with a maximum of $d = 4$ components. Larger values of $d$ would have required too many samples for correctly estimating the symmetric Kullback-Leibler divergence by (5.25).

In this experiment we cannot exactly compute the symmetric Kullback-

Leibler divergence between the pre- and post- change distributions, since $\phi_0$ is unknown. Therefore, we resort to estimating the change magnitude as follows. We split each dataset $S_0$ in two set $S_{0,1}$ (training) and $S_{0,2}$ (validation): the set $S_{0,1}$ contains approximately one third of the samples of $S_0$ and it is used for estimating the roto-translation parameters by all the methods, while $S_{0,2}$ is exclusively used for estimate the sKL corresponding to the identified changes. From $S_{0,1}$ we generate 1000 datastreams for each dimension $d = 1, \dots, d_{max}$ by randomly choosing $d$ components out of the $d_{max}$ available and compute $Q$ and $\mathbf{v}$ using the considered methods[3]. Then we estimate the corresponding $\text{sKL}(\widehat{\phi}_0, \widehat{\phi}_0(Q \cdot + \mathbf{v}))$ over $S_{0,2}$. More precisely, in the experiments on the Particle dataset we fit $\widehat{\phi}_0$ on $\S_{0,1}$, and we split the set $S_{0,2}$ in two subsets $V_0$ and $V_1$. Then we estimate the symmetric Kullback-Leibler divergence using (5.7), where the $\psi_0$ and $\psi_1$ are defined from $\widehat{\phi}_0$ (which is fitted on $S_{0,1}$) and $\widehat{\phi}_0(Q \cdot + \mathbf{v})$, respectively. In the experiments on the Protein dataset, the symmetric Kullback-Leibler divergence is estimated using Algorithm 5.4, where $B_0$ and $B_1$ in (5.25) are extracted exclusively from $S_{0,2}$.

Figures 5.3 shows the values sKL obtained when increasing $d$, and demonstrates that CCM and NP-CCM can generate changes having controlled magnitude. The Swap, Offset and Normalized Offset do not preserve the change magnitude, and yield changes that are more apparent when $d$ increases. These results are consistent with those emerging from the experiments on the Gaussian dataset, and further indicate that controlling the change magnitude is very important when manipulating real-world datasets.

### 5.5.4 Execution times

To estimate the execution times of CCM and NP-CCM, we count the number of iterations that are required to CCM and NP-CCM to converge on the Particle and Protein datasets, respectively. These numbers are shown in Figure 5.4(a) and 5.4(b) as function of $d$. We observe that, in general, less than 20 iterations are enough to converge to the target value $\kappa = 1$ with a tolerance of $\varepsilon = 0.01$. This translates in execution times that are reported in Figure 5.5 and show that CCM can be used to generate a large number of datastreams for the evaluation of change-detection algorithms. This times are measured using our MATLAB implementation of CCM and NP-CCM on a PC mounting an Intel Core i5 2.30 GHz CPU and 12 GB RAM. Computational times of the other approaches are not reported even though these are substantially smaller since the Swap and Offset methods do not require

[3]Note that the swap method does not need training data to define the permutation matrix.

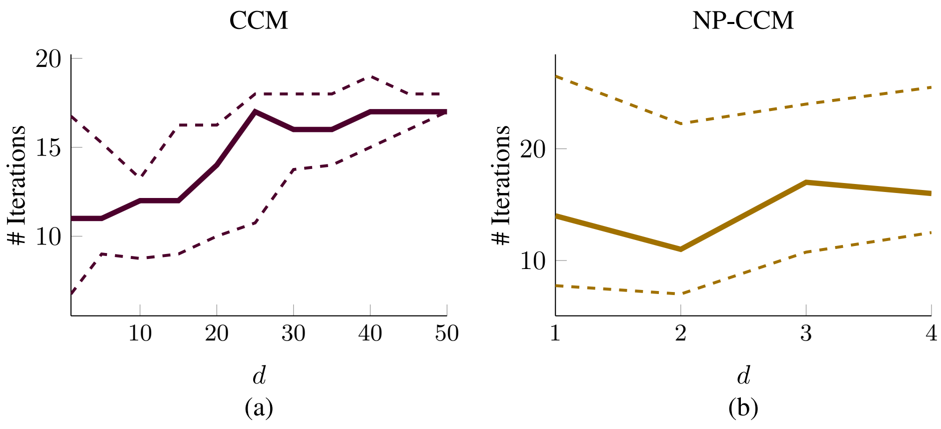


**Figure 5.4:** *Number of iterations required for the convergence of Algorithm 5.3 for (a) CCM and (b) NP-CCM, in case of the Particle and the Protein dataset, respectively. For each considered dimension, less the 20 iterations are sufficient to achieve convergence, with a tolerance $\varepsilon = 0.01$, i.e. 1% of the target sKL. The solid lines represent the median computed over 50 realizations, while the dashed lines represents the first and the third quartiles.*

any computation, while Normalized Offset method has only to compute the standard deviation of each components over the entire dataset.

## 5.6 MATLAB Framework

We have implemented CCM in a MATLAB package that is publicly available for download[4]. This package allows to import any numerical dataset $S_0$ and generates an arbitrarily number of datastreams of the same length, containing a change of a desired magnitude at a given location. The main class of the framework is `CCMframework`, which implements all the functionalities used in Algorithm 5.1, namely: fitting of the Gaussian mixture $\widehat{\phi}_0$ to the whole dataset $S_0$, computation of the roto-translation, and preparation of the datastreams. The `CCMframework` class has two methods: the constructor and `generateDatastreams`.

The constructor of `CCMframework` takes as input the dataset $S_0$ and a flag parameter that specifies which version of the framework to use (CCM or NP-CCM). In case of CCM, the constructor fits a Gaussian mixture to the dataset (Algorithm 5.1, line 1) for a given number $k$ of Gaussians, which has to be defined by the user, for instance through the procedure described in in Section 5.2.1. Then an object of the class `gmDistr` is accordingly

[4] `home.deib.polimi.it/carrerad`

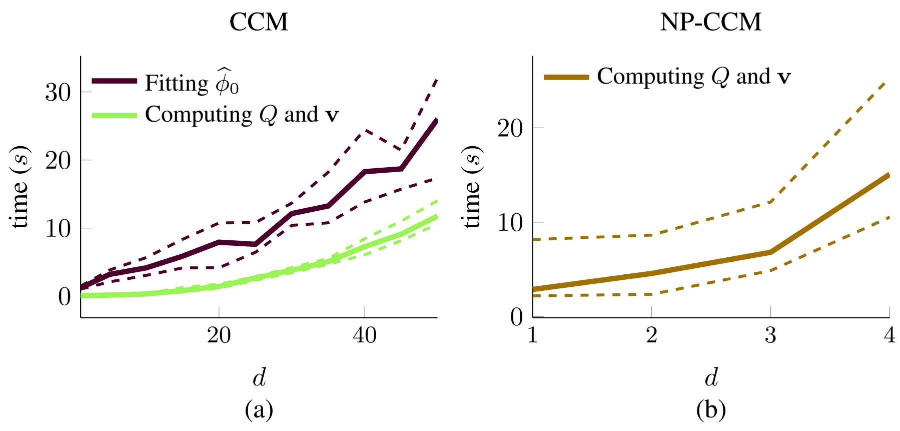


**Figure 5.5:** *Execution times of (a) CCM and (b) NP-CCM required to compute the roto-translation parameters on the Particle and Protein datasets, respectively. The solid lines represent the median computed over 50 realizations, while the dashed lines represents the first and the third quartiles. CCM requires an additional step to fit the Gaussian mixture $\widehat{\phi}_0$. Although this step is more computationally demanding than the computation of the rotation matrix $Q$ and the shift vector $\mathbf{v}$, it has to be performed only once when computing several change parameters for the same dataset.*

instantiated. When the flag parameter indicates NP-CCM, the constructor instantiates an object of the class `empiricalDistr`, which relies on an `KDTreeSearcher` MATLAB object to efficiently compute the 1-nn distance in (5.25). Both of `gmDistr` and `empiricalDistr` implement the method `symmetricKullbackLeibler`, which is used to estimate the symmetric Kullback-Leibler divergence between the pre- and post-change distributions. The `gmDistr` class implements the procedure described in Section 5.2.2, while the class `empiricalDistr` implements Algorithm 5.4.

The method `generateDatastreams` implements Algorithm 5.2 to initialize the roto-transalation parameters, and then runs the bisection method in Algorithm 5.3 until the matrix $Q$ and the vector $\mathbf{v}$ provide the desired change magnitude or when the number of iterations reaches a maximum value. Finally, the datastream is assembled by randomly selecting samples from $S_0$ and by applying the computed roto-translation to samples that are located after the specified change-point location.

# CHAPTER 6

# QuantTree

In this chapter we introduce QuantTree, a novel change-detection algorithm pursuing histograms to approximate $\phi_0$. QuantTree uses an iterative splitting scheme that adaptively defines the histogram bins to ease the detection of of any distribution change. We theoretically prove that the distribution of any statistics computed over histograms defined by QuantTree does not depend on $\phi_0$, thus we can compute thresholds for these test statistics using Monte Carlo simulations. Moreover, since QuantTree is a non-deterministic algorithm, we can exploit the diversity of the computed histograms by combining them in an ensemble and show that this approach mitigate the effect of detectability loss investigated in Chapter 4.

In Section 6.1 we formally define the histograms, and in Section 6.2 we introduce QuantTree. Section 6.3 is devoted to the demonstration of our main results, while in Section 6.4 we show how to aggregate histograms computed by QuantTree in ensembles. Finally, we show the results of our experimental campaign in Section 6.5 and discuss them in Section 6.6.

## 6.1 Histograms for Change Detection

We consider histograms to approximate the unknown stationary distribution having probability density function $\phi_0$, whose support is $\Omega \subset \mathbb{R}^d$. We define a histogram as:

$$h = \{(B_k, \pi_k)\}_{k=1,\dots,K}, \tag{6.1}$$

where the $K$ subsets (*bins*) $B_k \subseteq \Omega$ form a partition of $\Omega$, i.e., $\bigcup_{k=1}^{K} B_k = \Omega$ and $B_j \cap B_i = \emptyset$, for $j \neq i$, and each $\pi_k \in [0,1]$ corresponds to the probability for data generated from $\phi_0$ to fall inside $B_k$. Both the bins $\{B_k\}_k$ and probabilities $\{\pi_k\}_k$ can be adaptively defined from training data $S_0$, and in particular $\pi_k$ is typically estimated as $\widehat{\pi}_k = L_k/N$, i.e. the number of training samples $L_k$ belonging to $A_k$ over the number of points in $S_0$.

As described in Section 2.1, we analyze the incoming data in batches $W = \{\mathbf{s}_1, \dots, \mathbf{s}_\nu\}$ of $\nu$ samples and perform the hypothesis tests in (2.2). We focus on hypothesis tests that are based on a test statistic $\mathcal{T}_h$ defined over the histogram $h$, like for instance the Pearson statistic [64]. Thus, $\mathcal{T}_h$ uniquely depends on $\{y_k\}_{k=1,\dots,K}$, where $y_k$ denotes the number of samples in $W$ falling in $B_k$. We detect a change in the incoming $W$ when

$$\mathcal{T}_h(W) = \mathcal{T}_h(y_1, \dots, y_K) > \delta, \tag{6.2}$$

where $\delta \in \mathbb{R}$ is a threshold that controls the FPR.

Our goal is two-fold: *i*) learn a histogram $h$ from $S_0$ to be used for change-detection purposes and *ii*) for each given test statistic $\mathcal{T}_h$ and reference FPR value $\alpha$, define a threshold $\delta$ such that

$$P_{\phi_0}(\mathcal{T}_h(W) > \delta) \leq \alpha, \tag{6.3}$$

where $P_{\phi_0}$ denotes the probability under the null hypothesis that $W$ contains samples generated from $\phi_0$. We observe that in classical statistical hypothesis settings, the threshold $\delta$ is set such that the equality holds in (6.3). However, when the statistic $\mathcal{T}_h$ assume discrete values, as for statistics computed over the histogram, equality might not hold, thus $\alpha$ becomes an upper-bound for the probability of type I errors.

## 6.2 The QuantTree Algorithm

Here we introduce QuantTree, an algorithm to define histograms $h$ through an iterative binary splitting of the input space $\Omega$. This algorithm takes as

**Algorithm 6.1** QuantTree

```
Input: Training set $S_0$ containing $N$ stationary points in $\Omega$; number of bins $K$; target
    probabilities $\{\pi_k\}_k$.
Output: The histogram $h = \{(B_k, \widehat{\pi}_k)\}_k$.
 1: Set $N_0 = N$, $L_0 = 0$.
    // Iterative split of the bins
 2: for $k = 1, \dots, K$ do
 3:     Set $N_k = N_{k-1} - L_{k-1}$, $\Omega_k = \Omega \setminus \bigcup_{j<k} B_j$, and $L_k = \text{round}(\pi^k N)$.
 4:     Choose a random component $i \in \{1, \dots, d\}$.
 5:     Define $z_n = [\mathbf{s}_n]_i$ for each $\mathbf{s}_n \in \Omega_k$.
 6:     Sort $\{z_n\}$: $z_{(1)} \leq z_{(2)} \leq \dots z_{(N_k)}$.
 7:     Draw $\beta \in \{0, 1\}$ from a Bernoulli$(0.5)$.
 8:     if $\beta = 0$ then
 9:         Define $B_k = \{\mathbf{s} \in \Omega_k \quad [\mathbf{s}]_i \leq z_{(L_k)}\}$.
10:     else
11:         Define $B_k = \{\mathbf{s} \in \Omega_k \quad [\mathbf{s}]_i \geq z_{(N_k - L_k + 1)}\}$.
12:     end if
13:     Set $\widehat{\pi}_k = L_k / N$.
14: end for
```

input a training set $S_0$ containing $N$ stationary points, the number of bins $K$ in the histogram, and the target probabilities on each bin $\{\pi_k\}_{k=1,\dots,K}$, and returns a histogram $h = \{(B_k, \widehat{\pi}_k)\}_{k=1,\dots,K}$, where each $\widehat{\pi}_k$ represents an estimate of the probability for a sample drawn from $\phi_0$ to fall in $B_k$.

Algorithm 6.1 presents in detail QuantTree, which constructs a new bin of $h$ at each step $k$. We denote by $\Omega_k \subseteq \Omega$ the subset of the input space that still has to be partitioned (i.e., $\Omega_k = \Omega \setminus \bigcup_{j<k} B_k$) and by $N_k$ the number of points of $S_0$ belonging to $\Omega_k$. We compute (line 3) the number of training points that has to fall inside $B_k$ as $L_k = \text{round}(\pi_k N)$. The subset $B_k$ is then defined by splitting $\Omega_k$ along a component $i \in \{1, \dots, d\}$ that is randomly chosen with uniform probability (line 4). The splitting point is defined by sorting $z_n = [\mathbf{s}_n]_i$, i.e., the values of the $i$-th component for each $\mathbf{s}_n \in \Omega_k$ (lines 5). We thus obtain $z_{(1)} \leq z_{(2)} \leq \cdots \leq z_{(N_k)}$ (line 6) and we define $B_k$ by splitting $\Omega_k$ w.r.t. $z_{(L_k)}$ or $z_{(N_k - L_k + 1)}$ (lines 7-11). In both cases $B_k$ contains $L_k$ points among the $N$ in $S_0$, thus the estimated probability of $B_k$ is $\widehat{\pi}_k = L_k / N$ (line 13). This procedure is iterated until $K$ subsets are extracted.

QuantTree divides $\Omega$ in a given number of subsets, where each set $B_k$ has an estimated probability $\widehat{\pi}_k \simeq \pi_k$, and the equality holds when $\pi_k N$ is integer. Since the probabilities $\pi_k$ are set a priori, in what follows we use $\pi_k$ in place of $\widehat{\pi}_k$. Indexes $i$ and parameter $\beta$ are randomly chosen to add variability to the histogram construction. Figure 6.1 shows a tree obtained

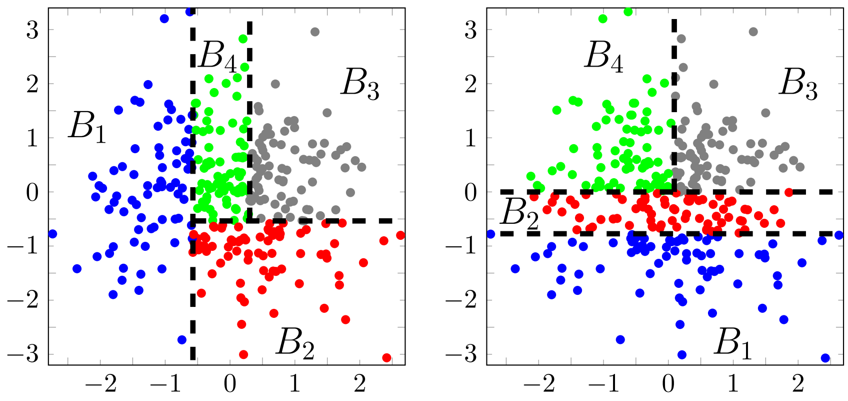


**Figure 6.1:** *Two examples of histograms with $K = 4$ bins computed by QuantTree to yield uniform density on the bins.*

from a bivariate Gaussian training set, defined by $K = 4$ bins, each having probability $\pi_k = N/4$.

### 6.2.1 Computation of Distribution-Free Test Statistics

A key feature of a histogram computed by QuantTree is that any statistic $\mathcal{T}_h$ built over it has a distribution that is independent from $\phi_0$. This result follows from Theorem 6.1, that is proved in Section 6.3.

**Theorem 6.1.** *Let $\mathcal{T}_h(\cdot)$ be defined as in* (6.2) *over the histogram $h$ computed by QuantTree. When $W \sim \phi_0$, the distribution of $\mathcal{T}_h(W)$ depends only on $\nu$, $N$, $K$ and $\{\pi_k\}_k$.*

Theorem 6.1 implies that we can numerically compute the thresholds for any statistic $\mathcal{T}_h$ defined on histograms, provided $\nu$, $N$, $K$ and $\{\pi_k\}$, thus disregarding $\phi_0$ and the data dimension $d$. To this end, we synthetically generate data from a conveniently chosen distribution $\psi_0$, and we follow the procedure outlined in Algorithm 6.2 to estimate the threshold $\delta$ for the hypothesis test in (2.2) yielding a desired FPR $\alpha$. At first we generate $M$ training sets $\{S_m\}_{m=1,\dots,M}$, sampling $N$ points from $\psi_0$ and, for each training set, we build a histogram $h_m$ using QuantTree (lines 2-3). Then, for each $h_m$ we generate a batch $W_m$ of $\nu$ points drawn from $\psi_0$, and compute the value of the statistic $\delta_m = \mathcal{T}_h(W_m)$ (lines 4-5). Finally, we estimate $\delta$ (line 7) from the set $T_M = \{\delta_1, \dots, \delta_M\}$ as the $1 - \alpha$ quantile of the empirical distribution of $\mathcal{T}_h$ over the generated batches, i.e.

$$\delta = \min\Big\{ t \in T_M \ : \ \#\{v \in T_M \ : \ v > t\} \leq \alpha M \Big\}, \tag{6.4}$$

**Algorithm 6.2** Numerical procedure to compute thresholds

```
Input: Test statistic $\mathcal{T}_h$; arbitrarily chosen $\psi_0$; the number $M$ of datasets and batches
    to compute the threshold; the number of points $\nu$ in each batch; $N$,$K$, and $\widehat{\pi}_k$ as in
    Algorithm 6.1; the desired FPR $\alpha$.
Output: The value $\delta$ of the threshold
1: for $m = 1, \dots, M$ do
2:     Draw from $\psi_0$ a training set $S_m$ of $N$ samples.
3:     Use QuantTree to compute the histogram $h_m$ with $K$ bins and target probabilities
       $\{\pi_k\}_k$ over $S_m$.
4:     Draw a batch $W_m$ containing $\nu$ points from $\psi_0$.
5:     Compute the value $\delta_m = \mathcal{T}_h(W_m)$.
6: end for
7: Compute the threshold $\delta$ as in (6.4).
```

where $\#A$ denotes the cardinality of a set $A$.

To take full advantage of the distribution-free nature of the procedure, we set $\psi_0$ to a univariate uniform distribution $U(0,1)$. This allows to obtain high accuracy on the estimation of the thresholds, since we can use very large values of $M$ with limited computational cost.

### 6.2.2 Considered Statistics

We consider two meaningful examples of statistics $\mathcal{T}_h$ that can be employed for batch-wise monitoring through histograms: the Pearson statistic and the total variation [64]. The Pearson statistic is defined as

$$\mathcal{T}_h^P(W) = \sum_{k=1}^{K} \frac{(y_k - \nu\pi_k)^2}{\nu\pi_k}, \tag{6.5}$$

while the total variation is defined as

$$\mathcal{T}_h^{TV}(W) = \frac{1}{2}\sum_{k=1}^{K} |y_k - \nu\pi_k| \ . \tag{6.6}$$

It is well known that, when $\{\pi_k\}_k$ are the true probabilities of the bins $\{B_k\}_k$, under the null hypothesis the statistic $\mathcal{T}_h^P(W)$ is asymptotically distributed as a $\chi^2_{K-1}$. However, when the $\pi_k$ are estimated, the threshold obtained from the $\chi^2_{K-1}$ distribution does not allow to properly control the FPR, and this effect is more evident when $y_k$ is small. In contrast, thresholds defined by Algorithm 6.2 hold also in case of limited sample size, since they are not based on an asymptotic result.

| $\alpha$ | Pearson | | Total Variation | | | |
|---|---|---|---|---|---|---|
| | $K = 32$ | $K = 128$ | $K = 32$ | $K = 128$ | $N$ | $\nu$ |
| 0.001 | 64 | 192 | 25 | 43 | 4096 | 64 |
| | 62.75 | 187 | 52 | 85 | 16384 | 256 |
| 0.01 | 54 | 172 | 23 | 42 | 4096 | 64 |
| | 53.25 | 171 | 47 | 81 | 16384 | 256 |
| 0.05 | 46 | 156 | 21 | 41 | 4096 | 64 |
| | 45.75 | 157 | 44 | 78 | 16384 | 256 |

**Table 6.1:** *Examples of thresholds $\delta$ that guarantee FPR below $\alpha$ using a uniform histogram h, i.e. by settings $\pi_k = 1/K$, $k = 1, \dots, K$. The thresholds are computed by Algorithm 6.2 using $U(0, 1)$ as $\psi_0$ and different values of $N$, $\nu$ and $K$.*

We report in 6.1 the thresholds for these two statistics computed for different values of $N, K, \nu$ and choosing $\pi_k = 1/K, k = 1, \dots, K$. These values have been computed applying the procedure described in Algorithm 6.2 with $M = 2.5 \cdot 10^6$. We note that both statistics $\mathcal{T}_h^P$ and $\mathcal{T}_h^{TV}$ assume only discrete values, therefore it is not always possible to set the threshold $\delta$ yielding the FPR exactly equal to $\alpha$, but only to ensure that the FPR does not exceed $\alpha$.

### 6.2.3 Computational Remarks

We remark that since the histogram $h$ computed by QuantTree is exclusively defined on the marginal probabilities of single components, the dimensionality of the input data $d$ does not impact the overall computational cost. In fact, the computational cost of QuantTree is dominated by sorting the covariates (Algorithm 6.1 line 6), which is performed $K$ times on an progressively smaller number of samples at each iteration. Therefore, the overall complexity of constructing a QuantTree is $O(KN \log N)$. In case of univariate distribution (i.e., $d = 1$), the complexity is reduced to $O(N \log N)$, since the partition $\{B_k\}_k$ can be defined through a single sorting operation.

Since any histogram $h$ computed by QuantTree can be represented as a tree structure, it is very efficient to identify the bin where any testing point belongs to. In fact, during monitoring, at most $K$ IF-THEN operations (that reduces to $\log K$ when $d = 1$) have to be performed for each input sample s. Moreover, in contrast with histograms based on regular grids, the number of bins $K$ is here a priori defined, and does not need to grow exponentially with $d$.

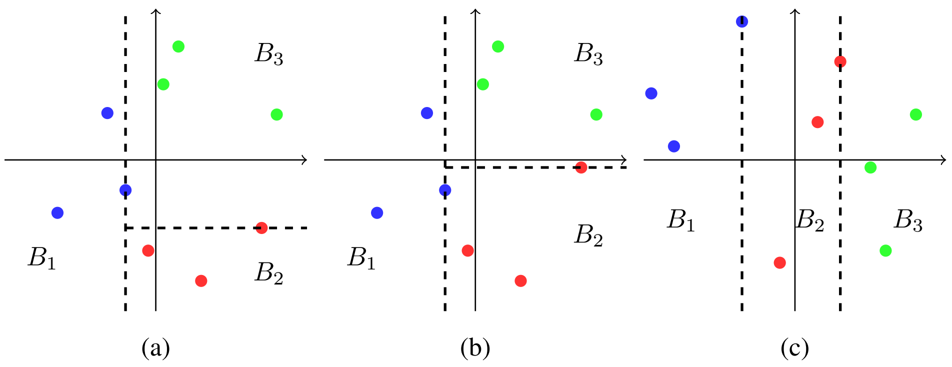


**Figure 6.2:** *Examples of values assumed by $\widetilde{L}_k$ in three different configurations, when $N = 9$ and $L_1 = L_2 = L_3 = 3$. In these cases $\widetilde{L}_1 = L_1$, while in (a) $\widetilde{L}_2 = 3$, in (b) $\widetilde{L}_2 = 5$, and in (c) $\widetilde{L}_2 = 6$. Note that when QuantTree chooses always the same component, we have that $\widetilde{L}_2 = L_1 + L_2$, as in (c).*

## 6.3 Theoretical Analysis

We prove Theorem 6.1 showing that the distribution of any test statistic $\mathcal{T}_h$ defined over an histogram $h$ computed by QuantTree does not depend on $\phi_0$. To this end, we first prove some preliminary propositions to characterize the distribution of the true probability of each bin $B_k$ under $\phi_0$:

$$p_k = P_{\phi_0}(B_k), \tag{6.7}$$

which is also a random variable as it depends on the training set $S_0$.

For the sake of simplicity, we assume that QuantTree always splits with respect to the left tail, i.e., $\beta = 0$ in line 8 of Algorithm 6.1 (proofs hold when $\beta \sim \text{Bernoulli}(0.5)$) and, to simplify the notation, we will omit the subscript $_{\phi_0}$ from $P_{\phi_0}$, thus $P$ denotes the probability computed w.r.t. $\phi_0$. The following proposition will be used to derive the distributions of $p_k$.

**Proposition 6.1.** *Let $\mathbf{s}_1, \dots, \mathbf{s}_M$ be i.i.d. realizations of a continuous random vector $\mathbf{S}$ defined over $\mathcal{D} \subseteq \mathbb{R}^d$. Let us define the $i$-th component of $\mathbf{s}$ as $z = [\mathbf{s}]_i$, and denote with $z_{(1)} \leq z_{(2)} \leq \cdots \leq z_{(M)}$ the $M$ sorted components of $\mathbf{s}_1, \dots, \mathbf{s}_M$. For any $L \in \{1, \dots, M\}$ we define the set*

$$Q_{i,L} := \{\mathbf{s} \in \mathcal{D} \;:\; [\mathbf{s}]_i \leq z_{(L)}\}. \tag{6.8}$$

*Then, for each $i \in \{1, \dots, d\}$, the random variable $p = P_{\mathbf{S}}(Q_{i,L})$ is distributed as a Beta$(L, M - L + 1)$.*

*Proof.* The proof consists of showing that $p$ is an order statistic of the uniform distribution, which in turns follows a Beta distribution. For this purpose, we consider S defined over $\mathbb{R}^d$ and $P_{\mathbf{S}}(\mathbb{R}^d \setminus \mathcal{D}) = 0$, thus $p$ can be expressed as

$$\begin{aligned} p = P_{\mathbf{S}}(Q_{i,L}) = P_{\mathbf{S}}(\mathbf{s} \in \mathbb{R}^d \, : \, [\mathbf{s}]_i \leq z_{(L)}) = \\ = P_Z(z \in \mathbb{R} \, : \, z \leq z_{(L)}), \end{aligned} \tag{6.9}$$

where $P_Z$ denotes the marginal probability of $Z = [\mathbf{S}]_i$, namely the marginal of S w.r.t. the component $i$. We denote with $F_Z$ the cumulative distribution of $Z$ and define $U = F_Z^{-1}(Z)$ and $u_n = F_Z^{-1}(z_n)$, $n = 1, \dots, M$, where

$$F_Z^{-1}(z) = \inf\{t \in \mathbb{R} \, : \, F_Z(t) > z\}. \tag{6.10}$$

The function $F_Z^{-1}(\cdot)$ is monotonically nondecreasing, thus it preserves the order and the $L$-th sorted value of $\{u_n\}$ can be computed as $u_{(L)} = F_Z^{-1}(z_{(L)})$. Then, (6.9) becomes

$$\begin{aligned} p = P_Z(z \in \mathbb{R} \, : \, z \leq z_{(L)}) = \\ = P_U(u \in [0,1] \, : \, u \leq u_{(L)}) = F_U(u_{(L)}) = u_{(L)}. \end{aligned} \tag{6.11}$$

Since $U$ follows a uniform distribution over $[0, 1]$, it follows that $p$ is the $L$-th order statistic of the uniform distribution, that is a distributed as a Beta$(L, M - L + 1)$ [93]. $\square$

Thus, $p_1$ in (6.7), namely the probability of $B_1$ under $\phi_0$, is distributed as a Beta$(L_1, N - L_1 + 1)$. To derive the distribution of the remaining $p_k$, $k \geq 2$, we define the conditional probability

$$P_{B_1}(\mathbf{s} \in A) = P_{\phi_0}(\mathbf{s} \in A \mid \mathbf{s} \notin B_1), \tag{6.12}$$

where $A$ is any Borel subset of $\Omega$. Then, from the definition of conditional probability and the fact that $\mathbf{s}_1, \dots, \mathbf{s}_N$ are i.i.d. according to $\phi_0$, it can be easily proved that the $N - L_1$ points that do not belong to $B_1$ are i.i.d. according to $P_{B_1}$. Therefore, we can apply Proposition 6.1 to the subset of the $N - L_1$ points that do not fall in $S_1$ by setting $\mathcal{D} = \mathbb{R}^d \setminus S_1$ and considering $P_{S_1}$ in place of $P_{\mathbf{S}}$. Thus, the random variable $\widetilde{p}_2 = P_{S_1}(S_2)$ is distributed as Beta$(L_2, N_2 - L_2 + 1)$, where $N_2 = N - L_1$. Iterating the above procedure, we obtain that all the random variables $\widetilde{p}_k$, $k = 1, \dots, K$, defined as[1]

$$\widetilde{p}_k = P_{\bigcup_{j=1}^{k-1} B_j}(B_k), \tag{6.13}$$

are distributed as Beta$(L_k, N_k - L_k + 1)$, where $N_k = N - \sum_{j=1}^{k-1} L_j$.

[1]We adopt the following conventions: an empty union of sets is the empty set, an empty sum is zero, and an empty product is 1.

We remark the different roles of $p_k$ and $\widetilde{p}_k$. While $p_k$ in (6.7) the measure of the bin $B_k$ under $\phi_0$, $\widetilde{p}_k$ in (6.13) is the ratio between $p_k$ and the measure under $\phi_0$ of $\Omega_k = \mathbb{R}^d \setminus \bigcup_j^{k-1} B_j$, namely the space that remains to be partitioned at step $k$. As an example, for a tree with $K = 3$ leaves, if we set target probabilities $\pi_1 = \pi_2 = \pi_3 = 1/3$, we obtain $\widetilde{p}_1 \approx 1/3$, $\widetilde{p}_2 \approx 1/2$ and $\widetilde{p}_3 = 1$. To prove Theorem 6.1 we need to derive the distribution of $p_k$, that are expressed in terms of $\widetilde{p}_k$ by the following proposition.

**Proposition 6.2.** *In case of histograms defined by QuantTree, the following relation holds between $p_k$ and $\widetilde{p}_k$:*

$$p_k = \widetilde{p}_k \cdot (1 - \sum_{j=1}^{k-1} p_j) = \widetilde{p}_k \prod_{j=1}^{k-1}(1 - \widetilde{p}_j). \tag{6.14}$$

*Proof.* From the law of total probability we have that

$$\begin{aligned} p_k &= P_{\phi_0}(\mathrm{s} \in B_k) = \\ &= P_{\phi_0}\left(\mathrm{s} \in B_k \mid \mathrm{s} \notin \cup_{j=1}^{k-1} B_j\right) \cdot P_{\phi_0}\left(\mathrm{s} \notin \cup_{j=1}^{k-1} B_j\right) + \\ &\quad + P_{\phi_0}\left(\mathrm{s} \in B_k \mid \mathrm{s} \in \cup_{j=1}^{k-1} B_j\right) \cdot P_{\phi_0}\left(\mathrm{s} \in \cup_{j=1}^{k-1} B_j\right). \end{aligned} \tag{6.15}$$

Since sets $\{B_k\}$ defined by QuantTree are disjoint, it follows that $B_k$ and $\bigcup_{j=1}^{k-1} B_j$ are also disjoint, thus the second term in the sum in (6.15) is equal to 0. The first equality in (6.14) follows from the definition of $\widetilde{p}_k = P_{\phi_0}(\mathrm{s} \in B_k \mid \mathrm{s} \notin \bigcup_{j=1}^{k-1} B_j)$ and the fact that $P_{\phi_0}(\mathrm{s} \notin \bigcup_{j=1}^{k-1} B_j) = 1 - \sum_{j=1}^{k-1} p_j$.

The second equality in (6.14) can be proved by induction over $k$. □

The following proposition allows us to express $p_k$ as a product of independent Beta distributions.

**Proposition 6.3.** *The random variables $\widetilde{p}_k$ defined over histograms computed by QuantTree are independent.*

*Proof.* To prove the independence of the $\widetilde{p}_k$, $k = 1, \dots, K$, we show that $\widetilde{p}_k$ is independent from $\widetilde{p}_j$, $j = 1, \dots, k-1$. In particular, we prove that

$$P_{\phi_0}(\widetilde{p}_k \leq t_k \mid \widetilde{p}_j = t_j,\ j = 1, \dots, k-1) = P_{\phi_0}(\widetilde{p}_k \leq t_k). \tag{6.16}$$

To this end, we follow the proof of Proposition 6.1, and express $\widetilde{p}_k$ as an order statistic of the uniform distribution.

At iteration $k$, QuantTree randomly selects a dimension $i_k$ and performs a split w.r.t. the $L_k$-th order statistic of the $i_k$ components over the remaining $N_k$ points (line 9 of Algorithm 6.1). Let $\widetilde{L}_k$ be the position of this splitting point in $\{z_n = [\mathrm{s}_n]_{i_k},\ n = 1, \dots, N\}$, namely the sequence of ordered

$i_k$ components of all the points in $S_0$. The value of $\widetilde{L}_k \in \mathbb{N}$ depends on realizations $\mathrm{s}_1, \dots, \mathrm{s}_N$, and is a random variable ranging in $\{L_k, \dots, M_k\}$, where $M_k = \sum_{j=1}^{k} L_j$. Obviously, at the first iteration $L_1 = \widetilde{L}_1$ but then the two may differ, as shown in Figure 6.2. Let us now consider the splitting point with respect to $\widetilde{L}_k$, i.e., $z_{(\widetilde{L}_k)}$. From the definition of $\widetilde{p}_k$ we have that

$$\widetilde{p}_k = P_{\bigcup_{j=1}^{k-1} B_j}(B_k) = P_{\bigcup_{j=1}^{k-1} B_j}(z \leq z_{(\widetilde{L}_k)}). \tag{6.17}$$

As in the proof of Proposition 6.1, we denote with $F_Z$ the cdf of $Z = [\mathbf{S}]_{i_k}$, and define $U = F_Z^{-1}(Z)$, that has a uniform distribution on $[0,1]$. Therefore it holds that

$$\begin{aligned}\widetilde{p}_k = P_{\bigcup_{j=1}^{k-1} B_j}(z \leq z_{(\widetilde{L}_k)}) = P_{\bigcup_{j=1}^{k-1} B_j}(u \leq u_{(\widetilde{L}_k)}) = \\ = F_U(u_{(\widetilde{L}_k)}) = u_{(\widetilde{L}_k)}.\end{aligned} \tag{6.18}$$

We use the law of total probability w.r.t. the events $\{\widetilde{L}_k = a\}$, $a \in \{L_k, \dots, M_k\}$, to decompose the left hand side in (6.16):

$$\begin{aligned}&P_{\phi_0}(\widetilde{p}_k \leq t_k \mid \widetilde{p}_j = t_j,\, j = 1, \dots, k-1) = \\ &\quad = P_{\phi_0}(u_{(\widetilde{L}_k)} \leq t_k \mid \widetilde{p}_j = t_j,\, j = 1, \dots, k-1) = \\ &\quad = \sum_{a=L_k}^{M_k} P_{\phi_0}(u_{(\widetilde{L}_k)} \leq t_k \mid \widetilde{L}_k = a, \widetilde{p}_j = t_j,\, j = 1, \dots, k-1) \cdot P_{\phi_0}(\widetilde{L}_k = a) \\ &\quad = \sum_{a=L_k}^{M_k} P_{\phi_0}(u_{(a)} \leq t_k \mid \widetilde{p}_j = t_j,\, j = 1, \dots, k-1) \cdot P_{\phi_0}(\widetilde{L}_k = a).\end{aligned} \tag{6.19}$$

Since the distribution of $u_{(a)}$ does not depend on $\widetilde{p}_j$, we have that

$$P_{\phi_0}(u_{(a)} \leq t_k \mid \widetilde{p}_j = t_j,\, j = 1, \dots, k-1) = P_{\phi_0}(u_{(a)} \leq t_k), \tag{6.20}$$

therefore it follows

$$\begin{aligned}&P_{\phi_0}(\widetilde{p}_k \leq t_k \mid \widetilde{p}_j = t_j) = \\ &\quad = \sum_{a=L_k}^{M_k} P_{\phi_0}(u_{(a)} \leq t_k) \cdot P_{\phi_0}(\widetilde{L}_k = a) \\ &\quad = \sum_{a=L_k}^{M_k} P_{\phi_0}(u_{(\widetilde{L}_k)} \leq t_k \mid \widetilde{L}_k = a) \cdot P_{\phi_0}(\widetilde{L}_k = a) = \\ &\quad = P_{\phi_0}(u_{(\widetilde{L}_k)} \leq t_k) = P_{\phi_0}(\widetilde{p}_k \leq t_k),\end{aligned} \tag{6.21}$$

and (6.16) is proved. □

The proof of Theorem 6.1 follows from Proposition 6.3.

*Proof of Theorem 6.1.* For any stationary distribution $\phi_0$, the random vector $[y_1, \dots, y_K]$ conditioned on $p_1, \dots, p_K$ follows a Multinomial distribution with parameters $(\nu, p_1, \dots, p_K)$ [94]. From Proposition 6.3 each $p_k$ is a product of independent Beta distributions, thus depends only on $\{L_k\}$ and it is independent from $\phi_0$. Therefore any statistic $\mathcal{T}_h$ that is a function of $\{y_k\}$ depends only on $\nu$, and on $N$ and $\{\pi_k\}$ which determines $\{L_k\}$. $\square$

## 6.4 Ensemble of Histograms

Histograms generated by QuantTree might not be able to detect changes that affect single components of the datastream. In fact, at each iteration QuantTree randomly selects a component to perform the split and generate a new bin, thus the components affected by the change might not be chosen, making the histogram unable to detect the change. To guarantee that each component is chosen at least once with high probability, it is necessary to perform a large number of iteration, especially when the data dimension $d$ is large, and generate histograms with a lot of bins. The main drawback of such histograms is that each bin contains very few training samples, which leads to a large variance in the test statistic.

We address this issue by using ensembles of histograms. In particular, we consider histograms having few bins to control the variance of the test statistic, and combine them to eventually partition the data space $\Omega$ in many bins.

### 6.4.1 Ensemble of histograms

An ensemble $\mathcal{E}$ of histograms is a collection of $R$ individual histograms $h_i = \{(B_{i,k}, \widehat{\pi}_{i,k})\}_{k=1,\dots,K_i}$, each estimated from the training set $S_0$, yielding different partitions. The number $K_i$ of bins can be different for each histogram. In this way, we can define hierarchies of partitions, to approximate $\phi_0$ using histograms at different resolutions. For simplicity here we consider a fixed $K$, but our analysis can be easily extended to different settings.

The most important feature for a successful ensemble is the diversity among the single individuals [95]. This is a typical requirement in ensemble methods and multiple hypothesis testing, and it holds also in the change-detection context, as shown in Figure 6.1 that depicts an example of two different histograms computed by QuantTree. If we consider only the histogram (a), any change occurring for value of $s_1$ smaller than $-0.6$ would not be detected. The same issue arises in the histogram (b) which would not be able to detect changes occurring only in region where $s_2 < -0.8$. Yet, the combination of the two histograms allows to detect both changes.

QuantTree provides variability to individual histograms in the ensemble by construction, since it randomly selects a component w.r.t. perform the split. Therefore it can be used straightforwardly to build powerful ensembles for change detection.

### 6.4.2 Statistic over the Ensemble

For each incoming batch $W$ to be tested, we compute the ensemble statistic $\mathcal{T}_{\mathcal{E}}$ by aggregating the statistics $\{\mathcal{T}_{h_i}(W)\}_{h_i \in \mathcal{E}}$ computed over each histogram in $\mathcal{E}$. We consider two relevant aggregation schemes. The first one consists in averaging all the statistics from the individual histograms $\mathcal{T}_{h_i}(W)$:

$$\mathcal{T}_{\mathcal{E},\text{avg}}(W) = \frac{1}{R}\sum_{i=1}^{R} \mathcal{T}_{h_i}(W). \tag{6.22}$$

Such aggregation recalls the bagging [66] and is used to reduce the variance of the statistic, that in principle yields an improvement in the power of the hypothesis testing.

Another aggregation scheme, that is strictly related to the multiple statistical hypothesis settings, is obtained by computing the maximum of $\{T_{h_i}(W)\}$:

$$\mathcal{T}_{\mathcal{E},\text{max}}(W) = \max_{i=1,\dots,R} \mathcal{T}_{h_i}(W). \tag{6.23}$$

In both cases, we reject the null hypothesis if $\mathcal{T}_{\mathcal{E},\text{max}}(W) > \delta_{\mathcal{E}}$, the threshold $\delta_{\mathcal{E}}$ has to be properly set to control probability of type I errors.

### 6.4.3 Setting the threshold

To use the statistic $\mathcal{T}_{\mathcal{E}}$ in the hypothesis test (2.2), we have to set the threshold $\delta_{\mathcal{E}}$ that controls the FPR as in (6.3). Even if the distribution of each individual statistic $\mathcal{T}_{h_i}$ is known, the distribution of the aggregated statistic $\mathcal{T}_{\mathcal{E}}$ in stationary conditions is typically unknown. In fact, the distribution of $\mathcal{T}_{\mathcal{E}}(W),\ W \sim \phi_0$ is difficult to model, since partitions of individuals $h_i$ obviously overlap, and the statistics $\mathcal{T}_{h_i}$ are not independent. Thus, it is typically not possible to obtain a closed-form expression for $\delta_{\mathcal{E}}$. To overcome this problem, we can resort to bootstrap and estimate the distribution of $\mathcal{T}_{\mathcal{E},\text{avg}}(W)$ or $\mathcal{T}_{\mathcal{E},\text{max}}(W),\ W \sim \phi_0$, empirically from bootstrap samples $W$ from $S_0$.

In the case of the statistic $\mathcal{T}_{\mathcal{E},\max}$, another viable option consists in estimating the threshold directly from the thresholds of the individual histograms $\{\delta_i, i = 1, \dots, R\}$, as long as these are known. For instance, thresholds for the Pearson statistic are provided by asymptotic approximation [64] while, in case of histograms constructed via the QuantTree algorithm, $\delta_i$ is known since any statistic $\mathcal{T}_{h_i}$ does not depend on $\phi_0$, and the quantiles can be efficiently precomputed through a Monte Carlo procedure. In these circumstances, it is possible to adopt some multiple hypothesis testing procedure [64]. The most well known of such techniques is perhaps the Bonferroni correction, which defines $\delta$ as the $(1-\alpha)/R$-quantile of $\mathcal{T}_{h_i}$ in stationary conditions. In fact, it can be easily shown that this $\delta$ satisfies (6.3) for $\mathcal{T}_{\mathcal{E},\max}$.

## 6.5 Experiments

We quantitatively assess the advantages of change-detection tests based on QuantTree w.r.t. other general-purpose tests able to detect any distribution change $\phi_0 \to \phi_1$. In particular, we show that: *i*) thresholds provided by Algorithm 6.2 can better control the FPR w.r.t. alternatives based on asymptotic results or bootstrap, *ii*) hypothesis tests based on histograms provided by QuantTree yielding a uniform-density partition of $\mathbb{R}^d$ achieve higher power than other partitioning schemes, and *iii*) ensembles of histograms provided by QuantTree mitigate the detectability loss for large value of $d$.

### 6.5.1 Datasets and Change Models

We employ both synthetic and real-world datasets: we consider several dimensions for the synthetic datasets, from $d = 2$ to $d = 128$. For each dimension $d$, 250 pairs $(\phi_0, \phi_1)$ of Gaussians, where $\phi_0$ has a randomly defined covariance, and $\phi_1 = \phi_0(Q \cdot + \mathbf{v})$ is a roto-transalation of $\phi_0$ such that the symmetric Kullback-Leibler divergence $\text{sKL}(\phi_0, \phi_1) = 1$, as in Chapter 4. The parameters $Q$ and $\mathbf{v}$ of the roto-translations are computed using CCM.

We also employ four real-world high-dimensional sets: *MiniBooNE Particle* ("particle", $d = 50$), *Physicochemical Properties of Protein Tertiary Structure* ("protein", $d = 9$), *Sensorless Drive Diagnosis* ("sensorless", $d = 48$) from the UCI Machine Learning Repository [76], and *Credit Card Fraud Detection* ("credit", $d = 29$) from [96]. We standardize these datasets and add to each component of the "particle" and "sensorless" an imperceivable amount of noise $\eta \sim N(0, 0.001)$ to scramble the many repeated values, which harms histogram construction. For each dataset we simulate 150 changes $\phi_0 \to \phi_1$ by randomly selecting $S_0$ and defining a random shift drawn from a normal distribution.

### 6.5.2 Change Detection Methods

Four of the considered methods rely on the same histogram computed through QuantTree (Algorithm 6.1) to provide a uniform density partition of $\mathbb{R}^d$, i.e. the target probabilities are $\pi_k = 1/K$, $\forall k$. These methods differ only for the threshold adopted and have been considered mainly to investigate the control over false positives.

- **Pearson Distribution Free / TV Distribution Free**: thresholds are computed by Algorithm 6.2 for the Pearson $\mathcal{T}_h^P$ (6.5) and the total variation $\mathcal{T}_h^{TV}$ statistics (6.6), respectively. The adopted thresholds are reported in Table 6.1.

- **Pearson Asymptotic**: thresholds for $\mathcal{T}_h^P$ are provided from the classic $\chi^2$ goodness-of-fit test [64], which provides an asymptotic control over the FPR.

- **TV Bootstrap**: thresholds for $\mathcal{T}_h^{TV}$ are computed empirically by bootstrapping $S_0$.

Three other methods built on different density models have been considered to assess the advantages – also in terms of statistical power – of histograms providing uniform density.

- **Voronoi**: a histogram where the $\{B_k\}_k$ are defined as Voronoi cells around $K$ randomly chosen centers in $S_0$. Here we compute $\mathcal{T}_h^{TV}$ and use thresholds estimated by bootstrapping over $S_0$.

- **Density Tree**: A binary tree aiming at approximating $\phi_0$, where splits are defined by a maximum information-gain criterion, in a similar fashion to random density trees like [97]. We use $\mathcal{T}_h^{TV}$ with thresholds empirically computed by bootstrap over $S_0$.

- **Parametric**: in the synthetic experiments we consider also an hypothesis test based on a parametric density model. In particular, we fit a Gaussian density on $S_0$, compute the log-likelihood [14, 15] of each incoming batch $W$, and detect changes by means of the $t$-test. Since this method exploits the true density model, it has to be considered as an ideal reference.

All the methods are configured and tested on the same $S_0$ and tested on the same batches $W$. We perform a PCA transformation, estimated from $S_0$, to all the methods based on trees as density models. We have in fact experienced that this improves the change-detection performance, since it aligns the coordinate axes – along which splits are performed – with the principal components that become parallel to the bin boundaries.

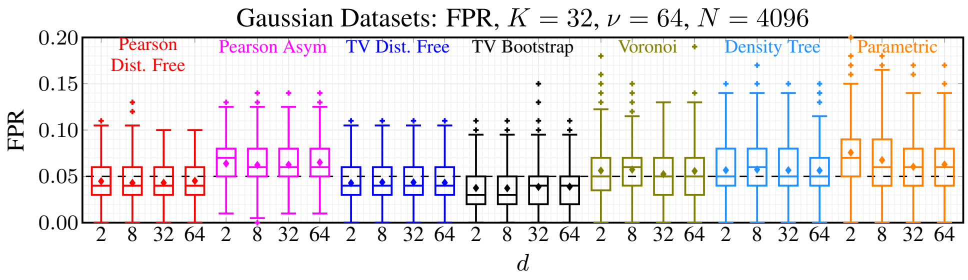
### 6.5.3 Performance Measures

We assess the change detection performance by empirically measuring, for each change $\phi_0 \to \phi_1$ the statistical power (or true positive rate, TPR) of the test and the FPR, where thresholds have been set as described in Section 6.4.3 to yield a target value $\alpha = 0.05$. Both the FPR and the TPR are computed considering 100 batches $W$ with and without changes, respectively. Since both the FPR and the TPR depend on the threshold $\delta$, which is subject to estimation errors, we build the Receiver Operating Characteristic (ROC) curve by plotting the measured FPR and TPR for different values of the threshold. We then measure the Area Under the ROC Curve (AUC) to get a single figure of merit of the change-detection performance, which does not depend on $\delta$. We recall that the AUC ranges in $[0, 1]$, and that higher values indicate better performance.

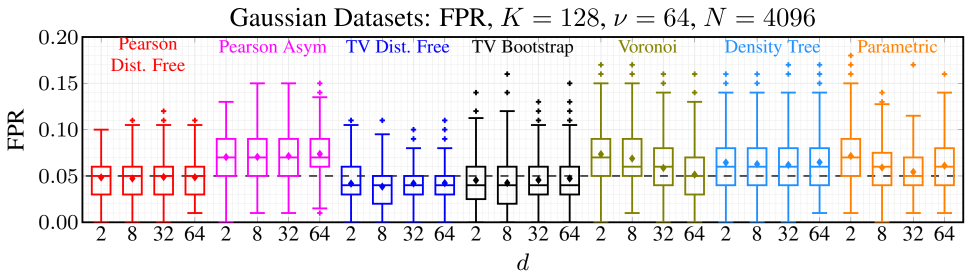
### 6.5.4 Experiments using Individual Histograms

In these experiments we focus on the evaluation of change-detection algorithms based on a individual histograms. We consider a *small* configuration, where $N = 4096$ and $\nu = 64$, and a *large* configuration, where $N = 16384$ and $\nu = 256$. Both configurations have been tested with a number of bins $K = 32$ and $K = 128$, leading to 4 different combinations $(N, \nu, K)$. Here we show the results obtained using the small configurations. The results on the large configuration are qualitatively similar and are reported in Appendix A.

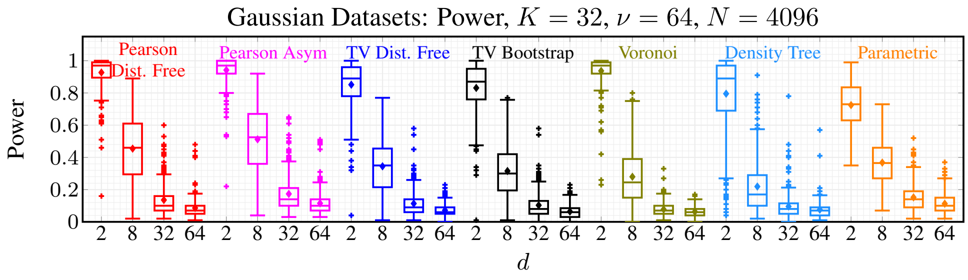
Figure 6.3 and Figure 6.4 shows the FPR and the power of all the methods in synthetic and real world datasets, respectively. In particular, Figures 6.3(a-b) and Figures 6.4(a-b) confirm that QuantTree effectively controls the FPR, for both the Pearson and total variation statistics, which is very important in change-detection. The peculiar QuantTree construction and Algorithm 6.2 provide very accurate thresholds resulting in FPR below the reference value $\alpha = 0.05$. Moreover, even if histograms defined by QuantTree feature a small number of bins, they are able to effectively monitor high-dimensional datastreams. In case of large configuration, the



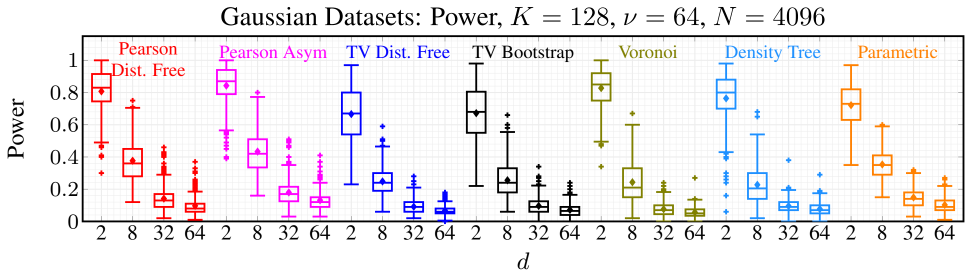
(a)



(b)



(c)



(e)

**Figure 6.3:** *Results on synthetic (a)-(d) datasets using the small configuration. In (a) and (b) we report the FPR computed on the Gaussian datasets using $K = 32$ and $K = 128$ bins, respectively, while (c) and (d) reports the corresponding powers. Thresholds computed by Algorithm 6.2 successfully yield averaged FPR values smaller than the desired value $\alpha$ disregarding the data dimension d, as expected by Theorem 6.1. Moreover, histograms computed QuantTree yield the highest power.*

histograms are better estimated, and this yield a larger power for all methods.

The FPRs of the total variation statistic are typically lower than others: this is due to the discrete nature of the statistics, which affects both testing and quantile estimation. The same problem occurs, but to a lesser extent, in the Pearson statistic, since the expression (6.5) contains a square that allows this statistic to assume a larger number of distinct values. Clearly, increasing $K$ attenuates this problem, bringing the FPR closer to $\alpha$. Thresholds used in the traditional Pearson test achieve larger FPR values, as the number of training samples in each bin is too low for the asymptotic approximation to hold: in the large configuration (see Appendix A), the problem attenuates. Since the likelihood values do not follow a Gaussian distribution, the FPR are not properly controlled in the $t$-test of the Parametric method either. In all these tests, smaller values of $K$ provide a better control over FPR, since the number of samples in each bin is larger.

Concerning the power, Figures 6.3(c-d) and Figures 6.4 show a clear decay when $d$ increases: this is consistent with the detectability loss. In general, all the methods on Synthetic datasets achieve satisfactory performance, and uniform histograms obtained through the QuantTree appear a better choice than Density Tree and Voronoi. There are minor differences among methods based on QuantTree which are nevertheless consistent with the FPR in Figures 6.3(a-b) and 6.4(a-b). Uniform density histograms outperforms others on real world datasets, see Figures 6.4(c-d), indicating that their partitioning scheme is better at detecting changes. Obviously, increasing $N$ and $\nu$ provides superior performance (see the results reported in the supplementary materials).

### 6.5.5 Experiments on Ensemble of Histograms

We construct our ensemble using histograms computed by QuantTree and for simplicity we consider only the Pearson's statistics, that outperforms the total variation, as justified also by Theorem 14.3.2 in [64]. We design two types of experiments: *exhaustive tests*, where we consider many possible combinations of number of individuals $R$ and number of bins for each individuals $K$, and *constrained budget tests*, where the overall number of bins in the ensembles is set to a constant. As in the previous experiments, we consider a *small* and *large* configurations, where the training set size is $N = 4096$ and $N = 16384$, while each batch $W$ contains $\nu = 64$ and $\nu = 256$ samples, respectively. In what follows we report the results obtained on the small configuration, while those on the large configuration are

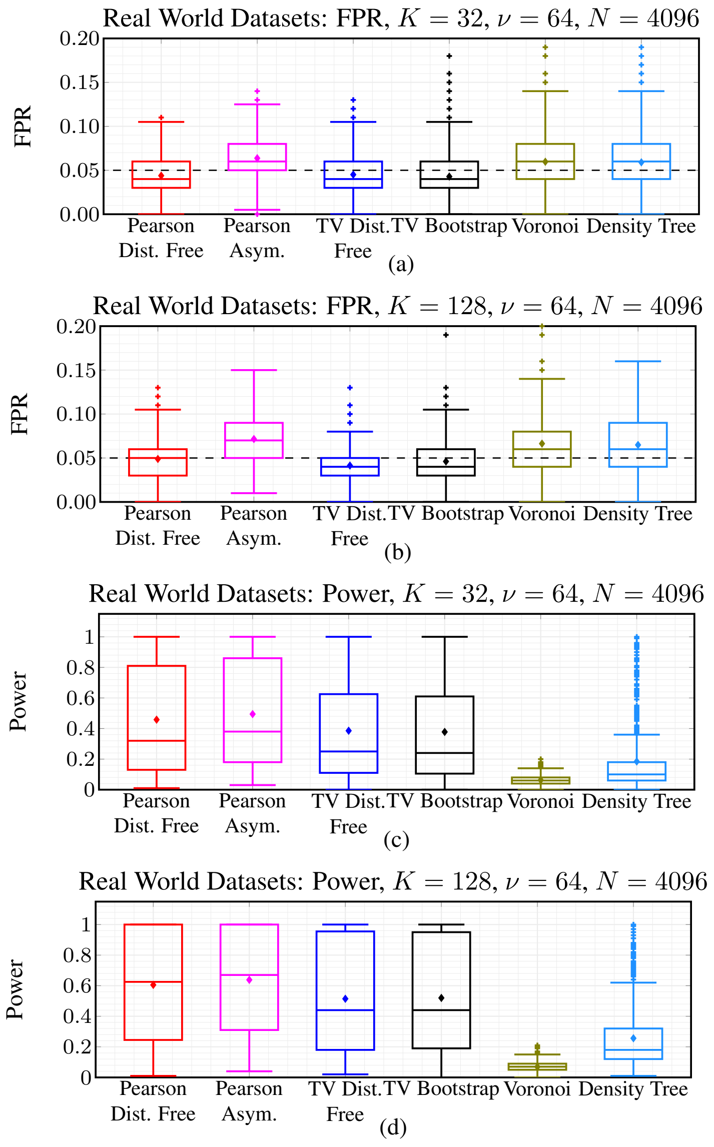


**Figure 6.4:** *Results on real world datasets using the* small $TR$ *configuration. The FPR values are shown in (a) and (b), for* $K = 32$ *and* $K = 128$*, respectively, and the powers are reported in (c) and (d). Also in this cases Algorithm 6.2 provides thresholds that successfully control the FPR and, as on Gaussian datasets, methods based on uniform histograms outperform the other in terms of power.*

reported in Appendix A.

**Exhaustive Tests.** We test ensembles of histograms by considering all the possible combinations of $R \in \{1, 2, 4, 8, 16, 32\}$ and $K \in \{2, 4, 8, 16, 32, 64\}$. For each ensemble, we compute the average AUC over the 200 changes $\phi_0 \to \phi_1$, and display these values in Figures 6.5 for different dimensions $d$ of the Gaussian dataset.

Max aggregation achieved AUC results that qualitatively are in line with the average aggregation, even though slightly less performing. Moreover, the performance improves for when $N = 16384$ since more training points are used to construct histograms.

**Constrained Budget Tests.** This experiment was designed to determine whether advantages of ensembles are merely due to the larger number of bins with respect to an individual histogram, or indicate a superior detection performance. We thus constrain all the ensembles to use the same number of bins $R \cdot K = 64$ (i.e., the budget). Since the budget is somehow related to the operations to be performed on each batch $W$, this test enables a fair comparison among ensembles and single histograms. We investigate both the performance in terms of AUC, as well as in terms of Power and FPR.

While the average AUC values computed at constant $R \cdot K = 64$ settings are along the upper-left to lower-right diagonals of images in Figure 6.5, to ease the comparison and portray their distribution, we draw the boxplots of the AUC values in Figures 6.6 and 6.7.

For the real-world datasets, we report the AUC from average aggregation for all the types of histograms in both the small and large settings, see Figures 6.4. To determine whether ensembles are advantageous or not, we perform a statistical test to assess whether, for each dimension $d$ and for each considered type of histogram, differences in the AUC of different ensemble configurations (i.e., values of $R$) are statistically significant or not. To rigorously compare multiple methods over multiple datasets, we follow the approach in [98] and perform *i*) a preliminary Friedman test to determine whether differences among the tested ensembles are statistically significant and, *ii*) a Nemenyi test in a *post hoc* analysis to rank the different ensembles according to their performance. More specifically, when the Friedman test is rejected (being the null hypothesis that there is no performance difference among the methods), we assign each ensemble to a performance group A, B or C. Group A contains the best ensembles whose performances are all comparable and all statistically different with respect to group C, which contains the worst ones, while group B collects ensembles that are not statistically different from all the elements in A or C, and thus cannot be ascribed to those groups. Performance groups are reported

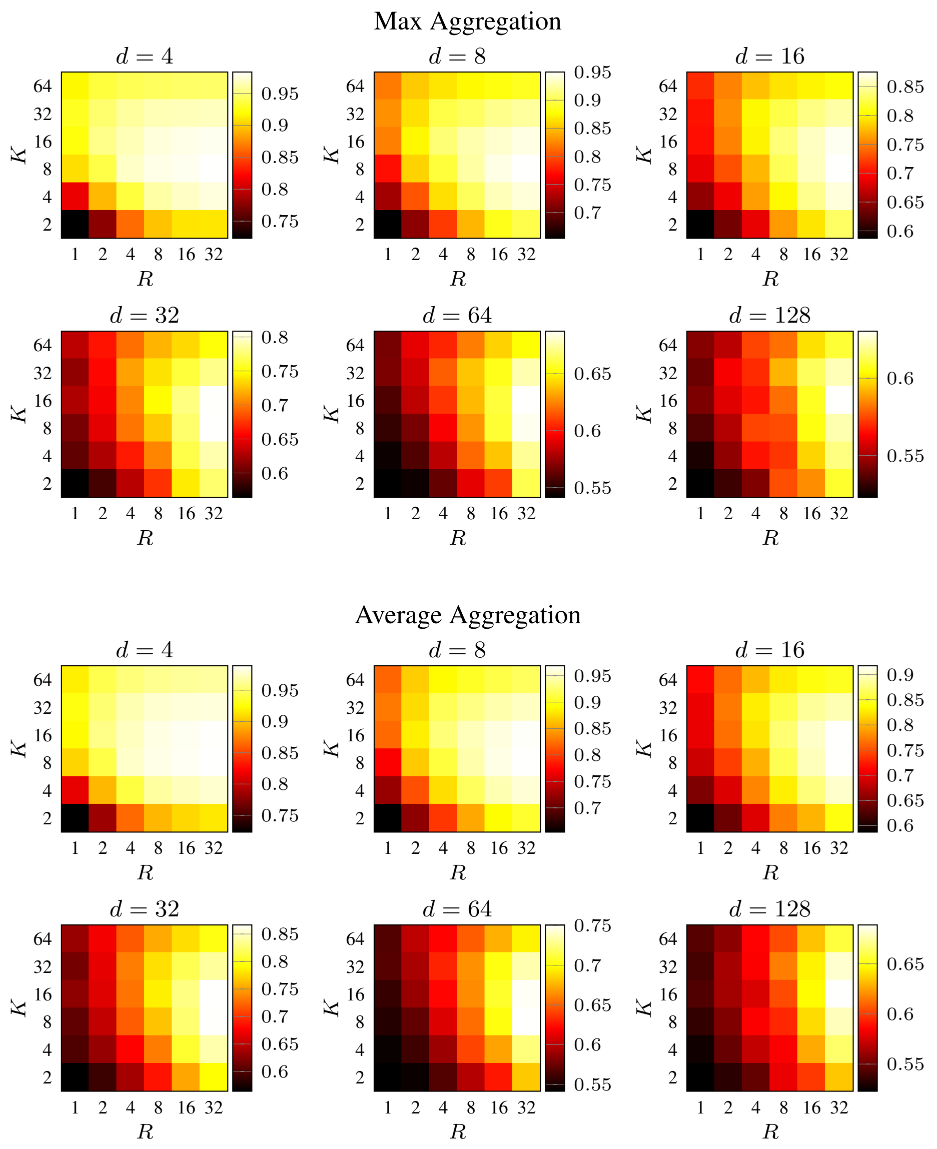
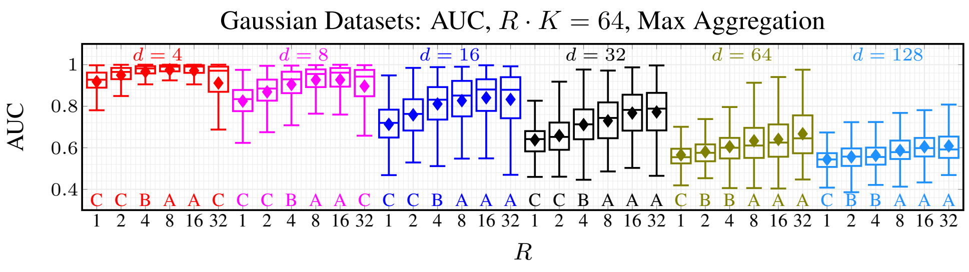
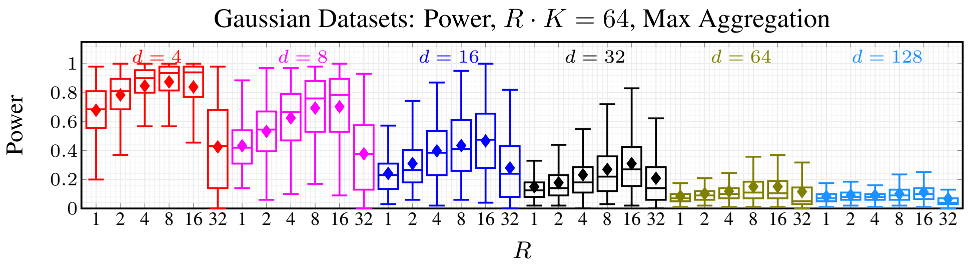


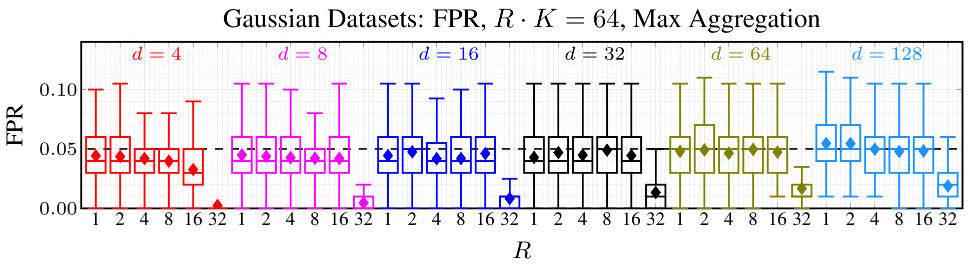
**Figure 6.5:** *Results of the exhaustive tests over a mesh of $(R, K)$ pairs for the synthetic datasets, suing the small configuration. We report the AUC obtained using both the Max Aggregation (top figures) and the Average Aggregation (bottom figures) schemes for the tested dimension d. Lighter squares correspond to better AUC performance. For a given value of $K$, increasing $R$ always improves the AUC, confirming the advantage of the ensemble approach, and this effect is more evident as d grows. However, for any level of $R$ there is an optimal number of $K$ fo histograms, after which the AUC deteriorates, meaning that the bins contain too few points to correctly estimate the associated probabilities.*



(a)



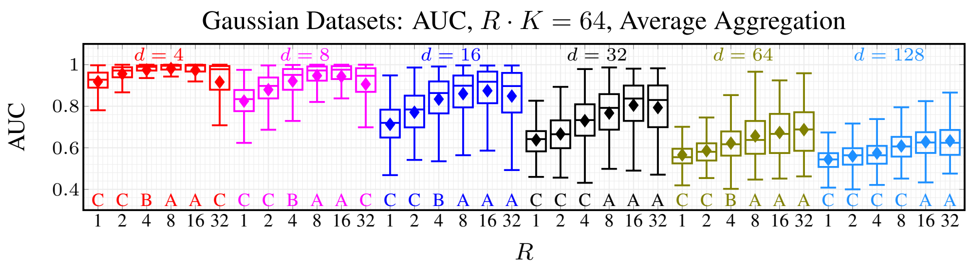
(b)



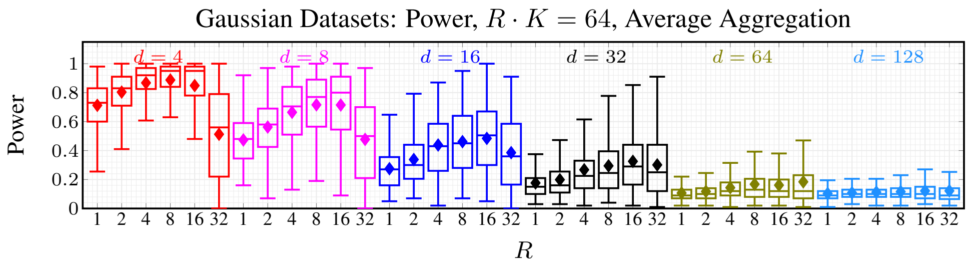
(c)

**Figure 6.6:** *Results on the constrained budget tests for the ensemble of histograms with the synthetic datasets in all the tested dimension d, using the small configuration and the Max Aggregation scheme. Plot (a) report the AUC, together with the labels that illustrate the results of the test for multiple comparisons described in Section 6.5.5. In (b) and (c) the power and the FPR are presented. The AUC and power plots indicate that it is better to spend the budget of total bins on ensemble of histograms with fewer but more populated bins, with respect to a single histograms made of many small bins ($R = 1$). However, increasing $K$ too much arms the performance, since it leads to bins having too few points. The FPR plot shot that the Bonferroni correction guarantees the desired FPR, as the number of individuals grows it becomes over-conservative and the FPR is too small.*

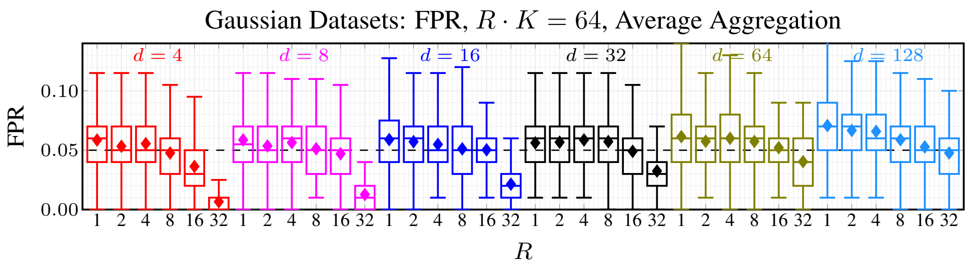
below the boxplots in Figures 6.6(a) and 6.7(a). We further investigate how the ensemble performance varies in terms of test power and FPR, when considering the two aggregation strategies proposed in Section 6.4.2. These are reported in Figures 6.6(b-c), Figures 6.7 (b-c), Figure 6.8 (b-c). It is impor-



(d)



(e)



(f)

**Figure 6.7:** *Results on the constrained budget tests for the ensemble of histograms with the synthetic datasets in all the tested dimension d, using the small configuration and the Max Aggregation scheme. Plots (a) and (b) report the AUC and the power and are qualitatively similar to the plots obtained using the Max Aggregation scheme (see Figure 6.6). (c) FPR plot points out that it is difficult to find threshold for the aggregated statistic that guarantee the desired FPR.*

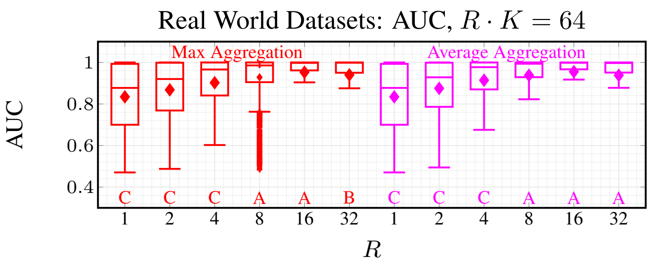
tant to determine whether ensembles can reliably control the FPR to the target level, which in our case is set to $\alpha = 0.05$. In particular, we pursue the approaches described in Section 6.4.3, and set thresholds of the statistic $\mathcal{T}_{\mathcal{E},\text{avg}}$ by bootstrapping $S_0$. Thresholds for the statistic $\mathcal{T}_{\mathcal{E},\max}$ have been set by Bonferroni correction.

## 6.6 Discussions

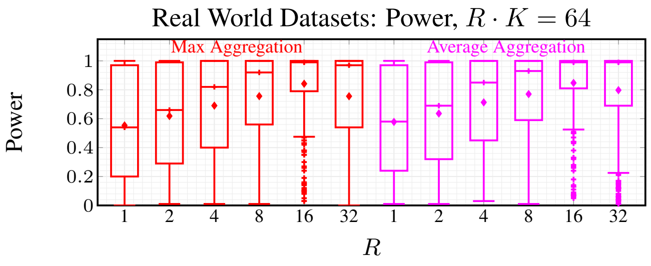
Our experiments show the effectiveness of QuantTree to compute histograms for change detection. In particular, our thresholds (estimated using samples drawn from a univariate uniform distribution) enable a better control of the FPR than asymptotic ones or those estimated by bootstrap, which is no longer necessary when using such histograms. Moreover, the boxplots for the real datasets in Figure 6.4 indicate that the histograms computed using QuantTree are the most effective in the high-dimensional settings, in terms both of power and AUC.

Our experiments show that, ensembles of histograms are beneficial for change detection purposes, and in particular, they mitigate the detectability loss phenomenon. In fact, by comparing results on small and large configurations (reported in Appendix A) we see that the performance gap between ensembles and individual histograms is larger when few training samples are provided. Experiments on constrained budget further indicate that, assuming a maximum number of bins $R \cdot K$ is available, it is often convenient to adopt multiple histograms having a few bins.

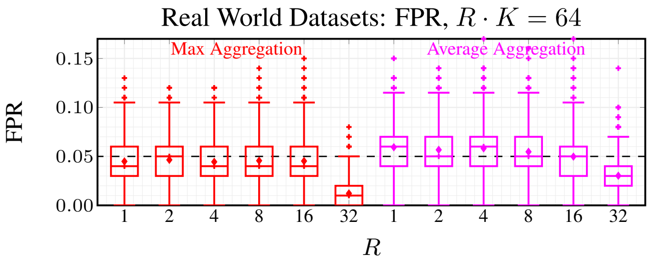
The main drawback of ensembles of histograms is that it becomes more difficult to set $\delta_{\mathcal{E}}$ in (6.3), either by bootstrap or Bonferroni correction. In fact, large ensembles might contain very similar histograms, thus the statistics $\{\mathcal{T}_{h_i}\}_i$ can be highly correlated, making the ensemble less effective. In particular, in such cases, the Bonferroni correction becomes over-conservative and the FPR becomes too small for large values of $d$ (see, for instance, Figures 6.6(c) and 6.7(c)). Yet, it is also worth remarking that, in all these plots, for a suitable value of $R > 1$, the FPR values of the ensemble matches the target values $\alpha = 0.05$.



(a)



(b)



(c)

**Figure 6.8:** *Results on the constrained budget tests with the real world datasets, using the small configuration and $R \cdot K = 64$ using both Max and Aggregation schemes. Plot (a) reports together with the labels that illustrate the results of the test for multiple comparisons described in Section 6.5.5. Plots (b) and (c) report the power and the FPR boxplots for the same above-defined settings. The plots show similar behavior w.r.t. the synthetic datasets in Figures 6.6 and 6.7.*

# Part II

# Complex Data

# CHAPTER 7

# Related Literature

In this chapter we presents an overview of the algorithm designed to handle complex data in the scenarios we considered. At first we focus on algorithms designed to learn meaningful representations for data (Section 7.1), then in Section 7.2 we summarize transfer learning algorithms that are mostly related to our modeling assumption, namely that data can be described by a dictionary yielding sparse representations. Finally, we presents the main algorithms that address the specific real world problems we considered in the thesis, i.e. anomaly detection in images (Section 7.3) and ECG monitoring (Section 7.4).

## 7.1 Learning Representations for Complex Data

Complex data such as signal and images are expected to live close to a low-dimensional manifold embedded in a high-dimensional space [99]. To successfully perform any task on this king of data, e.g., anomaly detection and classification, it is necessary to learn meaningful representations that allows to extract useful feature. These representations can be learned in an unsupervised manner to provide a parametrization of the low-dimensional manifold where data live, or in a supervised framework [100], where representation are learned to solve a specific task, e.g., classification. Our focus is on unsupervised learning algorithms, as these allow to address the detection problems we consider.

The simplest unsupervised algorithm to learn representations is probably Principal Component Analysis, that assumes that the low-dimensional manifold is a linear subspace. This assumption is hardly met in practice, since the correlations between data components are typically nonlinear. An extension of PCA that is able to handle nonlinear correlation is Kernel PCA [101], that uses kernel trick to map data on a higher dimensional space, where correlations become linear. Other approaches are based on the computation of the $k$-nearest neighbor graph over training data. Among these some compute a low-dimensional embedding of this graph that preserves certain properties of the neighborhood [102,103], while spectral methods diagonalize the Laplacian of this graph to capture nonlinearities [104].

As described in Chapter 2, we model complex data using dictionaries yielding sparse representations, i.e., we assume that data can be approximate as a sparse linear combination of atoms in a dictionary. These models approximate the nonlinear manifold where data live as a union of low-dimensional subspace.

Sparse representations w.r.t. an orthonormal dictionary have successful history in signal and image processing, stretching back to the classical wavelet shrinkage denoising technique [105]. More recently, dictionary learning algorithm have been proposed [106] to learn more powerful representation than dictionary corresponding to fixed transforms, such as the Discrete Cosine Transform and Wavelet Transform. The first data-driven dictionaries were learned in an unsupervised manner [106, 107] to address image processing tasks such as denoising [106, 108], inpainting [109] and demosaicking [110]. Recently, several method have been proposed to learn dictionary specifically designed to solve a specific task [111], in particular classification [112, 113].

Various attempts to extend sparse representations have been made in the literature, in particular to compute representations that are multiscale [114–118] or translation-invariant [119]. Among these, we cite Convolutional Sparse Representations, that are translation invariant and directly support the use of multiscale dictionaries [120]. These representations have recently begun to attract attention for solving image restoration problems [121–125], however their properties are still not thoroughly understood. In particular, white noise denoising, arguably the simplest image restoration problem, has never been addressed using Convolutional sparse representations.

## 7.2 Transfer learning

Domain adaptation can be seen as a particular instance of transfer learning [126], which have been widely investigated in the machine learning literature as a way to mitigate the performance degradation that algorithms trained in the source domain would encounter when testing data from a different, target domain. The typical scenario where these methods are used is the classification: the goal is to learn a classifier from labeled data in the source domain that is able to operate also on the a target domain. To this purpose, the classifier is adapted using a set of unlabeled data in the target domain [127, 128], and in some cases also additional labeled data in the target domain [129]. Other approaches instead assume that few labeled data are available only in the target domain, while a lot of unlabeled can be collected in the source domain [130]. An excellent categorization of transfer learning algorithms have been proposed in [126]. Here, we focus on the transductive transfer learning (also known as domain adaptation), where labeled data are available only in the source domain.

Transfer learning techniques specifically designed for dictionaries yielding sparse representation have been also widely investigated, mainly to address the image classification tasks [131–134], and go under the name of *dictionary adaptation*. In this scenario training images are acquired under different conditions than the test ones, e.g. different lightning and view angles, and therefore live in a different domain. The framework in [131] transforms dictionaries while maintaining a domain-invariant sparse representation of the data. In [132] dictionary adaptation is performed by learning a sequence of intermediate dictionaries to gradually adapt source to target data, while in [134] a shared discriminative dictionary is learned to provide group-sparse representations to both source and target domain data. Finally, [133] performs dictionary adaptation by learning representations for both source and target data in a common low-dimensional subspace. The orthogonal projections from source and target domains to the common subspace are jointly learned with a dictionary yielding sparse representations of the projected data.

## 7.3 Anomaly Detection in Images

Algorithms for detecting anomalies [135, 136] in images can be categorized w.r.t. the model adopted to characterize normal data [137]. A first class of algorithms does not use data-driven models, but adopt a probabilist model to describe raw patches, such as Gaussian distribution [138] or Gaussian mixture [139], or low-dimensional hand-crafted features extracted from the image [140].

Other algorithms are based on the non-local self-similarity prior, that has been shown to be very effective in natural images [141]. In practice, a small patch is detected as anomalous if the distance between other patches in its neighborhood [142] or in a reference image [143] is above a threshold. The distance can be defined either on raw patches or feature extracted from them.

In the last few years, generative models [144, 145] have shown impressive performance in several computer vision tasks, such as inpainting [146], video prediction [147, 148] and style transfer [149]. These model allows to learn the low-dimensional manifold where complex data live and perform sampling directly on it. Anomaly detection using generative models have been performed by computing the distance between each new sample and the learned manifold [150], where the distance is computed by solving an optimization problem. In [151] a variational autoencoder is used to estimate the probability of test data to be drawn from the probability distribution generating normal data.

Finally, sparse representations have been successfully for anomaly detection purposes. In particular, in [152] a specific sparse coding procedure is proposed to computed an anomaly score together with the sparse representation w.r.t. a dictionary learned on normal data. Other solutions monitor instead the value of the functional minimized during the sparse coding to detect unusual events in video sequences [153, 154].

## 7.4 ECG Monitoring

Since the introduction of Holter devices in 1940s, cardiac monitoring has helped physicians to determine whether users are experiencing anomalous heartbeats. Over the past few years, ECG monitoring devices have evolved from large, wired systems, as the original Holter, to small, wire-free wearables that allow users to perform everyday activities with minimal disturbance. Most of ECG monitoring devices typically do not implement on board anomaly-detection and heartbeat-classification functionalities, and ECG signal are sensed, processed and transferred to caregivers that remotely monitor the users [155, 156]. The challenge we address here is to integrate anomaly-detection capabilities directly on a low-power wearable device, as this could raise timely alarms and prevent massive data-transfer that would reduce the device battery lifetime.

Pattern recognition and machine learning techniques have been widely exploited for anomaly-detection and heartbeat-classification purposes. A first class of solutions addressing these tasks are the so called feature-driven methods [157, 158], which exploit hand-crafted morphological features that mimic the criteria used by clinicians to analyze ECG signals. Typical examples of features are the RR interval, namely the distance between two consecutive R-peaks, the amplitude and the width of QRS complex, as well as shape descriptors for the local waveforms like P-wave, T-wave, and ST-segment. Other features can be extracted from the vectorcardiogram [159], computed through a linear transformation of the 12 leads ECG, or in wavelet domain [160] or through Hermite transform [161]. In the last few years, feature-driven approaches are being replaced or combined with data-driven methods, which do not reproduce any clinical criterion, but are directly learned from training data. Data-driven methods typical leverage a model yielding meaningful representations of the ECG signals [5,22,23,162], and often refer to the time series literature [163–166] and detect anomalies by monitoring the prediction error. Others resort to clustering [22, 167] or Gaussian mixture Models [162, 163] to describe normal heartbeats. The main drawbacks of these methods is that they process the global ECG signal offline, and are not suitable for online monitoring. Recently, deep neural networks have shown very good performance in heartbeats classification [23, 168] and the 1-dimensional Convolutional Neural Network (CNN) in [23] reaches an accuracy of 98.6% on 24 recordings of the MIT-BIH database. The deep Long Short Term Memory (LSTM) network in [169] detects anomalies in time series where the pattern duration is unknown. Unfortunately, the computational requirements of deep neural networks are not compatible with the limited resources available on most wearable device: the CNN in [23] classifies a single heartbeat in few milliseconds on a 2.4 GHz CPU with 16 GB of RAM, while the CPU frequency of most wearable devices is in the order of tens of MHz.

As discussed in Chapter 2, to successfully perform long-term ECG monitoring it is necessary to adapt the learned model to track heart-rate variations. This problem has been so far ignored in ECG monitoring, including [170, 171], which are examples of ECG monitoring algorithms meant for wearable devices.

CHAPTER 8

# Sparse Representations for Anomaly Detection

In this chapter we address the problems of anomaly detection when data are signals characterized by a complex structure. In Section 8.1 we introduce dictionaries yielding representations to model such data in Section 8.1 and we present our anomaly-detection algorithm in Section 8.2. Then we customize this algorithm to address the problem of detecting defects in SEM images (8.3) and monitoring ECG signals (Section 8.4).

## 8.1 Modeling Normal Signals

Our modeling assumption is that normal signals $\mathrm{s} \sim \mathcal{P}_N$ can be well approximated by the following linear model

$$\mathrm{s} \approx D\mathbf{x}, \tag{8.1}$$

where $D \in \mathbb{R}^{d \times n}$ is a matrix called *dictionary* and the coefficient vector $\mathbf{x} \in \mathbb{R}^n$ is *sparse* [18]. Sparsity means that $\mathbf{x} \in \mathbb{R}^n$ has few of nonzero components, thus in practice that the $\ell^0$ "norm" $\|\mathbf{x}\|_0$, which is the number of nonzero components of $\mathbf{x}$, is small. Although $\|\cdot\|_0$ is not properly a norm, since it does not satisfy the homogeneity property, we adopt this notations as it is customary in the literature [18]. The sparsity of the coefficient vector $\mathbf{x}$ implies that each normal signal $\mathbf{s}$ can be well approximated by a linear combinations of few columns (*atoms*) of $D$. This is equivalent to assume that $\mathbf{s}$ lives close to union of low-dimensional subspaces $\mathfrak{D}$, where each subspace is generated by few atoms of $D$.

The coefficients of the sparse representation $\mathbf{x}$ of the signal $\mathbf{s}$ are computed by solving the *sparse coding* problem, that can be formulated in several ways, by pursuing different norms to assess the sparsity of the coefficient vector $\mathbf{x}$. The first formulation we consider is the following:

$$\begin{aligned} \mathbf{x} = \underset{\widetilde{\mathbf{x}} \in \mathbb{R}^n}{\arg\min} & \ \|\mathbf{s} - D\widetilde{\mathbf{x}}\|_2 \\ \text{such that} & \ \|\widetilde{\mathbf{x}}\|_0 \leq \kappa, \end{aligned} \tag{8.2}$$

where $\kappa$ denotes the maximum number of nonzero coefficients allowed in the representations. The problem (8.2) is NP-Hard and suboptimal solutions are computed by greedy algorithms, such as the Orthogonal Matching Pursuit (OMP) algorithm [172].

Another formulation of the sparse coding problem is based on the $\ell^1$ norm of the coefficient vector, that is known to promote sparsity [105, 173], and is known as Basis Pursuit DeNoising (BPDN) in the literature:

$$\mathbf{x} = \underset{\widetilde{\mathbf{x}} \in \mathbb{R}^n}{\arg\min} \frac{1}{2}\|\mathbf{s} - D\mathbf{x}\|_2^2 + \lambda\|\mathbf{x}\|_1, \tag{8.3}$$

where $\|\mathbf{x}\|_1 = \sum_{i=1}^n |x_i|$ promotes the sparsity of $\mathbf{x}$ while the first term assess the reconstruction error that we commit when recovering $\mathbf{s}$ using its sparse representation $\mathbf{x}$ and the dictionary $D$. Problem (8.3) is convex, thus its solution can be efficiently computed via convex optimization algorithms, such as the Alternating Direction Method of Multipliers (ADMM) [174] and Least Angle Regression (LARS) [175].

If the parameters $\kappa$ and $\lambda$ are properly chose and $D$ satisfy some hypotheses [176], the two sparse coding problems are equivalent, namely they have the same solution $\mathbf{x}$. The advantage of solving (8.3) instead of (8.2) is that the former is convex, and the solution can be found using fast algorithms. However, the equivalence between the two problems does not hold in practice, and we choose the sparse coding formulation depending on the application scenario.

The OMP, that addresses problem (8.2), is computationally cheap, especially for small $\kappa$, that is also the number of iterations required to compute the suboptimal solution. Moreover, when $D \in \mathbb{R}^{d \times n}$ is undercomplete, i.e., $d < n$ the OMP can be further accelerated, as we will show in Section 8.4. For these reason, we adopt (8.2) in ECG monitoring, since the anomaly-detection algorithm has to be run on a wearable devices having limited computational capabilities.

In contrast, problem (8.3) is computationally more demanding, but its unconstrained formulation allows the computation of more powerful indicators for anomaly detection, as described in Section 8.2. Therefore, we adopt (8.3) in the defect detection in SEM images, where it is possible to use more powerful hardware than in ECG monitoring.

Since the dictionary $D$ is typically unknown, it has to be learned from a training set of normal signals drawn from $\mathcal{P}_N$ that we collect column-wise in the matrix $S_0 \in \mathbb{R}^{d\times m}$. The *dictionary learning* problem actually corresponds to learn both the dictionary $D \in \mathbb{R}^{d\times n}$ and the sparse representations $X \in \mathbb{R}^{n\times m}$ for all the signals in the training set. In practice, we consider two formulations of dictionary learning, each one corresponding to a sparse coding problem. The first formulation pursues the $\ell^0$ norm:

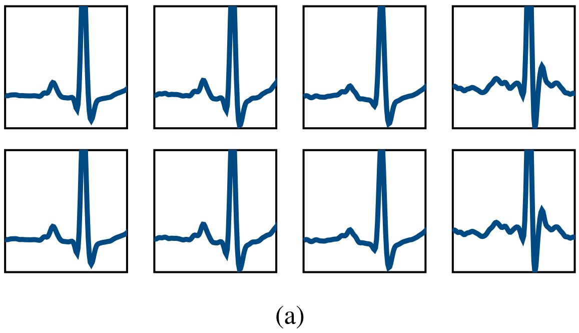
$$\begin{aligned} D, X = & \underset{\widetilde{D}\in\mathbb{R}^{d\times n}, \widetilde{X}\in\mathbb{R}^{n\times m}}{\arg\min} \quad \|S_0 - \widetilde{D}\widetilde{X}\|_2, \\ & \text{such that } \|\mathbf{x}_i\|_0 \leq \kappa, \quad i \in 1, \dots, n, \end{aligned} \tag{8.4}$$

where the sparsity constraint applies to each column of the matrix $X \in \mathbb{R}^{n\times m}$, which stacks the coefficient vectors of all the signals in $S_0$. Here, we consider the K-SVD algorithm [106] to address (8.4), an iterative algorithm that alternates the optimization over $\widetilde{X}$ by keeping fixed the dictionary $\widetilde{D}$ and the optimization over $\widetilde{D}$ by keeping fixed the coefficient matrix $\widetilde{X}$. The optimization over $\widetilde{X}$ is performed using the OMP algorithm, while the optimization over $D$ is addressed by means of a specifically designed procedure that exploits the sparsity of $\widetilde{X}$.

The second formulation is the dictionary learning Basis Pursuit DeNoising problem, that is based on the $\ell^1$ norm:

$$D, X = \underset{\widetilde{D}\in\mathbb{R}^{d\times n}, \widetilde{X}\in\mathbb{R}^{n\times m},}{\arg\min} \frac{1}{2}\|S_0 - \widetilde{D}\widetilde{X}\|_2^2 + \lambda\|X\|_1, \tag{8.5}$$

where $\|\widetilde{X}\|_1$ denoted the sum of the $\ell^1$ norm of the columns of $\widetilde{X}$. Problem (8.5) is not convex, but a local minimum can be computed using the ADMM algorithm, that also alternates the calculation of the dictionary atoms and the sparse representations. In our experiments we use the MATLAB implementation of the ADMM provided in the SPORCO library [177].



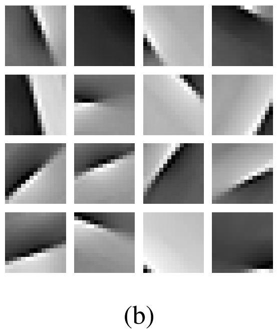


**Figure 8.1:** *(a) Atoms of a dictionary learned from normal heartbeats of a user by the KSVD algorithm. (b) Atoms of the dictionary learned by the ADMM algorithm from* $15 \times 15$ *patches extracted from normal SEM images.*

Figure 8.1 shows two example of learned dictionaries in the two considered scenario. We use the dictionary learning formulation corresponding to the sparse coding used in each application, thus the dictionary in (a) is learned by solving (8.4), while the one in (b) is learned by solving (8.5).

## 8.2 Detecting Anomalous Signals

To detect whether $\mathbf{s} \in \mathbb{R}^d$ is anomalous or not, we determine if it conforms or not to the learned dictionary $D$. To this purpose, we pursue the sparse representation $\mathbf{x}$ of $\mathbf{s}$ w.r.t. $D$ to compute a low dimensional indicator vector $\mathbf{f}(\mathbf{s}) \in \mathbb{R}^p$, and adopt a decision rule over $\mathbf{f}$:

$$\mathbf{s} \text{ is anomalous} \quad \Longleftrightarrow \quad \mathcal{T}(\mathbf{f}) > \gamma, \tag{8.6}$$

where $\mathcal{T} \colon \mathbb{R}^p \to \mathbb{R}$ and $\gamma \in \mathbb{R}$ controls the FPR. To guarantee a desired FPR equal to $\alpha$, the threshold $\gamma$ can be estimated as the $1-\alpha$ sample quantile of $\mathcal{T}$ over an additional training set of normal signals. In what follows we focus on the design of both the indicator vector $\mathbf{f}$ and the decision function $\mathcal{T}$ depending on the specific sparse coding problem considered.

Let us first focus on (8.2). This problem has a clear geometrical interpretation: its solution can be used to compute $D\mathbf{x}$, that is the projection of $\mathbf{s}$ onto $\mathfrak{D}$, i.e., the union of $\kappa$-dimensional subspaces generated by the dictionary $D$. Since our modeling assumption is that normal signals live close to $\mathfrak{D}$, we consider as feature the distance between $\mathbf{s}$ and $\mathfrak{D}$, thus[1]

$$\mathbf{f}(\mathbf{s}) = [\|\mathbf{s} - D\mathbf{x}\|_2]. \tag{8.7}$$

[1]In this case $\mathbf{f}$ is a scalar and not a vector, but we will keep the vector notation for $\mathbf{f}$ for coherence with the rest of the section.

Then, we detect as anomalous any signal that is sufficiently far from $\mathfrak{D}$, i.e., we define the decision function $\mathcal{T}$ as the identity function $\mathcal{T}(\mathbf{f}) = \mathbf{f}$.

In case of (8.3) we compute a bivariate indicator vector $\mathbf{f}$ which jointly accounts for the reconstruction error and the sparsity of the representations:

$$\mathbf{f}(\mathbf{s}) = \begin{bmatrix} \|\mathbf{s} - D\mathbf{x}\|_2 \\ \|\mathbf{x}\|_1 \end{bmatrix} \tag{8.8}$$

Anomalous signal are expected to substantially deviate from normal ones in either their sparsity or reconstruction error (or possibly both). Thus, the corresponding indicator vectors would be outliers w.r.t. the distribution $\phi_0$ of indicator vectors extracted by normal signals. Here, we estimate $\phi_0$ by kernel density estimation (KDE), adopting a kernel based on linear diffusion with automatic bandwidth selection [178]. Then, a signal $\mathbf{s}$ is considered anomalous if $\mathbf{f}(\mathbf{s})$ falls in a low-density region of $\phi_0$, i.e., we define $\mathcal{T}$ as the negative log-likelihood w.r.t. $\phi_0$:

$$\mathcal{T}(\mathbf{f}) = -\log(\phi_0(\mathbf{v})). \tag{8.9}$$

## 8.3 Defect Detection in SEM Images

In this section we show how the proposed anomaly-detection algorithm can be used in the considered quality inspection system to estimate an anomaly mask $\widehat{\Omega}$ for each acquired SEM image, as described in Section 2.2.1.

At first, we need a training set of normal patches to learn the dictionary $D$ and the threshold $\gamma$. To this end, we assume that a set of defect-free images are available. These images can be easily obtained by manually cropping areas of SEM images that do not contain any defects.

Then we have to perform a preliminary preprocessing to each patch. In fact, to effectively capture the structure that characterizes normal filaments, we consider quite small patches; thus, there might be patches that do not overlap with any filament and are completely dark. Patches that are entirely zero can be perfectly reconstructed by any linear model, and achieve a (very) sparse representation, having all coefficients in (2.4) equal to zero. Unfortunately, null indicator vectors can impair the estimation of $\phi_0$, and it is safer to remove them from both the training patches. Thus, we consider for training only patches in the set $S_0$:

$$S = \{\mathbf{s} \,|\, \text{median}(\mathbf{s}) > \varepsilon\}, \tag{8.10}$$

where $\varepsilon > 0$ is a manually tuned parameter. The median in (8.10) was used to remove also dark patches that marginally overlap with a filament.

Then we split the training set $S_0$ in two subsets of patches: we learn the dictionary $D$ from the first subset using (8.5), and we estimate the threshold $\gamma$ from the second subsets as described in Section 8.2.

During test phase, we extract all the patches from each new image and preprocess them as in (8.10). It is worth mentioning that nanofibrous materials having too large holes might yield porosity values that are far from the normal ones. However, this sort of anomalies can be detected by straightforward morphological operations on the whole image and certainly do not require any learning method.

We analyze each patch suing our anomaly-detection algorithms and compute an initial estimate $\widetilde{\Omega}$ of the anomaly mask by setting

$$\widetilde{\Omega}(c) = \begin{cases} 0 & \text{if } \mathrm{s}(c) \text{ is normal} \\ 1 & \text{if } \mathrm{s}(c) \text{ is anomalous} \end{cases}, \tag{8.11}$$

where $c$ denotes the pixel and $\mathrm{s}(c)$ is the patch centered in $c$. Since patches centered in neighboring pixels largely overlap, we aggregate the decisions of the anomaly detector in all those patches that overlap with $c$. We perform such aggregation by post-processing the anomaly mask (8.11) by majority voting:

$$\widehat{\Omega}(c) = \begin{cases} 0 & \text{if } \#A_c < \#N_c \\ 1 & \text{if } \#A_c \geq \#N_c \end{cases}, \tag{8.12}$$

where $A_c = \{u \in \mathcal{U} \,|\, \widetilde{\Omega}(c+u) = 1\}$ denotes the set of pixels in $\mathrm{s}_c$ that are considered anomalous and $N_c = \{u \in \mathcal{U} \,|\, \widetilde{\Omega}(c+u) = 0\}$ the set of pixels that are considered normal.

Finally, to smooth the borders of anomalous regions in $\widehat{\Omega}$ we perform an additional post-processing by customary morphological operators [179]. More precisely, we apply an erosion followed by a dilation, which are non-linear filters based on order statistics: the minimum and the maximum over a given support, respectively. We experienced that adopting these binary operations over a neighborhood smaller than $\mathcal{U}$ can improve the coverage of anomalous regions.

## 8.4 Anomaly Detection in ECG signals

In this section, we show how our anomaly-detection algorithm can be customized to be more efficient in case of ECG monitoring. We have experienced that undercomplete dictionary $D \in \mathbb{R}^{d \times n}$, i.e., $n < d$, can successfully characterized normal heartbeats of a specific user, capturing all the variability of normal heartbeats. These settings are quite different from those traditionally used in image and signal processing, where dictionaries are redundant, i.e., $n > d$. However, as we will show in our experiments, the use of undercomplete dictionaries yields to good anomaly-detection performance.

In case of undercomplete dictionaries it is possible to optmize the OMP algorithm. This is a critical aspect for wearable devices: since the sparse-coding has to be solved for each acquired heartbeat, a reduction in the number of calculations performed can meaningfully extend the battery life of the device.

An undercomplete dictionary $D$ does not span the entire space $\mathbb{R}^d$, but only a subspace $\mathcal{U}$ having dimension $n$ that includes the union of low dimensional subspaces $\mathfrak{D}$ identified by $D$. Since solving the sparse coding (8.2) corresponds to computing the projection of s onto $\mathfrak{D}$, we can reduce the number of operations performed in the OMP by first projecting s onto $\mathcal{U}$, and then solving the sparse coding problem on the projected heartbeat. To this end, we select an orthonormal basis of $\mathcal{U}$ by computing the QR decomposition of $D = QR$, being $R \in \mathbb{R}^{n\times n}$ an upper-triangular matrix, and $Q \in \mathbb{R}^{d\times n}$ such that $Q^TQ = I_n$, where $I_n$ is the $n \times n$ identity matrix. Then, we address the following sparse-coding problem through OMP instead of (8.2)

$$\widehat{\mathbf{x}} = \arg\min_{x} \|Q^T\mathbf{s} - R\mathbf{x}\|_2^2, \text{ s.t. } \|\mathbf{x}\|_0 \le \kappa, \tag{8.13}$$

where $Q^T$s is the projection of s onto $\mathcal{U}$.

In what follows, we prove that problems (8.2) and (8.13) have the same solution. However, (8.13) is much cheaper than (8.2) since the computational complexity of OMP is determined by the target sparsity $\kappa$ and the number of atoms $n$. Solving (8.2) with a dictionary $D \in \mathbb{R}^{d\times n}$ yields a complexity $O(\kappa dn)$ [180]. In contrast, solving (8.13) (where $R \in \mathbb{R}^{n\times n}$) requires yields a complexity $O(\kappa n^2)$, thus the computational complexity of OMP is dominated by the cost of the matrix product $Q^T$s, i.e. $O(dn)^2$. In practice, we reduce the overall cost of the OMP algorithm by a factor $\kappa$, from $O(\kappa dn)$ to $O(dn)$. The following proves the equivalence of (8.2) and (8.13).

**Proposition 8.1.** *Let $Q \in \mathbb{R}^{d\times n}$ and $R \in \mathbb{R}^{n\times n}$ define the QR decomposition of the dictionary $D \in \mathbb{R}^{d\times n}$. Then, for every* $\mathbf{s} \in \mathbb{R}^d$ *and* $\mathbf{x} \in \mathbb{R}^n$ *it holds:*

$$\|\mathbf{s} - D\mathbf{x}\|_2^2 = \|Q^T\mathbf{s} - R\mathbf{x}\|_2^2 + \|\mathbf{s}\|_2^2 - \|Q^T\mathbf{s}\|_2^2. \tag{8.14}$$

[2]In our experiments we set $n = 8$ and $\kappa = 3$, while $d \approx 150$, thus $d > \kappa n$.

*Proof.* Let us first remark that the columns of $Q$ form an orthonormal basis of the $n$-dimensional subspace $\mathcal{U} \subset \mathbb{R}^d$ spanned by the columns of $D$. Since $n < d$, it is always possible to define a set of $(d-n)$ orthonormal vectors to extend $Q$ to an orthonormal basis of the entire $\mathbb{R}^d$. In particular, we define $P = [Q, Q_\perp]$, where $Q_\perp \in \mathbb{R}^{d \times n-d}$ is such that $Q_\perp^T Q = 0$ and $Q_\perp^T Q_\perp = I_{d-n}$. Since $P$ is an orthogonal matrix, for each $\mathbf{v} \in \mathbb{R}^p$ we have that:

$$\|\mathbf{v}\|_2^2 = \|P^T\mathbf{v}\|_2^2 = \|Q^T\mathbf{v}\|_2^2 + \|Q_\perp^T\mathbf{v}\|_2^2. \tag{8.15}$$

Substituting $D = QR$ in $\|\mathbf{s} - D\mathbf{x}\|_2^2$ and applying (8.15) to $\mathbf{v} = \mathbf{s} - D\mathbf{x}$ we obtain:

$$\begin{aligned}\|\mathbf{s} - D\mathbf{x}\|_2^2 &= \|\mathbf{s} - QR\mathbf{x}\|_2^2 = \|P^T(\mathbf{s} - QR\mathbf{x})\|_2^2 = \\ &= \|Q^T\mathbf{s} - Q^TQR\mathbf{x}\|_2^2 + \|Q_\perp^T\mathbf{s} - Q_\perp^TQR\mathbf{x}\|_2^2 = \\ &= \|Q^T\mathbf{s} - R\mathbf{x}\|_2^2 + \|Q_\perp^T\mathbf{s}\|_2^2,\end{aligned} \tag{8.16}$$

where the last equality holds since $Q_\perp^T Q = 0$ and $Q_\perp^T Q_\perp = I_{d-n}$. Proposition is proven by substituting $\mathbf{v} = \mathbf{s}$ in (8.15), yielding $\|Q_\perp^T\mathbf{s}\|_2^2 = \|\mathbf{s}\|_2^2 - \|Q^T\mathbf{s}\|_2^2$, and replacing this expression in (8.16). $\square$

Proposition 8.1 confirms that we can substitute the functional in (8.2) with the right-hand side of (8.14). The last two terms of (8.14) do not depend on $\mathbf{x}$ and can be ignored in the minimization, thus the problems (8.2) and (8.13) are equivalent.

## 8.5 Experiments

In this section we assess the performance of our anomaly-detection algorithm on both SEM images and ECG signals. We describe the figures of merit and the datasets used in our experiments in Sections8.5.1 and Section 8.5.2, respectively, while in Section 8.5.3 we presents the alternative solutions considered. Finally, we show our results on SEM images and ECG signals in Sections 8.5.4 and 8.5.5, respectively.

### 8.5.1 Figures of merit

To assess the anomaly detection performance in both scenarios we consider the following figures of merit:

- *False Positive Rate* (FPR), namely the percentage of normal data (pixel or heartbeats) identified as anomalous.

- *True Positive Rate* (TPR), namely the percentage of pixel or heartbeats correctly identified as anomalous.

Since both FPR and TPR depend on the threshold $\gamma > 0$ in (8.6), we consider the Receiving Operating Characteristic (ROC) curve, which are obtained by varying $\gamma$ and plotting the corresponding TPR against the FPR. An example of ROC curve is provided in Figure 8.3: the closer the curve to the point (0,1), the better. To get a quantitative assessment of the anomaly-detection performance, we measure the area under the curve (AUC), which for the ideal detector (namely the one having no false positives and no false negatives) is 1.

Moreover, in experiments on SEM images we consider an additional figure of merits. In fact, the AUC is mainly influenced by large defects, while our goal is to detect all of them disregarding their size. Therefore, to quantitatively assess the coverage of all defects, we extract the connected components [179] of the ground truth $\Omega$, thus assigning a blob to each defect. Then, we measure the *Defect Coverage* as the percentage of pixels covered by the output $\widehat{\Omega}$ of a detector yielding FPR = $0.05$. Of course, each defect yields one *Defect Coverage* value, and performance have to be evaluated by considering the distribution of *Defect Coverage* values.

### 8.5.2 Datasets

**SEM Images Dataset.** Our SEM images are acquired with the FE-SEM (Carl Zeiss Sigma NTS, Gmbh Öberkochen, Germany). A sample of $4 \times 4\ cm$ from the produced material is placed on a metallic support, and a thin gold coating of $5\ nm$ is applied on the sample surface to guarantee satisfactory electrical conduction. All images are acquired in the same conditions and with the same parameters, i.e., magnification of $8000$x, extra high tension of $5\ kV$, working distance of $7\ mm$, brightness of $45\%$, and contrast of $52\%$.

Our dataset contains 45 SEM images (dimension $1024 \times 696$ pixels): 5 images are anomaly-free, while 40 images contain anomalies of different size. For each image, we manually select all defects, defining the anomaly mask $\Omega$ that is used as a ground truth in our tests. Overall defects in these images are very small: on average they cover the $1.3\%$ of the image, and only the $0.5\%$ of the anomalies exceed the $2\%$ of the image size.

**ECG Datasets.** We consider two different datasets of ECG signal. The first one is acquied using the Pulse Device, a prototype wearable device produced by STMicroelectronics, while the second one is a benchmark datasets acquired using Holter device.

The *Pulse dataset* contains 20 ECG signals lasting from 40 minutes up to 2 hours, recorded from 10 healthy users (two signals per user). The two acquisitions from each user have been performed in different times, repositioning the Pulse Sensor such that the morphology of heartbeats changes. Due to motion artifacts or temporary device detachments, these signals sometimes contain low-quality segments, which have been discarded by an experienced cardiologist with the aid of a commercial software. While these heartbeats are not anomalous from a clinical point-of-view, we exclude them as they to not show the same morphology of others. These heartbeats could be possibly removed also by monitoring recordings from MEMS accelerometers.

The *MIT-BIH Arrhythmia Dataset* [181] contains 48 ECG signals lasting around 30 minutes each, that have been extracted from long-term Holter recordings. These segments have been selected by expert cardiologists which discarded the low-quality parts of the signal. Each ECG signal contains a few arrhythmias, and every heartbeat is provided with annotations by the cardiologists.

### 8.5.3 Alternative Solutions

The first alternative anomaly-detection algorithm we consider is [152], denoted as **Coding**. This general purpose algorithm employs dictionaries yielding sparse representations and embeds the anomaly-detection phase in a ad-hoc sparse coding procedure. More precisely, the sparse coding formulation includes an additional term $\mathbf{a}$ which gathers signals (patches or heartbeats) that cannot be sparsely represented by $D$. Anomalies are detected controlling whether the magnitude of $\mathbf{a}$ exceeds a fixed threshold. To enable a fair comparison with our anomaly detector, the two solutions use the same dictionary and have been manually configured to achieve their best performance.

In case of SEM images, we consider other four anomaly-detection solutions specifically designed for images. The first three (**Variance**, **Gradient**, **Grad&Var**) are baseline solutions that implement hand-crafted features to distinguish between normal and anomalous patches. In particular, the indicator vectors associated to these baseline solutions are suggested by the fact that defects are often flat, whereas normal regions are characterized by prominent edges (see Figure 2.1). The fourth solution (**STSIM**) is based on the structural texture similarity measure proposed in [182], which achieves state-of-the-art performance in texture classification.

Baseline solutions follow the same framework of the proposed algorithm: more precisely, during the training phase, we compute an indicator vector $\mathbf{f}(c)$ for all of the patches extracted from training images. Then, we fit the distribution $\phi_0$ on the computed indicators by KDE [178] and set a suitable threshold $\gamma$. During operations, the anomaly mask $\widetilde{\Omega}$ is computed as in (8.11). The only difference between the baseline solutions is the indicator vector $\mathbf{f}$ used:

- **Variance**: the indicator vector $\mathbf{f}(c)$ corresponds to the sample variance $v(c)$ computed over the patch $\mathbf{s}(c)$.
- **Gradient**: the indicator vector $\mathbf{f}(c)$ corresponds to $g(c)$, the average magnitude of the gradients in the patch $\mathbf{s}(c)$. More precisely, we compute at first the image of gradient magnitude $e$ as

$$e = \sqrt{(s \circledast d_x)^2 + (s \circledast d_y)^2}, \tag{8.17}$$

  where $d_x = [-1, 1]$ and $d_y = [-1; 1]$ are the horizontal and vertical derivative filters [179], respectively, and $\circledast$ denotes the 2-dimensional convolution. If we denote by $\mathbf{e}(c)$ the patch centered at $c$ extracted from $e$, then $g(c)$ is the average value of the patch $\mathbf{e}(c)$.
- **Grad&Var**: this solution stacks the indicators $v(c)$ and $g(c)$ in a two-dimensional indicator vector

$$\mathbf{f}(c) = \begin{bmatrix} g(c) \\ v(c) \end{bmatrix}. \tag{8.18}$$

The **STSIM** solution is based on structural texture similarity metric [182], which assesses the similarity between different textures. More precisely, a texture image $s$ is decomposed into steerable-filter subbands [183], and a feature vector $\mathbf{h}(c)$ is obtained by computing subband statistics over $\mathbf{s}(c)$. In [182] this is used for texture classification: each feature vector is assigned to the closest class in terms of Mahalanobis distance. In our scenario there is only one texture corresponding to normal images, and we perform anomaly detection using feature vectors as follows: during the training phase we compute the feature vectors from the training images, their mean $\overline{\mathbf{h}}$ and their covariance. Then, during operations, we compute $\mathbf{h}(c)$ for each patch and consider $\mathbf{s}(c)$ anomalous when the Mahalanobis distance between $\mathbf{h}(c)$ and $\overline{\mathbf{h}}$ exceeds a fixed threshold $\gamma$. This is equivalent to consider as anomalous any patch having an indicator falling outside a confidence region around $\overline{\mathbf{h}}$, defined by the Chebyshev inequality.

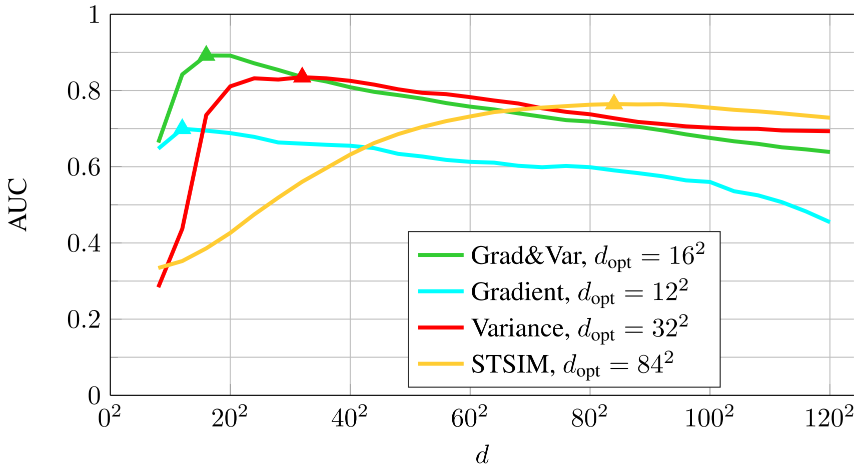


**Figure 8.2:** *Values of the AUC for the considered methods obtained by varying the patch size. The solid triangles indicate the points where the AUC is maximized and in the legend the optimal patch size* $p_{\text{opt}}$ *is reported.*

### 8.5.4 Experiments on SEM Images

All solutions have been tested in the same setting: they are trained from a set of 5 anomaly-free images, they preliminary remove dark patches as in (8.10), and the anomaly mask $\widehat{\Omega}$ is computed from the indicator vectors $\mathbf{f}(c)$ as described in Section 8.3. Of course, choosing the right patch size is very important, since small patches might not exhibit the typical structure of normal data, while large patches might prevent the detection of small anomalies. To fairly compare different methods, each one has to be tested using its optimal patch size. Therefore, we choose the best value of $p$ for baseline and STSIM solutions by testing $d \in \{4^2, 8^2, 12^2, \dots, 120^2\}$ over a validation set of 5 images containing anomalies. Figure 8.2 shows the average AUC values obtained for each solution, and reports the optimal patch sizes $d_{\text{opt}}$ that are used in our experiments. As far as the Coding and the Proposed solution are concerned, we manually set the patch size $d = 15^2$, since using larger patches would require too many training data to avoid overfitting in model (2.4) and would substantially increase the computational costs. The same 5 validation images are used to set the other parameters for all of the considered solutions, using cross-validation to maximize the AUC: $\lambda$ in (8.5) and (8.3), $\varepsilon$ in (8.10), as well as the parameters in the Coding solution. The Defect Coverage is computed by configuring the parameter $\gamma$ in each method to yield FPR = $0.05$ in these 5 validation images. Finally, these 5 images used for validation are not considered for perfor-

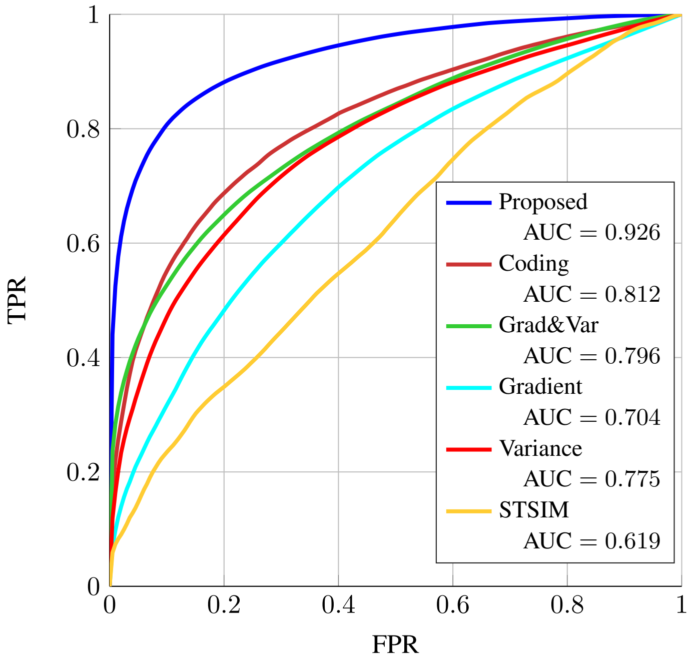


**Figure 8.3:** *ROC curves for all solutions considered in Section 8.5.3, with the corresponding AUC values reported in the legend. Proposed and Coding solutions use a patch size* $d = 15^2$*, while the others the optimal patch size* $d_{opt}$ *selected in Figure 8.2. The Proposed solution outperforms by far all of the others.*

mance assessment, thus our experiments involve the remaining 35 images.

The proposed solution is compared in two experiments against the five solutions described in Section 8.5.3. At first, we test each solution over the entire dataset, and we assess the overall anomaly-detection performance by the ROC curves averaged over 35 images. These curves are reported in Figure 8.3, together with the corresponding AUC values in the figure legend. ROC curves clearly indicate that the Proposed solution outperforms all of the others, achieving AUC values that are at least superior of $0.2$. In particular, the proposed solution outperforms the Coding, which uses the same dictionary $D$. Thus, we can conclude that (at least in this specific application) it is not convenient to embed the anomaly detection into the sparse-coding stage, while it is better to separately compute the indicators and then identify anomalies as outliers. The STSIM solution achieves the worse performance, probably because the anomalies in these images are very small and cannot be detected when using large patch sizes. However, as observed in [182] and in Figure 8.2, the performance of STSIM solu-

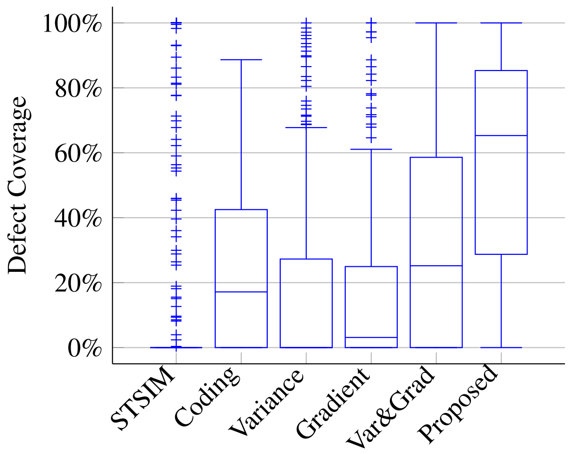


**Figure 8.4:** *Box-plots reporting the distribution of the Defect Coverage. All the considered solutions have been configured to yield at a fixed FPR* = $0.05$. *The proposed algorithm achieves the best performance, as it covers at least 50% of the anomalies for more than 60% of their area.*

tion degrades when considering smaller patches, since the local subband statistics cannot capture the texture structure.

In the second experiment we compare the Defect Coverage values of all these solutions, to make sure that the superior performance achieved by the proposed solution is not due to a superior coverage of few large defects (like the film in Figure 8.5). The box-plots in Figure 8.4 confirm that the proposed solution guarantees a Defect Coverage that is often better than others, having most of the defects covered more than the 60%. Thus, considering that small anomalies far outnumber the large ones (as described in Section 8.5.2), we can safely conclude that the proposed solution provides superior detection performance also of small defects. We also provide a visual comparison of the anomaly-detection performance. Figure 8.5 reports the masks $\widehat{\Omega}$ over three meaningful images for the three most effective solutions (according to Figures 8.3 and 8.4), generated by setting the same values $\gamma$ to compute the Defect Coverage values. These masks confirm that the proposed solution provides a superior coverage of very small anomalies, as it clearly emerges in the second image. The large film in the first image is successfully detected by all methods (and in particular by the Coding solution). However, the tiny anomalies in the second image are much better detected by the Proposed solution. Also, the Coding solution completely misses a large bead in the third image. Finally, most of the false alarms in the Proposed solution appears at junctions and pairs of filaments that are

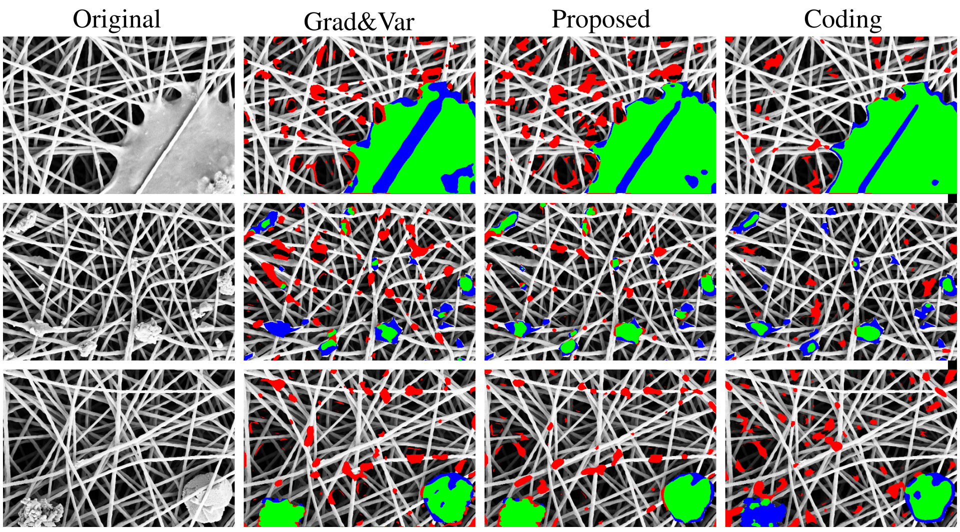


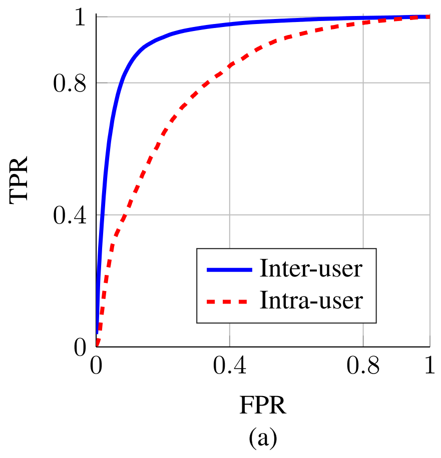
**Figure 8.5:** *Examples of anomaly detection on three meaningful SEM images. The first column reports the three original images, while the following ones present the detections obtained from Grad&Var, Proposed and Coding solutions, respectively. These solutions have been chosen as the best performing ones, according to Figures 8.3 and 8.4. The threshold $\gamma$ has been set as in Figure 8.4, to yield FPR = $0.05$. Pixels correctly identified as anomalous are marked in green, false positives in red, and false-negatives in blue.*

very close to each other (see third image), which however correspond to very few patches.

### 8.5.5 Experiments on ECG monitoring

We show that dictionaries yielding sparse representations are effective model to characterize normal heartbeats acquired using the Pulse device, and that the proposed anomaly detector successfully detects arrhythmias on the MIT-BIH Dataset

**Experiments on the Pulse Dataset.** We design two anomaly-detection experiments to detect heartbeats having a different morphology. In particular, we consider as anomalous those heartbeats acquired from a different user or from the Pulse Sensor in a different position. In practice, we use the KSVD algorithm to learn a dictionary $D$ for each of these 20 ECG signals, using a training set of 500 randomly selected heartbeats. We then consider as normal those heartbeats belonging to the same signal used to learn $D$ (namely the same pair user-position), and as anomalous heartbeats from any different signal. This procedure is performed $20 \cdot 19 = 380$ times,



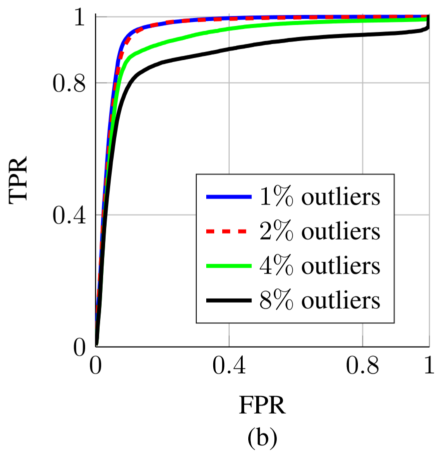


**Figure 8.6:** *ROC curves computed on Pulse dataset. (a) Inter-user and intra-user anomaly detection. This curves confirm that the morphology of the heartbeat depend on both user and position of the sensing device and the dictionaries yielding sparse representation can model normal morphology. (b) We consider the inter-user anomaly detection problem when the training set is corrupted by different percentage of outliers. The proposed algorithm can tolerate small percentage of outliers, as its performance clearly degrades when outliers reach* $8\%$ *of training data.*

one for each pair of ECG signals. Figure 8.6(a) shows the performance on *inter-user* anomalies, where normal and anomalous heartbeats come from different users. More precisely, we average ROC curve computed over the 380 combinations. Overall, the AUC value is quite large and this indicates that the considered algorithm can model the user-specific morphology and discriminate between users. In Figure 8.6(a) we also report the ROC curves on *intra-user* anomalies, where the anomalous heartbeats come from the same user having the Pulse Sensor in a different position. These curves are averaged over all the possible 20 combinations of the ECG signals. Still, the algorithm is able to successfully detect anomalies, although the AUC values are typically lower than in the inter-user case. This indicates that in this dataset, intra-user differences are more subtle than inter-user differences.

From these two experiments we conclude that dictionaries yielding sparse representations can successfully describe the morphology of normal heartbeats. Moreover, we show that the our anomaly detection algorithm provides an effective user-specific and position-specific monitoring solution.

Finally, we remark that in ECG signals acquired from wearable devices, user's movements can introduce low quality heartbeats, i.e., out-

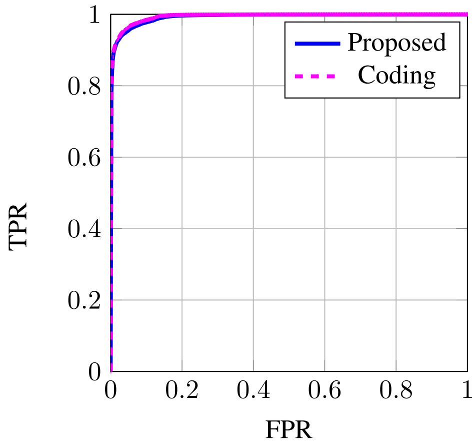


**Figure 8.7:** *ROC curves of the Proposed and Coding anomaly detection algorithms on the MIT-BIH dataset. These curves are obtained by averaging all the ROC curves computed for each user in the MIT-BIH dataset. The Coding algorithm (*$\text{AUC} = 0.9935$*) slightly outperforms the Proposed one (*$\text{AUC} = 0.992$*), but both of them achieve good performances on this dataset, containing several types of arrhythmias.*

liers, that might impair dictionary learning. Thus, we repeat the inter-user anomaly-detection experiment to assess whether the considered algorithm can tolerate small percentage of outliers in the training data. In particular, we consider the best configuration and introduce in the training sets, $1\%, 2\%, 4\%, 8\%$ of outliers, which are selected among those heartbeats that were initially discarded. This experiment is repeated 15 times, and the average ROC curves are reported in Figure 8.6(b). It can be seen that the performance of the anomaly detection are stable when including only $1\%$ and $2\%$ of outliers, but dramatically decreases when outliers are $8\%$. This suggests that it is necessary to reduce the number of outliers from the training set, e.g., by some prescreening method that analyzes MEMS recordings that are embedded on wearable devices such as the Pulse device.

**Experiments on the MIT-BIH Dataset.** For each user in the dataset, we compute the ROC curves and the AUC values obtained using our algorithm and Coding algorithm [152]. The average ROC curves are plotted in Figure 8.7. The coding algorithm achieves a median AUC equal to 0.9935 and outperforms our algorithm (median AUC equal to 0.992), as confirmed by a two-samples sign rank test performed over the populations of AUC values ($p$-value $= 0.002$). However, our algorithm achieves a very good performance with a computational complexity of $O(\kappa pn)$, that is significantly lower than Coding algorithm, which is $O(n^3t + pnt)$ [152], where

$t$ is the number of iterations required by the algorithm to achieve convergence, that is typically several tens.

CHAPTER 9

# Domain Adaptation for Sparse Representations

In this chapter we presents two domain-adaptation solutions to adapt our anomaly-detection algorithm when the process generating normal data changes. At first we adapt our anomaly-detection algorithm to make it scale-invariant (Section 9.1) and show that it successfully detect defects in SEM images acquired at different magnification levels (Section 9.2). Then, in Section 9.3 we consider ECG monitoring, and propose to learn user-independent transformations to adapt our user-dependent anomaly-detection algorithm. Finally, we show the results of our experiments on publicly available datasets of ECG signals in Section 9.4.

## 9.1 Multiscale Anomaly Detection

For simplicity, we illustrate the proposed scale-invariant anomaly detection algorithm assuming a single training image $s$ is provided, even though multiple training images can be easily handled. In the following we explain how to compute the multiscale dictionary $D$ and the coefficient vector $\mathbf{x}$ for a given patch $\mathbf{s}$, then we illustrate the anomaly-detection algorithm.

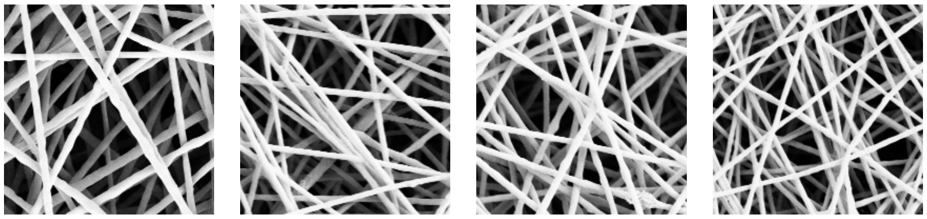

(a) Series A (b) Series B (c) Series C (d) Series D

**Figure 9.1:** *Example of normal images acquired at different magnification levels. Patches extracted from images are perceptually similar, but their content is very different.*

### 9.1.1 Multiscale Dictionary

The dictionary $D$ is expected to provide suitable representations of normal images acquired at different scales. Various algorithms for learning multiscale dictionaries have been proposed in the literature [114–118]. However, to better investigate the role of the sparse coding and the choice of suitable indicators to achieve scale-invariance, we adopt a simple design of the multiscale dictionary.

Let us denote by $s_\sigma$ the image $s$ with support rescaled by a factor $\sigma$. Consider now a set of $L$ scaling factors $\sigma_i$, $i \in \{1, \dots, L\}$ and construct, for each image $s$, a set of rescaled images $s_{\sigma_i}$ to simulate normal data at different scales. Since we assume the scale of the training image is higher than in test images, we consider scaling factors $\sigma \leq 1$. For each rescaled images $s_{\sigma_i}$ we extract a suitable set of patches, then assemble them as the columns of matrix $S_i \in \mathbb{R}^{d \times m}$. The dictionary $D_i \in \mathbb{R}^{d \times n_i}$ corresponding to the scale $\sigma_i$ is thus learned solving the dictionary learning problem (8.5) (BPDN) [184, 185] problem.

The multiscale dictionary $D \in \mathbb{R}^{d \times n}$ representing the training image at multiple scales is constructed by collecting all the learned dictionaries $D_i$ into a single matrix

$$D = [D_1 \mid D_2 \mid \cdots \mid D_L] . \tag{9.1}$$

In principle, the dictionaries $D_i$ may have different number of columns $n_i$, however here we consider $D_i$ having the same size $d \cdot n/L$.

### 9.1.2 Multiscale Sparse Coding

The sparse coding of each patch $\mathbf{s}$ with respect to the dictionary $D$ corresponds to computing a sparse vector $\mathbf{x} \in \mathbb{R}^n$ of coefficients that properly approximate $\mathrm{s}$. Given the specific form of $D$, each coefficient vector has the form:

$$\mathbf{x} = \left[\mathbf{x}_1^T \; \mathbf{x}_2^T \; \cdots \; \mathbf{x}_L^T\right]^T , \tag{9.2}$$

where $\mathbf{x}_i$ is a column vector that collects the coefficients corresponding to dictionary $D_i$ learned from the image at scale $\sigma_i$. For anomaly-detection purposes, it is not desirable to approximate a patch $\mathrm{s}$ by mixing atoms from different dictionaries $D_i$, as this mixture could possibly match anomalous structures. Therefore, we expect that in each sparse representation $\mathbf{x}$, only one, or possibly a few, groups $\mathbf{x}_i$ are active, i.e. $\mathbf{x}$ should be group sparse. This goal is achieved by formulating the sparse coding as a BPDN problem that includes an $\ell^{2,1}$-norm regularization term [186]

$$\hat{\mathbf{x}} = \arg\min_{\mathbf{x}} \frac{1}{2}\|\mathrm{s} - D\mathbf{x}\|_2^2 + \lambda\|\mathbf{x}\|_1 + \xi \sum_{j=1}^{L} \|\mathbf{x}_j\|_2 \, . \tag{9.3}$$

When solving (9.3), the group-sparsity term penalizes representations involving atoms belonging to different dictionaries. This problem is convex and can be solved via ADMM [174].

### 9.1.3 Anomaly Detection

We use the learned dictionary $D$ and the sparse coding procedure in (9.3) to define, for each patch $\mathrm{s}$, an indicator vector $\mathbf{f}(\mathrm{s})$ that assesses the extent to which $\mathrm{s}$ is consistent with $\mathcal{P}_N$.

The vector $\mathbf{f}(\mathrm{s})$ stacks all the summands of the cost function minimized during the sparse coding (9.3) of $\mathrm{s}$: the reconstruction error $\|\mathrm{s} - D\mathbf{x}\|_2$, the sparsity $\|\mathbf{x}\|_1$, and the group sparsity $\sum_i \|\mathbf{x}_i\|_2$. The group-sparsity term is used to assess the spread of significant coefficients among different scales of the dictionary atoms. Since normal patches are expected to involve atoms from one or few scales $\sigma_i$, this term is expected contribute to discriminate normal ad anomalous patches. Therefore, for each patch $\mathrm{s}$ we obtain an indicator vector having three components:

$$\mathbf{f}(\mathrm{s}) = \begin{bmatrix} \|\mathrm{s} - D\mathbf{x}\|_2 \\ \|\mathbf{x}\|_1 \\ \sum_i \|\mathbf{x}_i\|_2 \end{bmatrix} . \tag{9.4}$$

To detect whether a patch $\mathrm{s}$ is normal or anomalous we proceed as in Section 8.2 and adopt the following decision function $\mathcal{T}\colon \mathbb{R}^3 \to \mathbb{R}$:

$$\mathcal{T}(\mathbf{f}) = (\mathbf{f} - \overline{\mathbf{f}})^T \Sigma^{-1} (\mathbf{f} - \overline{\mathbf{f}}), \tag{9.5}$$

where $\overline{\mathbf{f}}$ and $\Sigma$ are the average and the sample covariance matrix of $\mathbf{f}$ computed on additional normal patches, respectively. Using (9.5) is equivalent to build a confidence region from the values of $\mathbf{f}$ computed on normal patches. We do not use KDE as in Section 8.3, since in this case the indicator vector is trivariate and we experience that using a confidence region yields more stable performance than density estimation.

Since we analyze test images in a patch-wise manner and we consider overlapping patches, in practice we assign to each pixel one label (normal/anomalous) for each patch including it. To aggregate all these labels, we consider the same majority-voting scheme described in Section 8.3: a pixel is considered anomalous when the majority of the patches containing that pixel are labeled anomalous.

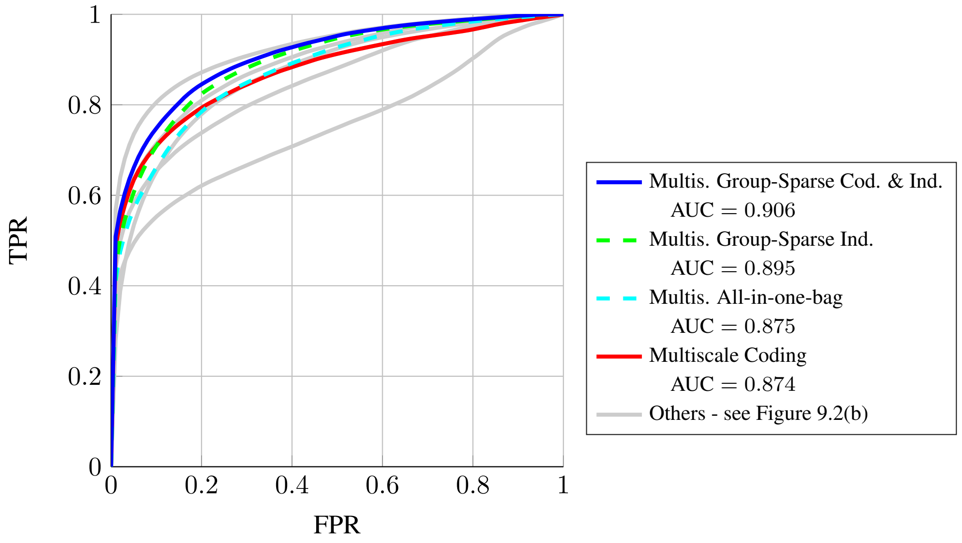
## 9.2 Experiments on Images at Multiple Scale

In this section we assess the performance of the proposed scale-invariant anomaly-detection algorithm. We consider 20 SEM images acquired at different magnification levels, and we group them into 4 series, each sharing the same magnification. An example of a normal image from each series is shown in Figure 9.1. It can be seen that images from Series A have been acquired at the highest scale (maximum magnification), thus they are used to train the proposed anomaly detector. In the test phase, we consider images containing anomalies from all these series, thus from different scales. The performance of the anomaly-detector can be assessed thanks to a binary mask that labels each pixel as normal or anomalous and that is provided for each image.

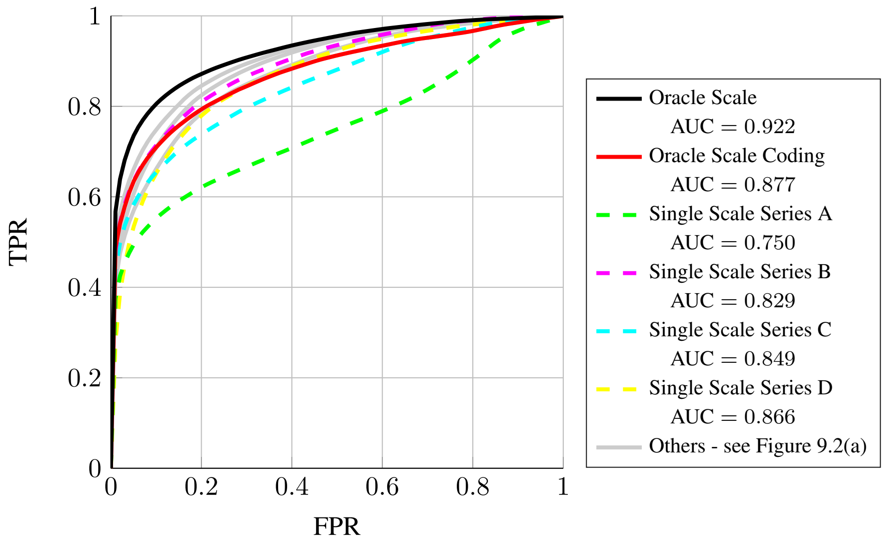
In this experiment we use patches having size $32 \times 32$ and, to speed up the dictionary learning (8.5) and the sparse coding (9.3) stages, we project patches in the Discrete Cosine Transform (DCT) domain and consider only the first 225 coefficients, ordered in a zig-zag fashion, so that $\mathbf{s}$ in (8.1) is a vector of 225 DCT coefficients.

We consider the usual figures of merits to assess the performance of the anomaly detector: FPR and TPR, that are plotted in the ROC curves, and AUC. We consider the following anomaly detection techniques:

- **Multiscale Group-Sparse Coding and Indicator**: this is our scale-invariant algorithm. The multiscale dictionary $D$ is learned by scaling the training images of a factor $\sigma \in \{1, 0.75, 0.5\}$. This method differs from the following ones because it is multiscale in all its parts: dictionary learning, sparse coding and computed indicators.



(a) Solutions based on multiscale dictionary $D$.



(b) Solutions based on single-scale dictionaries $D_i$.

**Figure 9.2:** *ROC curves for all methods presented in Section 9.2. (a) the ROC curves of solutions based on the multiscale dictionary $D$ (9.1); (b) the ROC curves of solutions based on single scale dictionary $D_i$. There are 4 ROC curves for single scale dictionaries: one for each series of images used in the training phase. The area under the curve (AUC) of each ROC curve is reported in the legend.*

- **Multiscale Group-Sparse Indicator**: here we perform the sparse coding via the standard BPDN without the group sparsity term, i.e. we set $\xi = 0$ in (9.3). We use the same multiscale dictionary $D$ (9.1), learned in the above solution. Then, we monitor the whole indicator vector (9.4) that includes also the group sparsity term, which is however ignored in the sparse coding.

- **Multiscale All-in-one-bag**: this is the solution we present in Section 8.3 and here it uses the multiscale dictionary $D$. In practice, the group sparsity is neither taken into account in the sparse coding nor in the indicator vector.

- **Multiscale Coding**: we use the same multiscale dictionary $D$ (9.1) as in the above solutions, and we perform the sparse coding following the Coding algorithm [152], described in Section 8.5.3.

- **Oracle Scale**: we learn 4 different dictionaries, one for each series, and process test images using the dictionary learned from images acquired at the same (correct) scale. This is considered ideal solution since the correct scale is rarely known, exactly. This algorithm is the one proposed in the previous chapter, which does not consider a group sparsity term.

- **Oracle Scale Coding**: as in the Oracle Scale, we use the dictionary learned from training images at the same scale of the test images, but the sparse coding and the anomaly detection using the Coding algorithm [152].

- **Single Scale**: we learn 4 dictionaries, one from each series, and use each of them to detect anomalies in images from all the series. Anomalies are detected using the anomaly detector proposed in 8, which is not multiscale. Obviously, the performance of this solution might vary according to the series used for training.

The ROC curves in Figure 9.2 and the corresponding AUC values indicate that using multiscale dictionaries is beneficial, as these provide better performance than single scale dictionaries in all the considered solutions. As expected, the *Oracle Scale* solution outperforms all the others, since test images are analyzed by a dictionary that was learned on normal images acquired at the same scale. However, these settings might not be realistic in all the practical applications. The *Multiscale Group-Sparse Coding and Indicator* achieves the best performance. In particular, it outperforms

*Multiscale All-in-one-bag* and *Multiscale Coding*, demonstrating that simple sparsity with respect to a multiscale dictionary is not enough to handle test images at a different scales, and that the group sparsity term is instead necessary in the design of the anomaly detector. Moreover, the comparison between *Multiscale Group-Sparse Coding and Indicator* and *Multiscale Group-Sparse Indicator* confirms that is not enough to measure the group sparsity in the indicator vector, but this has to be taken into account also during the sparse coding.

## 9.3 Domain Adaptation in ECG Monitoring

In this section we address the domain-adaptation problem in ECG monitoring, and present a solution to adapt our user-specific anomaly-detection algorithm by means of user-independent transformations that depend only on the source and target heart rates. In particular, we show that these transformations can be successfully learned from datasets containing heartbeats of several users at different heart rates, like [187]. While the anomaly-detection algorithm has to be configured every time the device is positioned, these transformations have to be learned offline and only once, and used during the online monitoring. In the following, we tackle the problem of learning transformations to adapt both the dictionary $D_{u,r_0}$ (Section 9.3.1), and threshold $\gamma_{u,r_0}$ (Section 9.3.2) to control the false positive rate.

### 9.3.1 Dictionary Adaptation

Our goal is to learn a collection of user-independent transformations $\{\mathcal{F}_{r,r_0}\}$ that maps each subspace in $\mathfrak{D}_{r,r_0}$ into a subspace in $\mathfrak{D}_{u,r}$. To preserve the subspace property, we set each $\mathcal{F}_{r,r_0}$ to be a linear function from $\mathbb{R}^{d(r_0)\times n}$ to $\mathbb{R}^{d(r)\times n}$. A linear function between two such vector spaces has in general $d(r_0)d(r)n^2$ degrees of freedom, which is quite a large number when the heartbeats are composed of hundred samples ($d \approx 150$). To reduce the number of parameters, thus the risk of overfitting, we constrain the linear transformation $\mathcal{F}_{r,r_0}$ to have a specific shape that reflects our modeling assumption (8.1). More precisely, these transformations have to map each atom of $D_{u,r_0}$ to an atom of $D_{u,r}$, thus each generator of the subspaces in $\mathfrak{D}_{u,r_0}$ is mapped to a generator of the subspaces in $\mathfrak{D}_{u,r}$. In this case, $\mathcal{F}_{r,r_0}$ is described by a matrix $F_{r,r_0} \in \mathbb{R}^{d(r)\times d(r_0)}$:

$$D_{u,r} = \mathcal{F}_{r,r_0}(D_{u,r_0}) = F_{r,r_0} D_{u,r_0}, \tag{9.6}$$

and the number of degrees of freedom of $\mathcal{F}_{r,r_0}$ reduces to $d(r)d(r_0)$ which is the number of entries of the matrix $F_{r,r_0}$. We point out that the dictionary-adaptation solution in (9.6) follows from the simple geometrical interpretation of dictionary yielding sparse representation. More complex models of heartbeats might not be straightforwardly adapted to track heart rate variations.

We learn $F_{r,r_0}$ from the datasets in [187] by extracting pairs of training sets $\{S_{u,r_0}\}$ and $\{S_{u,r}\}$ for many users $u \in \{1, \dots, L\}$ and solve the following optimization problem:

$$\begin{aligned} F_{r,r_0} = \arg\min_{F,\{X_u\}_u} \frac{1}{2} \sum_{u=1}^{L} \|S_{u,r} - F D_{u,r_0} X_u\|_2^2 + \mu \sum_{u=1}^{L} \|X_u\|_1 + \\ + \frac{\lambda}{2} \|W \odot F\|_2^2 + \xi \|W \odot F\|_1, \end{aligned} \tag{9.7}$$

where the columns of $X_u \in \mathbb{R}^{n \times m}$ contain the sparse w.r.t. $FD_{u,r_0}$ of the corresponding heartbeats in $S_{u,r}$, and the symbol $\odot$ denotes the Hadamard product. All the norms in (9.7) are vector norms, rather than matrix norms.

In what follows we describe the role of each term in (9.7): $\|S_{u,r} - FD_{u,r_0} X_u\|_2^2$ assesses how good the transformed dictionary is at approximating the heartbeats of the user $u$. This term sums up the reconstruction error over training heartbeats for the $L$ users, and guarantees that dictionaries transformed by $F_{r,r_0}$ can properly describe heartbeats in the target domain. We adopt three regularization terms controlled by the non-negative regularization parameters $\lambda, \mu, \xi$, and a suitable weight matrix $W \in \mathbb{R}^{d(r) \times d(r_0)}$. The first two regularization terms guarantee that the each transformed dictionary $FD_{u,r_0}$ provides sparse representations to the heartbeats in the target domain, for each user $u$. We adopt an $\ell^1$ regularization to enforce sparsity as a customary choice in the literature [173] since the $\ell^1$ norm is convex. The other two terms represent a weighted elastic net penalization over $F$, which improves the stability of the optimization problem, and the weighting matrix $W$ introduces some a priori information about the transformation $\mathcal{F}_{r,r_0}$ in the minimization. In our case, we expect $\mathcal{F}_{r,r_0}$ to be local, namely that each sample of a transformed atom is determined by only few neighboring samples in the input atom in $D_{u,r}$. Therefore, $W$ features larger weights in positions far from the diagonal of $F$, and small weights close to the diagonal. In particular, we define the entries of $W$ using Gaussian weights:

$$w_{ij} = 1 - c \cdot e^{-\frac{(j-i)^2}{\sigma}}, \quad i \in \{1, \dots, d(r)\},\ j \in \{1, \dots, d(r_0)\}, \tag{9.8}$$

where $\sigma > 0$ determines the width of the Gaussian and $c > 0$ is a constant to ensure that $0 \leq w_{ij} \leq 1, \forall\, i, j$. An example of $W$ is shown in Figure 9.3(a).

Solving (9.7) is not straightforward since it is not jointly convex in $X_u$ and $F$. However, the functional to be minimized is convex with respect to each variable when the other one is fixed. Therefore, we solve it by ADMM [174], which has been shown to enjoy good convergence properties in these settings [188]. The rationale behind the ADMM is to split the optimization problem in easier sub-problems, and alternate their optimization. To this purpose, we reformulate (9.7) in an equivalent form, where it is easier to derive the equations of ADMM sub-problems:

$$\begin{aligned} \underset{F,\{X_u\},G,\{Y_u\}}{\arg\min} \; & \frac{1}{2}\sum_{u=1}^{L} \|S_{u,r} - FD_{u,r_0}X_u\|_2^2 + \mu \sum_{u=1}^{L} \|Y_u\|_1 + \\ & + \frac{\lambda}{2}\|W \odot G\|_2^2 + \xi\|W \odot G\|_1, \\ \text{s.t.} \qquad & F - G = 0, \quad X_u - Y_u = 0, \; \forall u \end{aligned} \tag{9.9}$$

where $G \in \mathbb{R}^{d(r)\times d(r_0)}$ and $Y_u \in \mathbb{R}^{n\times m}$ are auxiliary variables. According to the ADMM framework, we define the Augmented Lagrangian [174] of (9.9) and solve it by alternating the optimization of the following sub-problems:

$$X_u^{(k+1)} = \underset{X_u}{\arg\min} \frac{1}{2}\|S_{u,r} - FD_{u,r_0}X_u\|_2^2 + \frac{\rho}{2}\|X_u + Y_u^{(k)} + Z_u^{(k)}\|_2^2, \tag{9.10}$$

$$Y_u^{(k+1)} = \underset{Y_u}{\arg\min}\, \mu\|Y_u\|_1 + \frac{\rho}{2}\|X_u^{(k+1)} - Y_u + Z_u^{(k)}\|_2^2, \tag{9.11}$$

$$F^{(k+1)} = \underset{F}{\arg\min} \frac{1}{2}\sum_{u=1}^{L} \|S_{u,r} - FD_{u,r_0}X_u^{(k+1)}\|_2^2 + \frac{\rho}{2}\|F + G^{(k)} + H^{(j)}\|_2^2, \tag{9.12}$$

$$G^{(k+1)} = \underset{G}{\arg\min} \frac{\lambda}{2}\|W \odot G\|_2^2 + \xi\|W \odot G\|_1 + \frac{\rho}{2}\|F^{(k+1)} - G + H_k\|_2^2, \tag{9.13}$$

$$Z_u^{(k+1)} = Z_u^{(k)} + X_u^{(k+1)} - Y_u^{(k+1)}, \tag{9.14}$$

$$H^{(k+1)} = H^{(k)} + F^{(k+1)} - G^{(k+1)}, \tag{9.15}$$

where $Z_u \in \mathbb{R}^{n\times m}$ and $H \in \mathbb{R}^{d(r)\times d(r_0)}$ are the scaled Lagrange multipliers of the constraints in (9.9), that are updated in (9.14) and (9.15), respectively. Sub-problems (9.10) and (9.12) are quadratic expressions which can be efficiently solved by Gaussian elimination. Problem (9.11) admits a

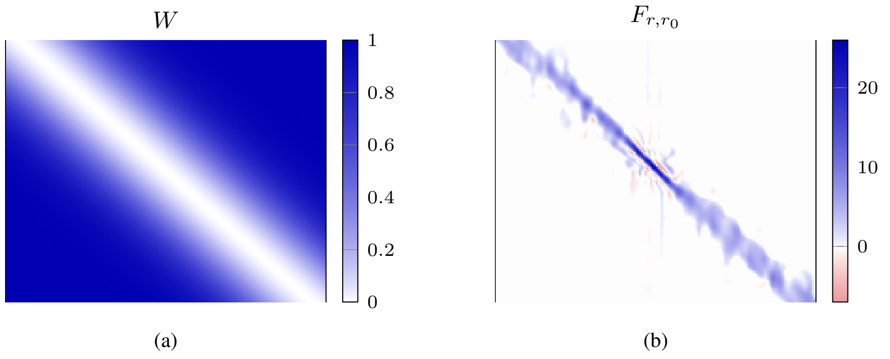


**Figure 9.3:** *(a) The weight matrix $W$ used in* (9.7) *to learn the matrix $F_{r,r_0}$. The elements of $W$ are set according to* (9.8) *to enforce the learned transformation to be local: the elements on the diagonal are small, while the ones far from the diagonal are large. (b) Example of learned $F_{r,r_0}$ for $r = 90$, $r_0 = 70$ using the weight matrix $W$ shown in (a). The learned transformation is local, since nonzero elements of $F_{r,r_0}$ are concentrated around the diagonal.*

closed-form solution which corresponds to the proximal mapping [189] of the function $(\mu/\rho)||\cdot||_1$

$$\left[Y_u^{(k+1)}\right]_{ij} = \mathcal{S}_{\mu/\rho}\left(\left[X_u^{(k+1)} + Z_u^{(k)}\right]_{ij}\right), \tag{9.16}$$

where $[\cdot]_{ij}$ denotes a matrix entry, and $\mathcal{S}_\eta \colon \mathbb{R} \to \mathbb{R}$:

$$\mathcal{S}_\eta(x) = \text{sign}(x) \cdot \max\{0, x - \eta\}, \tag{9.17}$$

is the soft-thresholding operator. From Theorem 4 in [189] and simple algebra, it follows that also (9.13) can be solved by soft thresholding:

$$\left[G^{(k+1)}\right]_{ij} = \frac{1}{1 + \lambda w_{ij}^2} \mathcal{S}_{\xi w_{ij}/\rho}\left(\left[F^{(k+1)} + H^{(k)}\right]_{ij}\right). \tag{9.18}$$

To initialize the ADMM algorithm, we set to zero the values of all the variable but $F^{(0)}$, which is initialized to uniformly distributed random values to avoid trivial solutions. Then, we iteratively solve (9.10–9.15) until a maximum number of iterations is reached or the primal and dual residuals [174] fall below given thresholds.

Figure 9.3(b) shows an example of learned $F_{r,r_0}$, which, as expected, is local, since the nonzero elements of $F_{r,r_0}$ are concentrated at the diagonal.

The same optimization problem is solved for several pairs $(r, r_0)$ to learn the collection $\{\mathcal{F}_{r,r_0}\}$ that is used to adapt the user-specific dictionary $D_{u,r_0}$ to any target heart rate $r$ during the online monitoring.

### 9.3.2 Threshold Adaptation

Adapting the dictionary $D_{u,r_0}$ when the heart rate changes is not enough to enable long-term ECG monitoring. In fact, also the threshold $\gamma_{u,r_0}$ in the decision rule (8.6) has to be adapted. To this purpose, we learn a set $\{f_{r,r_0}\}$ of user-independent transformations by solving the following optimization problem:

$$\gamma_{u,r} = f_{r,r_0}(\gamma_{u,r_0}) = \gamma_{u,r_0} \cdot \exp(a(r - r_0)), \tag{9.19}$$

where $a \in \mathbb{R}$ is the only parameter that has to be learned.

The choice of transformation in (9.19) is justified by an empirical consideration. Boxplots from Figure 9.4 report the empirical distribution of $\log e_{u,r}$ for two different users at heart rates $r$, being $e_{u,r}$ the random variable corresponding to $e(\mathbf{s}_{u,r})$ when $\mathbf{s}_{u,r}$ is drawn from the stochastic process $\mathcal{N}_{u,r}$ and the sparse coding is performed w.r.t. the adapted dictionary $D_{u,r}$. The trend of these boxplots suggests that $(\log e_{u,r} - \log e_{u,r_0})$ is proportional to $(r - r_0)$. This observation implies the following relation:

$$\log e_{u,r} = \log e_{u,r_0} + a(r - r_0), \tag{9.20}$$

where $a > 0 \in \mathbb{R}$ is the parameter yielding the first order approximation to the trend of such distributions. To maintain a fixed false positive rate (FPR) when the heart rate changes, we need to set $\gamma_{u,r}$ to guarantee a constant probability of considering a normal heartbeat $\mathbf{s}_{u,r}$ as anomalous:

$$P(e_{u,r} > \gamma_{u,r}) = P(e_{u,r_0} > \gamma_{u,r_0}). \tag{9.21}$$

Combining (9.20) and (9.21), we can derive the relation between $\gamma_{u,r_0}$ and $\gamma_{u,r}$:

$$\begin{aligned} P\left(e_{u,r_0} > \gamma_{u,r_0}\right) &= P\left(e_{u,r} > \gamma_{u,r}\right) = P\left(\log e_{u,r} > \log \gamma_{u,r}\right) = \\ &= P\left(\log e_{u,r_0} + a(r - r_0) > \log \gamma_{u,r}\right) = \\ &= P\left(e_{u,r_0} > \gamma_{u,r} \exp(-a(r - r_0))\right). \end{aligned} \tag{9.22}$$

The last equation implies that $\gamma_{u,r_0} = \gamma_{u,r} \exp(-a_{r,r_0}(r - r_0))$, thus (9.19).

We now address the problem of estimating such transformations from a training set containing heartbeats of multiple users. Figure 9.3(c) suggests that the distribution of $\log e_{u,r}$ is symmetric, thus we write

$$\log e_{u,r} = b_{u,r} + \eta_u, \tag{9.23}$$

where $b_{u,r}$ is the expected value of $\log e_{u,r}$, while $\eta_u$ is a stochastic term that has a symmetric distribution w.r.t the origin. Taking the expected value

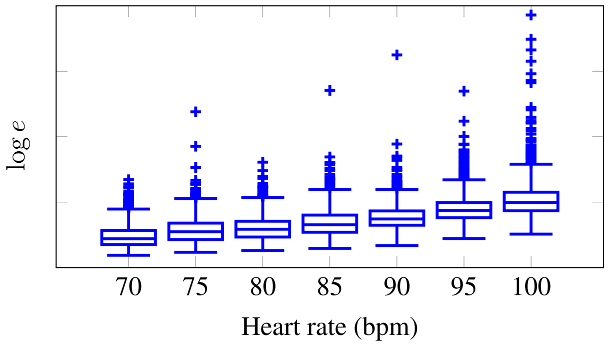


**Figure 9.4:** *Boxplot of the* $\log e(\mathbf{s}_{u,r})$ *computed on normal heartbeats acquired at different the heart rate* $r$*. In both cases the distribution of* $\log e(\mathbf{s}_{u,r})$ *seems to shift of a value that increases linearly with the heart rate* $r$*. This support pour choice of transformation in* (9.19).

in (9.20), we have that $b_{u,r} = b_{u,r_0} + a(r - r_0)$, which we can substitute in (9.23) and obtain:

$$\log e_{u,r} = b_{u,r_0} + a \cdot (r - r_0) + \eta_u, \tag{9.24}$$

which is a linear regression model for $\log e_{u,r}$ w.r.t. the heart rate $r$. The parameter $b_{u,r_0}$ is user-dependent and can be estimated from $S_{u,r_0}$, while we can estimate $a$ via least squares over multiple users by solving

$$a = \arg\min_{\tilde{a}} \frac{1}{2} \sum_{u=1}^{L} \sum_{j=1}^{R} \sum_{i=1}^{m} \left(\log\left(e(\mathbf{s}_{u,r_j}(i))\right) - b_{u,r_0} - \tilde{a} \cdot (r_j - r_0)\right)^2, \tag{9.25}$$

where $b_{u,r_0}$, $u \in \{1, \ldots, L\}$ is the average value of $e(\mathbf{s}_{u,r_0})$ over $\{S_{u,r_0}\}$. We estimate $a$ by setting to 0 the derivative of the functional in (9.25):

$$a = \frac{\sum_{u=1}^{L} \sum_{j=1}^{R} \sum_{i=1}^{m} \left(\log\left(e(\mathbf{s}_{u,r_j}(i))\right) - b_u\right) \cdot (r_j - r_0)}{mL \sum_{j=1}^{R} (r_j - r_0)^2}. \tag{9.26}$$

The estimated $a$ defines the user-independent transformation in (9.19), to adapt the decision rule. Our anomaly-detection algorithm can thus perform long-term ECG monitoring by transforming the user specific dictionary $D_{u,r_0}$ and the threshold $\gamma_{r_0}$ by means of the transformations in (9.6) and (9.19), respectively.

**Table 9.1:** *Datasets from Physionet [187] used in the experiments*

| Dataset | Number of users | Average duration (h) | Heart rate resolution (bpm) |
|---|---|---|---|
| LTSTDB | 80 | 22 | 5 |
| LTAFDB | 84 | 24 | 5 |
| LTDB | 7 | 16 | 5 |
| EDB | 45 | 2 | 10 |

## 9.4 Experiments on ECG signals

The goal of our experiments it to show that learned user-independent transformations can successfully adapt our anomaly-detection algorithm. To this purpose we consider 4 datasets publicly available from Physionet [187] which have been acquired using Holter devices. All the dataset from Physionet have been manually annotated using an automatic tool and corrected by cardiologists. Table 9.1 reports the number of users in each dataset and the durations of the ECG signals recorded. We use different resolution for heart rate quantization (see Table 9.1) on the duration of the ECG signal, to guarantee a sufficient number of heartbeats for each quantized heart rate. We consider as anomalous heartbeats all the annotated arrhythmias.

**Figures of Merit.** To assess how good the adapted dictionaries are at modeling the heartbeats, we select as figure of merit the distance $e$ of the heartbeats from the transformed union of low dimensional subspaces, that can be also interpreted as the reconstruction error yielded by the sparse approximation. Moreover, we adopt the usual figures of merit to asses the anomaly detection performance, i.e., FPR, TPR and AUC.

**Alternative Solutions.** As alternative anomaly-detection algorithm we consider **Coding** [152], described in Section 8.5.3. Moreover, we consider the following dictionary-adaptation solutions:

- **Cut**: since the support of each heartbeat contracts as the heart-rate increases (see Figure 2.2), the simplest form of adaptation consists in removing the first and the last samples of each column of $D_{u,r_0}$, thus "cutting" the support of each atom in the dictionary. This baseline solution transforms each dictionary $D_{u,r_0}$ using a pre-defined transformation which does not require training data.

- **DTW**: this solution is based on dynamic time-warping [190], an established signal-processing algorithm to align two vectors and measure their similarity. Given $\mathbf{v}_0 \in \mathbb{R}^{d_0}$ and $\mathbf{v}_1 \in \mathbb{R}^{d_1}$ where $d_0 \neq d_1$, dynamic time-warping performs a non uniform resampling of $\mathbf{v}_0$ and $\mathbf{v}_1$ to obtain two aligned vectors $\widetilde{\mathbf{v}}_0, \widetilde{\mathbf{v}}_1 \in \mathbb{R}^{d_a}$ that have a common

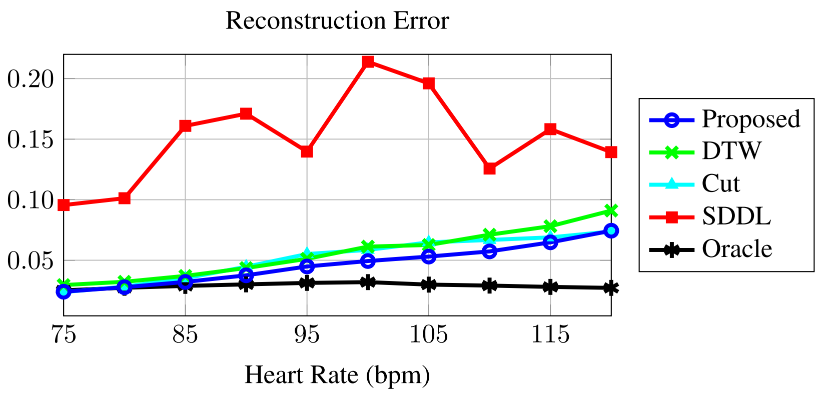


**Figure 9.5:** *The median reconstruction error over all users. As expected, in the Oracle solution the reconstruction error is nearly constant w.r.t. the heart rate $r$, and increases with $r$ in all domain adaptation solutions, confirming that the change in the heartbeat morphology becomes more evident for large heart rates. Among such solutions, the Proposed one achieves the lowest reconstruction error.*

support $d_a$, such that $d_a \geq d_0$ and $d_a \geq d_1$. The resampling patterns are estimated by minimizing the Euclidean distance between $\widetilde{\mathbf{v}}_0$ and $\widetilde{\mathbf{v}}_1$ through dynamic programming. Since resampling is a linear operation, it is always possible to express DTW by two matrices $A_i \in \mathbb{R}^{d_a \times p_i}$, $i = 0, 1$, such that $\widetilde{\mathbf{v}}_i = A_i \mathbf{v}_i$. The matrices define the transformation for each pair of heartbeats. In particular, we compute $A_{r_0}$ and $A_r$ by aligning the first principal components of $S_{u,r_0}$ and $S_{u,r}$. To obtain user-independent transformations, the resampling patterns are computed to minimize the sum of the Euclidean distance between the aligned first principal components of $S_{u,r_0}$ and $S_{u,r}$ over many users $u$. The corresponding transformation $\mathcal{F}_{r,r_0}$ to map dictionaries is linear as in (9.6) and is defined by $F_{r,r_0} = A_r^+ A_{r_0}$, begin $A_r^+$ the pseudo-inverse of $A_r$.

- **SDDL**: Shared Domain-adaptive Dictionary Learning is a domain-adaptation algorithm specifically designed for dictionaries [133]. For each source-target domain pairs, SDDL jointly learns two projections from these domains into a common subspace, as well as a shared dictionary providing sparse representations of the projected data. While this solution is claimed to be general in [133], it has been primarily developed for classification purposes, as it learns class-wise mutually incoherent dictionaries. We adapt [133] by learning the projection from the LTSTDB and LTAFDB datasets, and the dictionaries in the

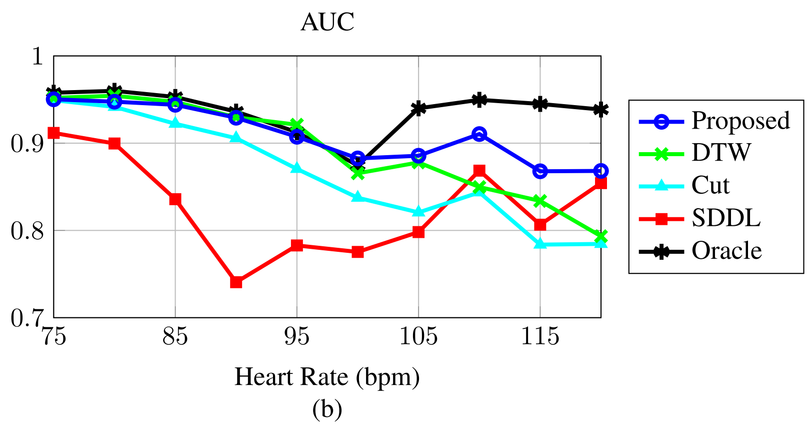


**Figure 9.6:** *The median AUC computed over all the users. As in case of the reconstruction error (see Figure 9.5), the Proposed solution leads to the best performance in case of domain adaptation, although the DTW achieves similar AUC.*

shared subspace by means of the KSVD [106] in place of the discriminative dictionary-learning algorithm in [133]. Thus, during the user-specific configuration, we project $S_{u,r_0}$ onto the low-dimensional subspace using the learned projection, then we learn a user-specific (but heart rate independent) dictionary $D_u$. During online monitoring, we project each heartbeat onto the subspace, then we compute the sparse representation w.r.t. $D_u$ and back-project the obtained reconstruction. The $\ell^2$ norm of the difference between the original and the back-projected heartbeats yield the reconstruction error used for anomaly detection.

- **Oracle**: this ideal solution directly learns a dictionary $D_{u,r}$ from a training set $S_{u,r}$ using KSVD in the target domain.As such, this cannot be pursued in practical monitoring, but represents a performance reference for domain-adaptation algorithms.

These domain-adaptation solutions have been trained on all ECG signals in LTSTDB and LTAFDB datasets, which present a large variability both in term of users and heart rates, and tested on all other datasets. Moreover, to assess the performance on all Physionet datasets we learn a set of transformations from LTSTDB dataset and test it on LTAFDB dataset, and viceversa. Finally, since all the domain-adaptation solutions depend on several hyper-parameters, we adopt a 5-fold cross-validation procedure during the training phase, and use random search [191] to select the hyper-parameters that achieve the best heartbeats reconstruction performance on

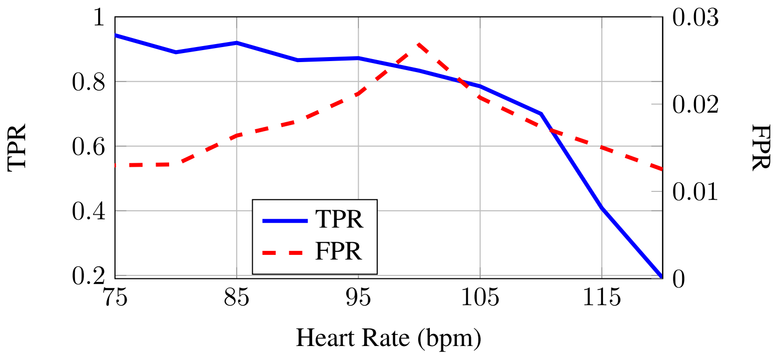


**Figure 9.7:** *Median TPR and FPR computed over all the users in Physionet datasets by setting the threshold $\gamma_{u,r_0}$ to achieve a desired FPR $\alpha = 0.01$.*

a validation set.

**Experiments.** We perform three domain adaptation experiments. The first experiment aims to assess the normal heartbeats reconstruction when adapting dictionaries to track heart rate changes. More precisely, for each test user $u$ we learn a dictionary $D_{u,r_0}$ from heartbeats at heart rate $r_0 = 70$ bpm and then we analyze the reconstruction error on normal heartbeats w.r.t. the dictionary obtained by adapting $D_{u,r_0}$ on different heart rates. Figure 9.5 shows the median reconstruction error over all the test users and heart rates (the lower the better). Our dictionary adaptation solution outperforms all the alternatives (except for the Oracle), in particular at the high heart rates. A signed-rank test confirms that our solution achieves lower reconstruction error than both DTW and CUT ($p$-value $< 0.00001$), except for low heart-rates ($r = 75, 80$), where Proposed and Cut solutions achieve similar performance.

In the second experiment we assess the anomaly-detection performance when the algorithm is adapted to operate at different heart rates on each specific user. Figure 9.6 reports the median AUC computed over all test user for each considered solution. The Proposed one achieves similar performance of DTW for low heart rates, and is the best when $r \geq 100$, although a signed rank test does not report enough statical evidence that the two solutions perform differently. This is not in conflict with results on the reconstruction error. Some anomalous heartbeats can be better perceived at low heart rates and be detected even when the reconstruction w.r.t. the adapted dictionaries is slightly worse.

Finally, in the third experiment we simulate an online monitoring scenario, where we set a desired FPR $\alpha = 0.01$ and assess the TPR achieved when the $D_{u,r_0}$ and $\gamma_{u,r_0}$ using the learned transformations. Figure 9.7 shows the median FPR and TPR computed over all the users. The learned transformation successfully maintain the FPR below the target value $\alpha$ for each heart rate. The TPR is large for small heart rates, but decreases when the heart rate becomes significantly larger than $r_0$. This is probably due to a quality degradation of heartbeats because of user movements which affects the capability of the detector to distinguish between normal and anomalous heartbeats.

CHAPTER 10

# Online ECG Monitoring

In this chapter we present a prototype wearable device able to perform online and long-term monitoring of ECG signals that embeds the anomaly-detection and domain adaptation algorithms presented in the previous chapters. We describe the device in Section 10.1 and the user-configuration that has to be performed every time the device is placed in Section 10.2. Finally, we present the results of online monitoring experiments in Section 10.3.

## 10.1 The Bio2Bit Dongle

The Bio2Bit-Dongle (Figure 10.1) is a wearable device developed by STMicroelectronics that is composed by the *Bio2Bit Move* [192], which is plugged on an adjustable chest strap and acquires and transmits via Bluetooth Low Energy (BLE) the ECG signals, and a low-power *dongle* that analyzes in real time the received signals.

The Bio2Bit Move is a prototype device placed in a 55x40x14 mm envelope, even though the final product sizes can be significantly reduced to yield a non-invasive wearable device. Its battery allows to steadily sense and transmit via BLE the ECG signals for up to 20 hours. The dongle has been replaced by a development board NUCLEO-L476RG [193] which

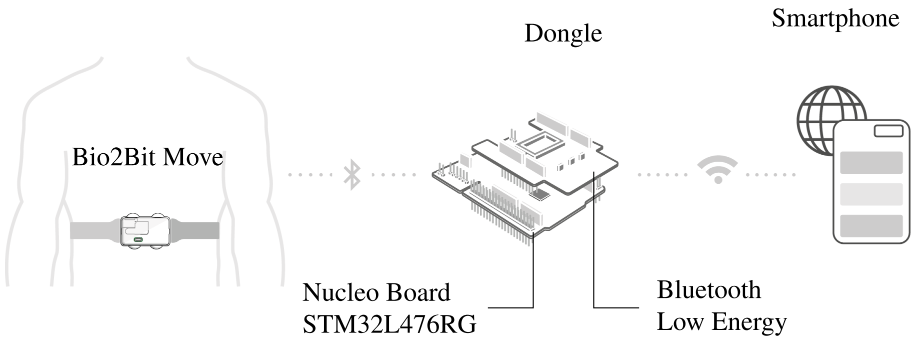


**Figure 10.1:** *The user wears Bio2Bit Move (on a chest strap) and the dongle (in a bracelet, or hanged on a belt), which communicate over a Bluetooth connection. The* user configuration *requires a few minutes of ECG signals acquired in resting conditions, which the dongle segments into heartbeats. These represent the training set that is sent to the user's smartphone, where the dictionary* $D_{u,r_0}$ *and the threshold* $\gamma_{u,r_0}$ *are learned. These are adapted for each heart rate* $r$ *and the QR decomposition of each dictionary is computed. The dongle sends back the adapted thresholds* $\{\gamma_{u,r}\}$ *and the decompositions* $\{(Q_{u,r}, R_{u,r})\}$ *to the dongle. During* online monitoring, *the dongle receives the ECG signal from by the Bio2Bit Move, segments each heartbeats and estimates the heart rate. For each heartbeat, it performs the sparse coding w.r.t. the dictionary that matches the current heart rate. Finally the dongle computes the reconstruction error, and raises an alert when it is above the selected threshold.*

embeds an ultra-low-power microcontroller unit (MCU) based on the high-performance ARM Cortex-M4 32-bit RISC core, operating at a frequency of 80 MHz. This MCU is a likely reference for low-power dongles. The board is also equipped by a BLE evaluation board (X-NUCLEO-IDB05A1) to communicate with the Bio2Bit Move.

## 10.2 User Configuration

Algorithm 10.1 describes the user-configuration phase, which has to be performed every time the Bio2Bit Move is positioned. User configuration includes learning $D_{u,r_0}$ and estimating $\gamma_{u,r_0}$, from the training set $S_{u,r_0}$, as described in Section 9.3.

We experienced that 10 minutes of ECG signals acquired in resting conditions are typically enough for the user configuration: acquired signals are initially segmented by detecting R peaks by means of the Pan-Tompkins algorithm [20]. The heart rate $\bar{r}$ associated to each heartbeat is the median of the inverse of distances between two consecutive R peaks over the last 10 seconds, quantized to a resolution of 10 bpm (beats per minute). The

**Algorithm 10.1** User Configuration

```
Input: Training set $S_{u,r_0}$, source heart rate $r_0$, user-independent transformations $\{\mathcal{F}_{r,r_0}\}$,
    $\{f_{r,r_0}\}$, desired false positive rate $\alpha$.
Output: Matrices $\{Q_{u,r}\}$, $\{R_{u,r}\}$, thresholds $\{\gamma_{u,r}\}$.
1: Split the training set $S_{u,r_0}$ in two sets $T$ and $V$.
2: Learn the user specific dictionary $D_{u,r_0}$ by solving problem (8.4), using $T$ in place of
   $S_{u,r_0}$ as training set.
3: Compute $e(\mathbf{s}_{u,r_0})$ for each heartbeat $\mathbf{s}_{u,r_0}$ in $V$.
4: Set $\gamma_{u,r_0}$ as the $(1-\alpha)$ quantile of the empirical distribution of $e(\cdot)$ over $V$.
5: for each $r$ in the range $[70, 120]$ do
6:     Adapt the dictionary $D_{u,r_0}$ by computing $D_{u,r} = \mathcal{F}_{r,r_0}(D_{u,r_0})$.
7:     Compute the QR decomposition of the adapted dictionary: $D_{u,r} = Q_{u,r}R_{u,r}$.
8:     Adapt the threshold $\gamma_{u,r_0}$ by computing $\gamma_{u,r} = f_{r,r_0}(\gamma_{u,r_0})$.
9: end for
```

most frequent heart rate in these 10 minutes is selected as $r_0$, and only the heartbeats associated to $r_0$ belong to the training set $S_{u,r_0}$. The next operations to configure the device are more computationally demanding than the online monitoring, and can be conveniently performed on a host, such as a smartphone.

The host receives the training set $S_{u,r_0}$ and $r_0$, and divides $S_{u,r_0}$ into a set used to learn $D_{u,r_0}$ and a set to estimate $\gamma_{u,r_0}$. The host pre-computes $D_{u,r} = \mathcal{F}_{r,r_0}(D_{u,r_0})$ and $\gamma_{u,r} = f_{r,r_0}(\gamma_{r_0})$ for a range of admissible heart rates $r$ (e.g., $[70, 120]$), as well as the QR decomposition of each transformed dictionary $\{D_{u,r}\}_r$. The matrices $Q_{u,r}$ and $R_{u,r}$ are then sent back to the dongle that stores them for the optimized sparse coding (see Section 8.4).

## 10.3 Experiments

We use the Biot2Bit-Dongle to collect a dataset of 15 ECG signals. 12 ECG signals are from healthy users, while 3 are from patients affected by a cardiovascular disease and has been annotated by a cardiologist. Each ECG signal in the dataset lasts at least 1 hour and is acquired following a protocol including normal-life activities (e.g. resting, lying down, walking, resting after a small effort), thus the heart-rate significantly varies in each ECG signal. We restrict our analysis to the most frequent heart rates, i.e. $\{70, 80, 90, 100\}$. Due to the limited number of leads and their reduced distance, heartbeats in the B2B dataset are very different from those in the datasets from Physionet used in Chapter 9. Low-quality heartbeats have been discarded by an automatic tool and the supervision of a cardiologist.

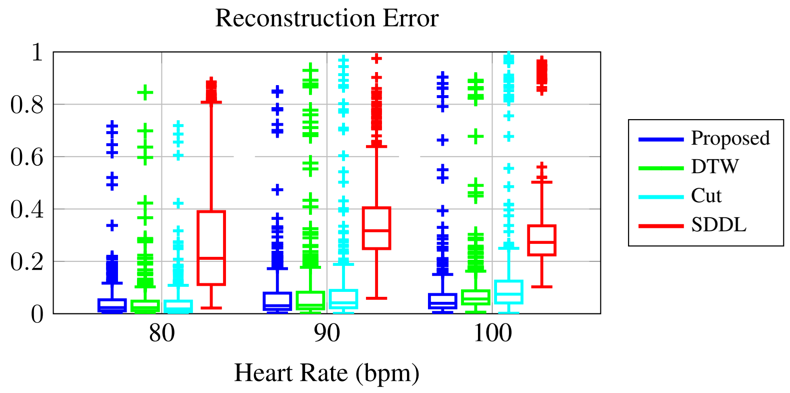


**Figure 10.2:** *Boxplots of the reconstruction error computed over all the user in the datasets. The proposed solution achieves the best performance, especially in case of* $r = 100$.

The goal of our experiments is to show that the transformations learned from Physionet datasets can successfully adapt dictionaries learned from other devices, such as Bio2Bit Dongle, which is completely different from the Holter used to acquire the ECG signals in the Physionet datasets. This is necessary for our solution to enable long-term monitoring. Therefore, we repeat the experiments in Section 9.4 using the same transformations on the B2B dataset to assess both heartbeat reconstruction and anomaly detection performance. The alternative solutions considered are the same as in Section 9.4, except for Oracle one, that cannot be used in this case, since we do not have enough heartbeats to learn a dictionary for each heart rate. For each user $u$ we learn $D_{u,r_0}$ and $\gamma_{u,r_0}$ from the first 10 minutes of ECG signal, and perform domain adaptation to heart rate $r \in \{80, 90, 100\}$.

Figure 10.2 shows the boxplots of the reconstruction error computed on normal heartbeats of each user for different $r$. This plot show that the median reconstruction error of the Proposed solution is lower the others even for large heart rates. As in the experiments on the Physionet datasets, we assess the performance in the online monitoring scenario by setting a desired FPR $\alpha = 0.01$, and computing the actual FPR and TPR score for each heart rate $r$. Figure 10.4(a) shows the performance of our solution on the 3 patients affected by cardiovascular diseases: the FPR exceeds the desired value of $\alpha = 0.01$, but it is maintained constant. The TPR is very large for one patient, but it is smaller in case of the other two, where the number of arrhythmias is low, thus the estimate of the TPR is subject to a large variance.

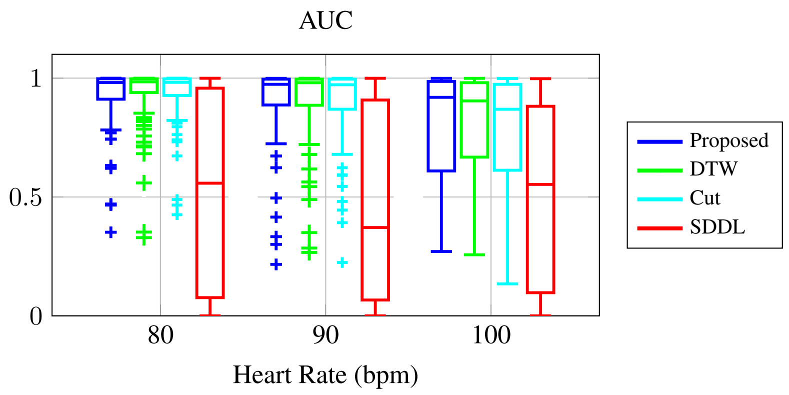


**Figure 10.3:** *Boxplots of the AUC achieved by all solutions to detect* inter-user *anomalous heartbeats. The AUC achieved by the Proposed algorithm is the highest among all the considered domain-adaptation solutions.*

To increase the number of anomalous heartbeats, we pursue the approach described in Section 8.5.5 and artificially introduce *inter-user* anomalies. More precisely, for each healthy user $u$ we consider as anomalous any heartbeat from a different healthy user, which indeed features a morphology that is different from training ones. Figure 10.3 shows the boxplot of the AUC: again, the proposed solution outperforms the others, although DTW performs comparably, as confirmed by a signed rank test ($p$-value $\approx 0.6$). Finally, we assess the anomaly detection performance in the online monitoring scenario also on the inter-user anomalies. Figure 10.4(b) shows the median FPR and TPR, confirming that our solution successfully detects anomalous heartbeats when the heart rate increases while maintaining constant FPR.

Finally, to determine whether our algorithm can be executed online on the Bio2Bit-Dongle in a long term monitoring scenario, we measure the execution time and the power consumption. Our optimized OMP [6] requires on average 1.360 ms to compute the reconstruction error of an heartbeat, which is way below the acquisition time of each heartbeat, and is $48\%$ less than the computing time of a standard OMP implementation. Dictionary adaptation has no overheads, since dictionaries can be pre-transformed during training and stored in memory. The average current absorbed by the dongle during 20 minutes of online monitoring is 10.01 mA. Thus, when the dongle is equipped with a 592 mWh battery such as the one adopted on the Bio2Bit Move, it has 16 hours monitoring autonomy.

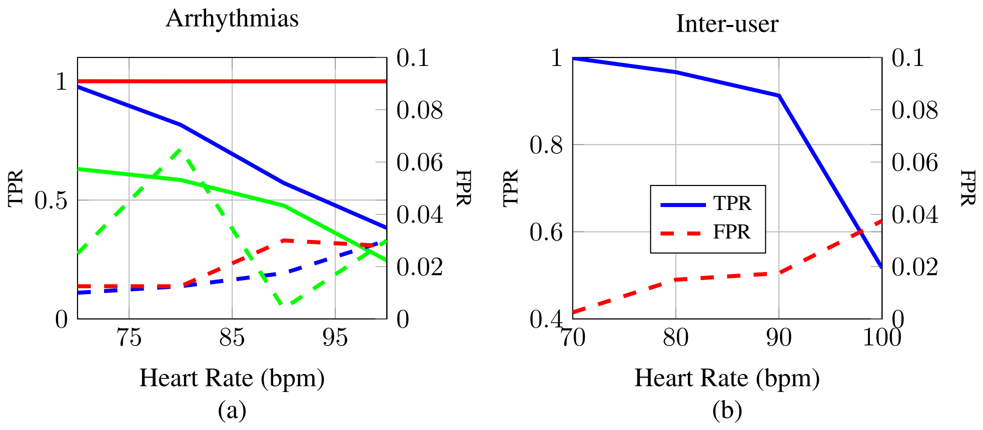


**Figure 10.4:** *Results of online monitoring experiments, where the threshold* $\gamma_{u,r_0}$ *has been set to achieve a desired FPR* $\alpha = 0.01$. *(a) FPR and TPR computed on the 3 patients affected by cardiovascular diseases in the B2B dataset. Different colors correspond to different users. (b) Median FPR and TPR in the inter-user anomaly detection experiments on the B2B dataset.*

CHAPTER 11

# Convolutional Sparse Representations

Convolutional sparse representations [120] are translation-invariant representations that extend traditional sparsity. In this chapter we investigate the use these representations as image models, and in particular we address the denoising task, as it is a customary testbed in image-processing literature. In particular, we compare two denoising approaches performed by computing a sparse representation of the noisy image w.r.t. a translation-invariant dictionary. The classical approach is cycle spinning [119], which aggregates partial estimates each of which is sparse w.r.t. a different translate of the basis. In contrast, convolutional sparse representations involves a global optimization over the entire dictionary. Intuitively, the global optimization might be expected to yield representations that are better suited for denoising. Consider the case of an image that admits a very sparse representation w.r.t. one of the orthonormal bases among the shifted copies in the dictionary: the global optimization would yield a very sparse estimate by activating only few atoms from that particular basis, while cycle spinning would aggregate all partial estimates from other shifted bases too, which might not be as sparse. Our goal is to address this absence by investigating the recent convolutional sparse representations in a careful comparison against the now-classical method of wavelet cycle spinning.

We formulate the denoising problem in Section 11.1, then we describe in details the two considered solutions in Section 11.2. Finally, we presents the results of our experimental campaign in Section 11.3.1.

## 11.1 Image Denoising

The input noisy image $\mathbf{s} \in \mathbb{R}^d$ corrupted by additive white Gaussian noise (AWGN) is modeled as

$$\mathbf{s} = \mathbf{y} + \boldsymbol{\eta}\,, \qquad \boldsymbol{\eta} \sim \mathcal{N}(0, \sigma^2)\,, \tag{11.1}$$

where $\mathbf{y} \in \mathbb{R}^d$ denotes the unknown noise-free image.

We consider denoising methods that approximate $\mathbf{y}$ as a linear combination $\widehat{\mathbf{y}}$ of atoms from a redundant set of generators of $\mathbb{R}^d$ that is formed by the union of all shifted copies $D_1, \ldots, D_d$ of a given orthonormal basis $D_1 \in \mathbb{R}^{d \times d}$:

$$\widehat{\mathbf{y}} = D\widehat{\mathbf{x}}\,, \qquad D = (\, D_1 \cdots D_d) \in \mathbb{R}^{d \times d^2}, \tag{11.2}$$

where $\widehat{\mathbf{x}} \in \mathbb{R}^{d^2}$ is the coefficient vector. Such redundant systems are typically used for building translation-invariant approximations of signals and images. There are two major approaches for solving (11.2), described below.

### 11.1.1 Aggregation of Partial Estimates

Techniques such as *cycle-spinning* [119] seek sparsity w.r.t. each orthonormal basis $D_i$, solving a penalized problem

$$\widehat{\mathbf{x}}_i = \underset{\mathbf{u} \in \mathbb{R}^d}{\arg\min} \frac{1}{2} \|D_i \mathbf{u} - \mathbf{s}\|_2^2 + \lambda \mathcal{R}(\mathbf{u})\,, \quad i \in \{1, \ldots, N\}, \tag{11.3}$$

where $\mathcal{R}(\cdot)$ is a regularization term promoting sparsity of the solution. Since each $D_i$ is orthonormal, problem (11.3) is equivalent to

$$\widehat{\mathbf{x}}_i = \underset{\mathbf{u} \in \mathbb{R}^d}{\arg\min} \frac{1}{2} \|\mathbf{u} - D_i^T \mathbf{s}\|_2^2 + \lambda \mathcal{R}(\mathbf{u})\,, \quad i \in \{1, \ldots, N\}, \tag{11.4}$$

so that the solution $\widehat{\mathbf{x}}_i$ is given by the proximal map [194] of the regularization function $\lambda \mathcal{R}(\cdot)$.

The final estimate $\widehat{\mathbf{y}}_{\mathsf{aggr}}$ is obtained aggregating the $N$ estimates $D_i \widehat{\mathbf{x}}_i$:

$$\widehat{\mathbf{y}}_{\mathsf{aggr}} = \frac{1}{d} \sum_{i=1}^{d} D_i \widehat{\mathbf{x}}_i = D \frac{\left( \widehat{\mathbf{x}}_0^T \cdots \widehat{\mathbf{x}}_d^T \right)^T}{d} = D \widehat{\mathbf{x}}_{\mathsf{aggr}}\,. \tag{11.5}$$

We refer to (11.5) as the *aggregation* of partial estimates.

### 11.1.2 Global Optimization

An obvious, but more computationally expensive alternative defines a single estimate by solving a *global* optimization that jointly considers all the possible shifts of $D_1$ :

$$\widehat{\mathbf{x}}_{\mathsf{glob}} = \arg\min_{\mathbf{x}\in\mathbb{R}^{d^2}} \frac{1}{2}\|D\mathbf{x} - \mathbf{s}\|_2^2 + \lambda\mathcal{R}(\mathbf{x})\,. \tag{11.6}$$

This problem can be also formulated in a convolutional form [195], replacing $D\mathbf{x}$ by convolutions against $M \leq N$ filters:

$$\widehat{\mathbf{x}}_{\mathsf{glob}} = \arg\min_{\mathbf{x}\in\mathbb{R}^{d^2}} \frac{1}{2}\|\sum_{m=1}^{M} \mathbf{d}_m * \mathbf{x}_{[m]} - \mathbf{s}\|_2^2 + \lambda\mathcal{R}(\mathbf{x})\,, \tag{11.7}$$

where $*$ denotes the convolution operator, $\mathbf{d}_m$ denotes the $m^{\text{th}}$ column of $D_1$ that is used as a linear filter in the convolution, $\mathbf{x}_{[m]} \in \mathbb{R}^d$ is a subvector of $\mathbf{x}$ with $\mathbf{x}_{[m]}(j) = \mathbf{x}(m+(j-1)N)$, $j \in \{1, \dots, N\}$, and $\mathbf{x}_{[m]} \equiv \mathbf{0}$ for $m > M$. The number of filters, $M$, involved in the convolutional representation (11.7) can be smaller than $N$ in cases where $D_1$ contains shifted versions of the same column, e.g. when $D_1^T$ is a wavelet basis. In these cases, we keep only those $M \leq N$ columns of $D_1$ that are distinct modulo shifts, thus that correspond to different convolutional filters. The final estimate is then given by

$$\widehat{\mathbf{y}}_{\mathsf{glob}} = D\widehat{\mathbf{x}}_{\mathsf{glob}} = \sum_{m=1}^{M} \mathbf{d}_m * \widehat{\mathbf{x}}_{[m]}\ , \tag{11.8}$$

where the *coefficient map* $\widehat{\mathbf{x}}_{[m]}$ is the subvector gathering the representation coefficients associated with $\mathbf{d}_m$, i.e. $\widehat{\mathbf{x}}_{[m]}(j) = \widehat{\mathbf{x}}_{\mathsf{glob}}(m+(j-1)N)$, $j \in \{1, \dots, N\}$.

### 11.1.3 Our Analysis

Our goal is to compare these two approaches, determining whether global sparsity is a desirable property, and under what conditions the global optimization provides a better solution to the denoising problem. First, we primarily consider convex optimization problems, adopting the $\ell_1$-norm as sparsity-promoting prior, for which a global minimum can be computed. Second, while the global optimization approach in the form of convolutional sparse representations has typically been applied with learned dictionaries, to fairly compare the two approaches we consider a wavelet dictionary $D_1$, which is fixed and not adaptively learned from training data.

Third, to further investigate problems (11.3) and (11.6), we decompose the mean squared error (MSE) of the obtained solutions into their squared bias and variance components. The former indicates how well the approximation fits the underlying data $\mathbf{y}$ in expectation, while the latter indicates how stable this approximation is w.r.t. different realizations of the random noise $\boldsymbol{\eta}$. Finally, our analysis is performed both on natural images and synthetically generated images admitting an extremely sparse representation w.r.t. $D$.

## 11.2 Optimization

### 11.2.1 Regularization Term

To promote sparsity in the solution of (11.3) and (11.6), one typically adopts $\mathcal{R}(\mathbf{x}) = \|\mathbf{x}\|_0$ or $\mathcal{R}(\mathbf{x}) = \|\mathbf{x}\|_1$ as the regularization term. In these cases, the proximal maps of $\lambda\mathcal{R}$ admit closed form expressions, given by hard- and soft-thresholding [194] respectively. Since $D_i$ is orthonormal, applying these proximal maps directly solves (11.3). Therefore, when $\mathcal{R}(\mathbf{x}) = \|x\|_0$, the solution of (11.3) is $\widehat{\mathbf{x}}_i = \mathcal{H}_\lambda(D_i^T\mathbf{s})$, where the hard-thresholding operator $\mathcal{H}_\lambda$ is defined as

$$[\mathcal{H}_\lambda(\mathbf{u})]_j = u_j \cdot 1_{\{|u_j|>\lambda\}}, \qquad j \in \{1, \ldots, N\}\,. \tag{11.9}$$

Similarly, when $\mathcal{R}(\mathbf{x}) = \|x\|_1$, the solution of (11.3) is obtained as $\widehat{\mathbf{x}}_i = \mathcal{S}_\lambda(D_i^T\mathbf{s})$, where the soft-thresholding operator $\mathcal{S}_\lambda$ is

$$[\mathcal{S}_\lambda(\mathbf{u})]_j = \text{sign}(u_j) \cdot \max(|u_j| - \lambda, 0), \;\; j \in \{1, \ldots, N\}. \tag{11.10}$$

Problem (11.6) can be approached via the Iterative Thresholding Algorithm [196], which alternates the thresholding operator corresponding to the specific regularization term $\mathcal{R}$, with a gradient descent step, where the gradient is computed on the data-fidelity term in (11.6).

We primarily consider $\mathcal{R}(\mathbf{x}) = \|\mathbf{x}\|_1$ since it makes problem (11.6) convex and algorithms like [196] converge to a global minimum. In contrast, when $\mathcal{R}(\mathbf{x}) = \|\mathbf{x}\|_0$ the problem is non-convex, so the Iterative Thresholding Algorithm [196] is guaranteed to converge only to a local minimum.

### 11.2.2 High-pass filtering

In cycle spinning, as in other wavelet approximations, coefficients from the coarsest level are not sparse [197]. Therefore, one typically shrinks only the detail coefficients [119, 198]. This corresponds to not regularizing the approximation coefficients in (11.3). This is not a viable solution for the convolutional case, since if we remove the coefficient map $\mathbf{x}_{[1]}$ corresponding to the approximation filter $\mathbf{d}_1$ from $\mathcal{R}(\mathbf{x})$ in (11.7), then $\widehat{\mathbf{x}}_{[1]}$ is the deconvolution of the noisy $\mathbf{s}$ w.r.t. to $\mathbf{d}_1$. Since this solution leads to a poor estimate $\widehat{\mathbf{y}}_{\mathsf{glob}}$, we do not perform convolutional sparse coding (11.7) directly on $\mathbf{s}$ but rather on a high-pass filtered $\mathbf{s}_h$ (which is common practice in convolutional sparse coding [199, Sec. 3]), computed by setting to 0 the approximation coefficients of the overcomplete wavelet transform $D^T$ of $\mathbf{s}$. This is equivalent to setting

$$\mathbf{s}_h = \mathbf{s} - \mathbf{d}_1 * (\bar{\mathbf{d}}_1 * \mathbf{s}) \ , \tag{11.11}$$

where $\bar{\mathbf{d}}_1$ denote the conjugate filter of $\mathbf{d}_1$. Hence, we exclude $\mathbf{d}_1$ and $\mathbf{x}_{[1]}$ from the data-fidelity and regularization terms in (11.7), and add $\mathbf{s}-\mathbf{s}_h$ back to $\widehat{\mathbf{y}}_{\mathsf{glob}}$ in (11.8). However, the noise affecting the high-pass filtered $\mathbf{s}_h$ is no longer white, but rather coloured by $\mathbf{v} = \boldsymbol{\delta} - \mathbf{d}_1 * \bar{\mathbf{d}}_1$, where $\boldsymbol{\delta}$ is the Dirac impulse, so the power spectrum of $\mathbf{v}*\boldsymbol{\eta}$ should be taken into account when denoising $\mathbf{s}_h$ in (11.6-11.7).

## 11.3 Experiments

We perform denoising experiments on natural images as well as on synthetic data that we specifically generated to admit an extremely sparse representation w.r.t. $D$. We corrupt each image $\mathbf{y}$ according to (11.1) and compute both $\widehat{\mathbf{y}}_{\mathsf{glob}}$ and $\widehat{\mathbf{y}}_{\mathsf{aggr}}$. Experiments are conducted with several noise variances $\sigma^2$, and for each $\sigma^2$ we separately tune the penalty parameter $\lambda$ for both methods to achieve the lowest MSE, averaged over all the considered images.

In our experiments the matrix $D_1$ corresponds to the orthonormal basis of the Daubechies db3 wavelet transform with 4 decomposition levels. In case of $\mathcal{R}(\mathbf{x}) = \|\mathbf{x}\|_1$, the convolutional sparse coding problem (11.6-11.7) can be efficiently solved by means of a formulation of the Alternating Directions Method of Multipliers (ADMM) in the Fourier domain [200]. In our experiments we used the MATLAB implementation of the ADMM algorithm provided in the SPORCO library [177].

### 11.3.1 Experiments on Natural Images

We consider five test images (Lena, Barbara, Man, Peppers, Cameraman), corrupted by noise with standard deviation $\sigma \in \{5, 10, \ldots, 40\}$. Each marker in Figure 11.1(a) represents the PSNR (average over 50 noise realizations) achieved by $\widehat{\mathbf{y}}_{\mathsf{aggr}}$ (vertical coordinate) and $\widehat{\mathbf{y}}_{\mathsf{glob}}$ (horizontal coordinate) for each image and $\sigma$ pair. The markers are very close to the

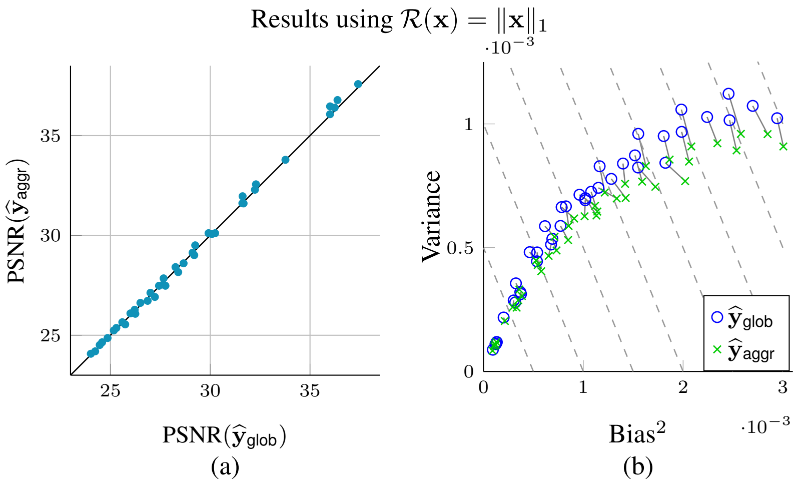


**Figure 11.1:** *Denoising results on natural images for $\mathcal{R}(\mathbf{x}) = \|\mathbf{x}\|_1$. (a) Comparison between the PSNR of $\widehat{\mathbf{y}}_{\text{glob}}$ and $\widehat{\mathbf{y}}_{\text{aggr}}$. Equal PSNR on the diagonal. Each marker corresponds to results for a given image $\mathbf{y}$ and noise level $\sigma$. At low noise level (top-right corner) $\widehat{\mathbf{y}}_{\text{aggr}}$ outperforms $\widehat{\mathbf{y}}_{\text{glob}}$, while the two achieve similar performance as the noise gets stronger. (b) Bias-Variance decomposition of the MSE of $\widehat{\mathbf{y}}_{\text{glob}}$ and $\widehat{\mathbf{y}}_{\text{aggr}}$. Dashed anti-diagonals are the MSE level lines. The relative position of linked circles and crosses shows that although $\widehat{\mathbf{y}}_{\text{glob}}$ and $\widehat{\mathbf{y}}_{\text{aggr}}$ perform similarly, $\widehat{\mathbf{y}}_{\text{glob}}$ features lower bias and higher variance than $\widehat{\mathbf{y}}_{\text{aggr}}$.*

diagonal, indicating that the two methods attain very similar PSNR values, and we can see that only at low noise levels, i.e. where PSNR values are highest, the aggregation of partial estimates slightly outperforms global optimization.

In Figure 11.1(b) we decompose the MSE into its squared bias (horizontal coordinate) and variance (vertical coordinate) components. In these plots, anti-diagonals (dashed lines) are level lines of the MSE, and the blue circles ○ correspond to $\widehat{\mathbf{y}}_{\text{glob}}$, while the green ×-marks to $\widehat{\mathbf{y}}_{\text{aggr}}$; markers corresponding to the same pair $(\mathbf{y}, \sigma)$ are linked by a segment. The relative position of linked circles and crosses confirms that the estimates of the two methods achieve similar PSNR. Most importantly, $\widehat{\mathbf{y}}_{\text{glob}}$ features a lower bias than $\widehat{\mathbf{y}}_{\text{aggr}}$, but has a higher variance.

## 11.3.2 Experiments under Extreme Sparsity

Since the marginal performance gap between $\widehat{\mathbf{y}}_{\text{glob}}$ and $\widehat{\mathbf{y}}_{\text{aggr}}$ may appear unexpected given that global optimization should intuitively be more successful on sparse signals, we investigate how sparse the image really needs to be for our intuition to be correct, and whether the SNR plays any role

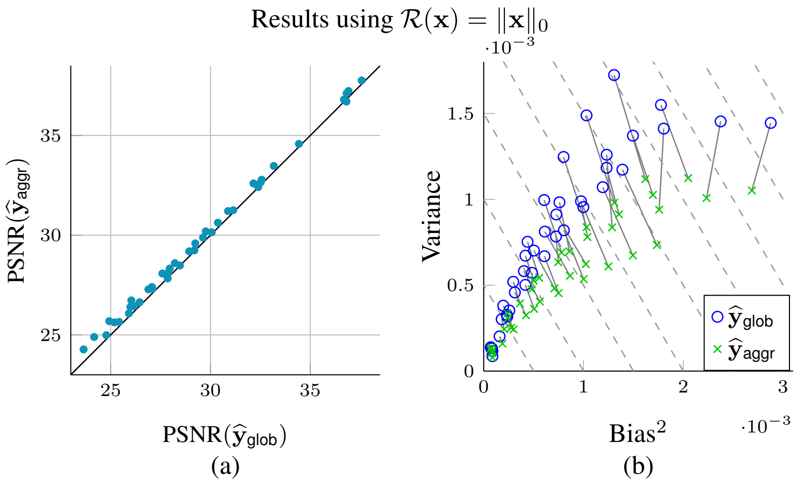


**Figure 11.2:** *Denoising results on natural images for $\mathcal{R}(\mathbf{x}) = \|\mathbf{x}\|_0$ (compare with Figure 11.1). (a) Comparison between the PSNR of $\widehat{\mathbf{y}}_{glob}$ and $\widehat{\mathbf{y}}_{aggr}$ on natural images. In this case, $\widehat{\mathbf{y}}_{glob}$ is clearly outperformed by $\widehat{\mathbf{y}}_{aggr}$, especially under stronger noise. (b) Bias-Variance decomposition of the MSE of $\widehat{\mathbf{y}}_{glob}$ and $\widehat{\mathbf{y}}_{aggr}$. Here $\widehat{\mathbf{y}}_{glob}$ suffers from a much larger variance than $\widehat{\mathbf{y}}_{aggr}$.*

in this question. We synthesize a $128 \times 128$ noise-free image $\mathbf{y} = D\mathbf{x}$ by generating $\mathbf{x}$ with $L$ nonzero components at random positions. Then we corrupt $\mathbf{y}$ with AWGN with standard deviation $\sigma$ such that the SNR of the noisy image $\mathbf{s}$ achieves a target value $\tau$. We consider $L \in \{2^0, 2^1, \dots, 2^{12}\}$ and $\tau \in \{-25, -22.5, \dots, 25\}$, and generate 50 realizations of $\mathbf{y}$ for each pair $(L, \tau)$, and 50 realizations of $\mathbf{s}$ for each such $\mathbf{y}$.

Figure 11.1(c) shows the output SNR difference between $\widehat{\mathbf{y}}_{\text{glob}}$ and $\widehat{\mathbf{y}}_{\text{aggr}}$ when varying the number of nonzero coefficients $L$ and the input SNR $\tau$. These plots indicate that when $L$ is small, $\widehat{\mathbf{y}}_{\text{glob}}$ can achieve much larger SNR than $\widehat{\mathbf{y}}_{\text{aggr}}$ thanks to its lower bias and lower variance. However, when $L$ increases, the SNR gap shrinks and the variance of $\widehat{\mathbf{y}}_{\text{glob}}$ becomes larger than that of $\widehat{\mathbf{y}}_{\text{aggr}}$, especially at high noise levels. This is consistent with the results on natural images, where the two methods attain comparable PSNR and $\widehat{\mathbf{y}}_{\text{glob}}$ features a larger variance. In fact, natural images arguably do not admit extremely sparse representations w.r.t. to $D$, and the two methods perform similarly to large $L$ in the plots of Figure 11.1(c).

## 11.3.3 Results using $\ell_0$ Regularization

Figure 11.2 reports the denoising results for natural and for extremely sparse images using $\mathcal{R}(\mathbf{x}) = \|\mathbf{x}\|_0$ regularization. On natural images, $\widehat{\mathbf{y}}_{\text{glob}}$ suf-

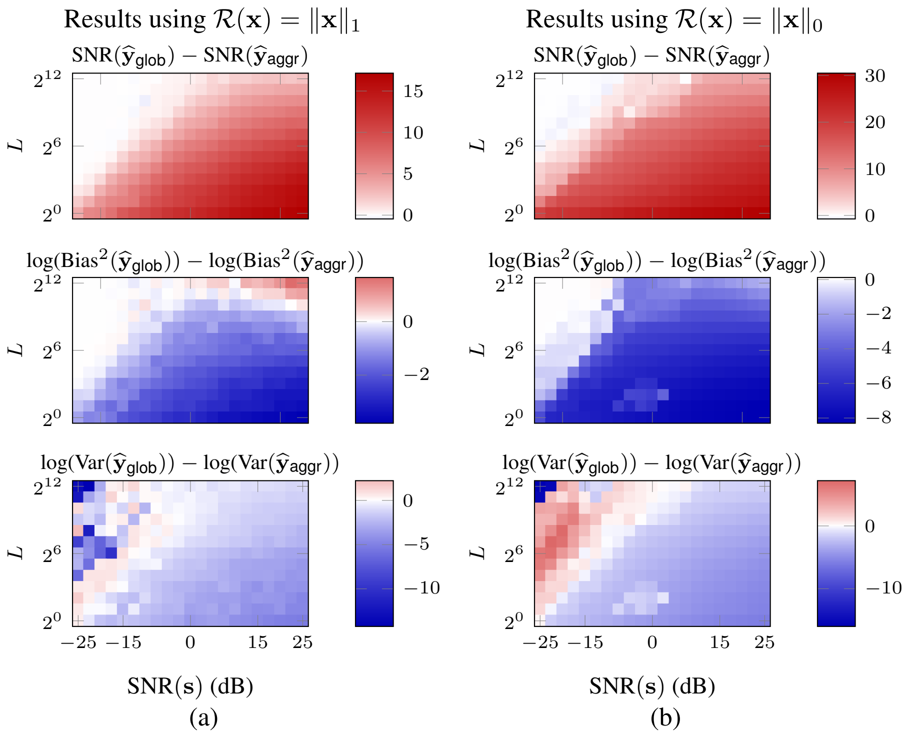


**Figure 11.3:** *Comparison on extremely sparse synthetic images using (a) $\mathcal{R}(\mathbf{x}) = \|\mathbf{x}\|_1$ and (b) $\mathcal{R}(\mathbf{x}) = \|\mathbf{x}\|_0$. The horizontal axis reports the SNR of noisy image $\mathbf{s}$, while the vertical axis the number of nonzero coefficients $L$. The advantage of $\widehat{\mathbf{y}}_{\text{glob}}$ is greatest on very sparse $\mathbf{y}$ (small $L$) and low noise (large $\text{SNR}(\mathbf{s})$).*

fers from a much larger variance than $\widehat{\mathbf{y}}_{\text{aggr}}$, which clearly achieves highest PSNR despite its typically higher bias. Not surprisingly, the performance gap increases with the noise level. By comparing the vertical positions of the markers in Figure 11.1(b) with those in Figure 11.2(b), we can see that the variance of $\widehat{\mathbf{y}}_{\text{aggr}}$ and, especially, of $\widehat{\mathbf{y}}_{\text{glob}}$ is larger when $\mathcal{R}(\mathbf{x}) = \|\mathbf{x}\|_0$ than when $\mathcal{R}(\mathbf{x}) = \|\mathbf{x}\|_1$.

The experiments on synthetic images that admit an extremely sparse representation are summarized in Figure 11.2(c). To deal with the lack of convexity, we initialize the iterative hard-thresholding algorithm [196] with the extremely sparse coefficient vector $\mathbf{x}_{\text{init}}$ that was used to generate $\mathbf{y}$. At least when $L$ is small and the noise is weak, the much lower variance of $\widehat{\mathbf{y}}_{\text{glob}}$ suggests that the estimate $\widehat{\mathbf{x}}_{\text{glob}}$ is very close to $\mathbf{x}_{\text{init}}$ and that [196] practically approaches the global minimum. Thus, on the extremely sparse images, the global optimization is confirmed to be superior to the aggregation of partial estimates also when employing $\mathcal{R}(\mathbf{x}) = \|\mathbf{x}\|_0$.

## 11.4 Discussions

We investigate the benefit of global optimization w.r.t. overcomplete dictionaries over aggregation of partial optimizations w.r.t. each orthogonal subbasis, specifically comparing the modern convolutional sparse representations with the classical cycle spinning. On the one hand, our experiments confirm that solving the global optimization problem leads to estimates that are characterized by a lower bias than the traditional aggregation of partial estimates. On the other hand, we show that the solution of the global optimization are characterized by a larger variance, which makes the two approaches comparable when the input images are not very sparse w.r.t. the dictionary $D$. We speculate that the high redundancy of $D$, which in case of convolutional sparse representations always contains shifted atoms that are highly correlated, is the primary cause of the larger variance.

Our results indicate that solving the computationally demanding global optimization problem only has a clear advantage when $D$ can provide a very sparse representation of the original image. It is unclear to us whether an adaptively learned dictionary can boost the sparsity enough to guarantee an advantage to the global optimization. When the representation is not very sparse, global optimization provides comparable performance to aggregation in the case of $\ell_1$ regularization, and slightly inferior performance in the case of $\ell_0$ regularization. This increased performance gap with $\ell_0$ regularization highlights a practical advantage of the aggregation with orthogonal dictionaries: while switching from $\ell_1$ to $\ell_0$ regularization makes global optimization much more difficult, such a change does not increase the difficulty of optimizing the partial problems involving orthogonal dictionaries. Similarly, the much higher variance for the global solution on natural images when switching from $\ell_1$ to $\ell_0$ regularization is probably due to the non-convex nature of the optimization problem: the solutions we obtain are typically local minima, which can be expected to contribute to the overall increase in the variance. Finally, it should be pointed out that, while to simplify the comparisons we aggregate with uniform weights (11.5) as in classical cycle spinning, the practical advantage of aggregation can be augmented by using sparsity-adaptive weighting [201].

# CHAPTER 12

# Concluding Remarks

In this thesis we address the problem of monitoring a datastream to detect whether the data generating process departs from normal conditions. In practical applications data are high dimensional and feature complex structures, and this raises both theoretical and practical challenges. We address these challenges from two different perspectives, depending on the modeling assumption made on the data generating process.

In the first part of the thesis we model data as realization of random vectors and focus on the change-detection problem. We provide the first rigorous study of the challenges that change-detection algorithms have to face when data dimension scales. Our theoretical and empirical analyses reveal that the popular approach of monitoring the log-likelihood of a multivariate datastream suffers detectability loss when data dimension increases. Remarkably, our analysis and experiments confirm that detectability loss is not a consequence of density-estimation errors – even though these further reduce detectability – but it rather refers to an intrinsic limitation of this change-detection approach. Our theoretical results demonstrate that detectability loss occurs independently on the specific statistical tool used to monitor the log-likelihood and does not depend on the number of input components affected by the change. Our empirical analysis confirms detectability loss also on real-world datastreams.

To investigate the detectability loss problem we propose CCM, a rigorous method to introduce changes in real-world datastreams by roto-translating stationary data. We prove that CCM can successfully control the magnitude of the introduced changes when data can be approximated by a Gaussian mixture. Moreover, our experiments show that experimental practices commonly adopted in the literature for introducing changes cannot control the change magnitude and prevent a fair performance assessment of change-detection algorithms, especially when data become high-dimensional.

To perform change detection we propose QuantTree, an algorithm to build histograms for change detection through an iterative binary splitting of the input space. The specific splitting criteria allows to compute the probability of each bin defined by QuantTree and our theoretical analysis shows that this probability is independent from the distribution of the data in the training set. This fact has the important consequence that statistics defined over QuantTrees are non parametric and thresholds can be estimated through numerical simulation on synthetically generated data. Experiments show that our thresholds (estimated using samples drawn from a univariate uniform distribution) enable a better control of the FPR than thresholds defined by asymptotic approximation of the Pearson statistic, or those estimated by bootstrap on the training set. Moreover, we show that ensembles of histograms can mitigate the detectability loss problem, and can be successfully use for change detection also in high dimensions.

In the second part of the thesis we consider data that feature a complex structure, in particular images acquired by a quality inspection system and ECG signals. We design an anomaly-detection algorithm that learns a dictionary yielding sparse representations of normal data and use these representations to extract low-dimensional indicators to assess whether new data conform or not to the learned dictionary, thus the normal conditions of the process. We remark the importance of a good design of these low-dimensional indicators. In fact, indicator vectors can be modeled as realization of a random vector and increasing their dimension using indicators that do not bring any useful information about data would lead to detectability loss also in anomaly-detection problems. Figure 12.1 shows the ROC curves obtained in an anomaly-detection experiment on SEM images for different dimensions of the indicator vectors. More precisely, we increase the indicator vector computed from the $\ell^1$ norm based sparse coding described in Section 8.2 (that corresponds to $d = 2$) by adding hand-crafted features computed on patch, i.e., the sample mean, the sample variance and the average manigtude of the gradient. The best performance are achieved

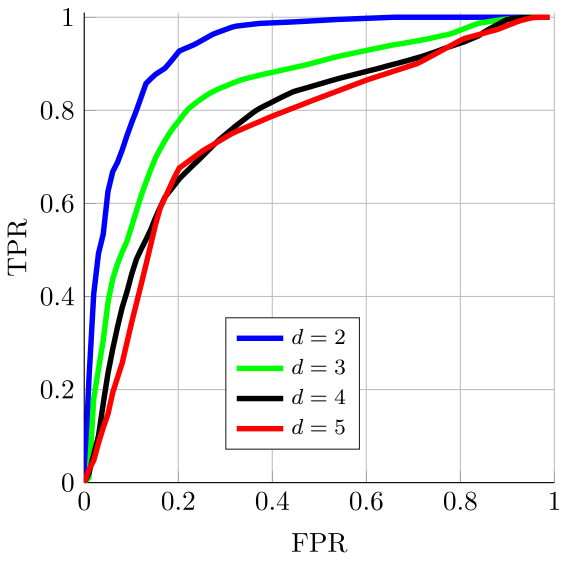


**Figure 12.1:** *Detectability loss experiences in the anomaly detection experiments on the SEM images. We include additional features (sample mean, sample variance and average gradient of the patch) in the bivariate indicators vector* (8.8)*, that contains only the reconstruction error and the $\ell^1$ norm of the representations. For each* $d$ *we plot the average ROC computed over all the combinations of* $d$ *indicators among the 5 considered, by always keeping the original indicators. The best performance are achieved by* $d = 2$*, meaning we are experiencing the detectability loss, as the detectability of the anomalies decreases due to alarger* $d$*.*

when $d = 2$. This means that the additional features do not increase the information carried by our original indicators, thus the (change) detectability of the anomaly decreases, as the anomalies, this the (change) magnitude, remain the same.

We propose two domain adaptation algorithms to adapt the anomaly detector, namely both the learned model and the decision rule, when the process generating normal data changes over time, as this happens in practical scenarios. In particular, we customize our general anomaly-detection algorithm to perform online and long term monitoring of ECG signals directly on a wearable device. We have shown that dictionaries modeling normal heartbeats can be successfully adapted when the heart rate – thus the heartbeat morphology – changes. Moreover, while dictionaries used to detect anomalies have to be user-specific, they can be successfully adapted by user-independent transformations, learned from large and publicly available datasets. Thus, a few minutes of ECG signals acquired in resting conditions are enough to configure the device for long-term monitoring.

Our algorithms have been implemented and successfully tested in a demo device performing online ECG monitoring.

We show that our anomaly-detection algorithm that can also successfully in a quality inspection system to detect defects in nanofibrous materials. Moreover, we make our algorithm scale-invariant by using a multiscale dictionary that aggregates atoms learned from synthetically resized normal images, and performing sparse coding by enforcing the group sparsity of the representations. This regularization term turns to be essential to achieve superior anomaly-detection performance. Our experiments conducted on a large dataset of SEM images show that the proposed algorithm can effectively detect also tiny defects and demonstrate it can effectively handle changes in magnification level that typically occurs in industrial imaging applications.

Finally, we investigate the use of convolutional sparse representation as image model and in particular in white noise denoising. In particular, we show that these translation-invariant representations outperform the traditional sparse representations only when the image admits an extremely sparse representation, while the two approaches attain comparable performance in case of natural images. We explain this phenomenon by separately studying the bias and variance of these solutions, and by noting that the variance of the global solution increases very rapidly as the original signal becomes less and less sparse.

## 12.1 Future works

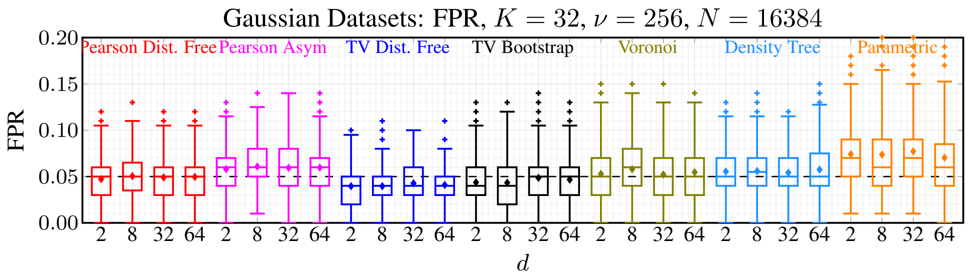
The work presented in this thesis can be extended along different directions. At first, the proposed QuantTree algorithm is very promising as it is one of the very few non-parametric and multivariate change detection algorithms, but many aspects have still to be investigated. While we have shown that ensembles of histograms computed by QuantTree mitigate the detectability loss problem, it is not clear which design is the best one for these ensembles. For simplicity we consider histograms having all the same number of bins, but using histograms with different resolutions might boost the change-detection performance. Another possible extension of our work regards the development of truly sequential monitoring algorithm based on QuantTree and to determine whether this histogram construction scheme has some important implications / consequences to control the false alarm, namely the average run length of the test as in [38].

A research directions that we do not explore in this thesis and is also rather unexplored in the literature is the design of representation learning algorithms specifically targeted for the anomaly-detection task. In fact, almost all anomaly-detection algorithms meant for complex data in the literature resort to unsupervised representations or the extraction of hand-crafted features from data. The recent successes of deep learning in classifications and other supervised tasks suggest that the best performance are achieved when the representations are learned to solve the specific task. To the best of our knowledge the only attempt in this direction is presented in the very recent work [202]. However, several challenges have still to be addressed to make representations learned for anomaly detection effective in practical applications.

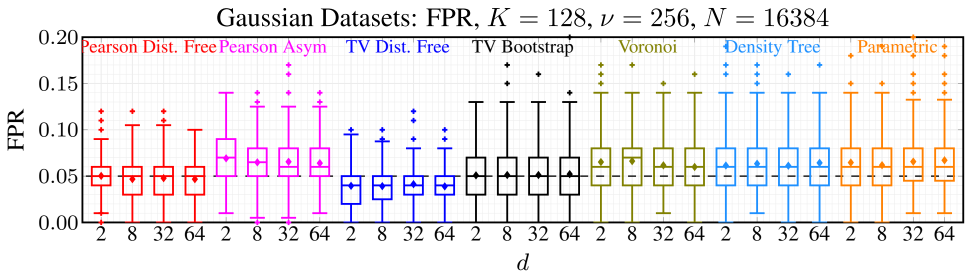
APPENDIX A

# Additional Results on QuantTree

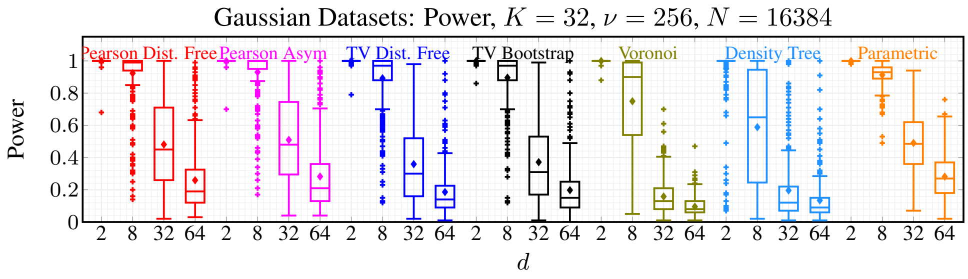
This appendix provides additional results that were not included in Chapter 6. In particular, we report the results we obtain in the experiments performed in 6 on the large configuration.



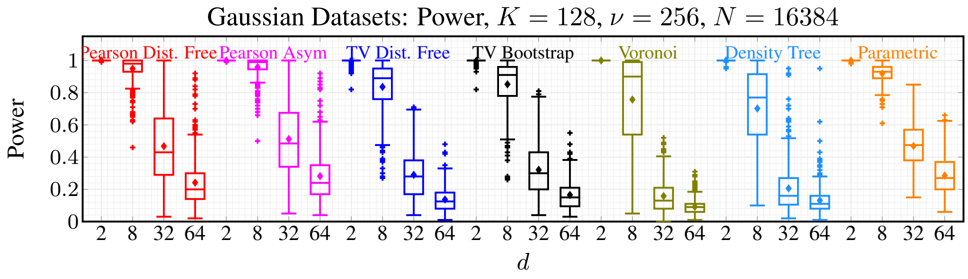
(a)



(b)



(c)



(d)

**Figure A.1:** *Results on synthetically generated datasets using the large configuration. The larger values of $N$ and $\nu$ guarantee higher power and a better control of the FPR than in small configuration. Qualitatively, we see the same trends as in in Figure 6.3.*

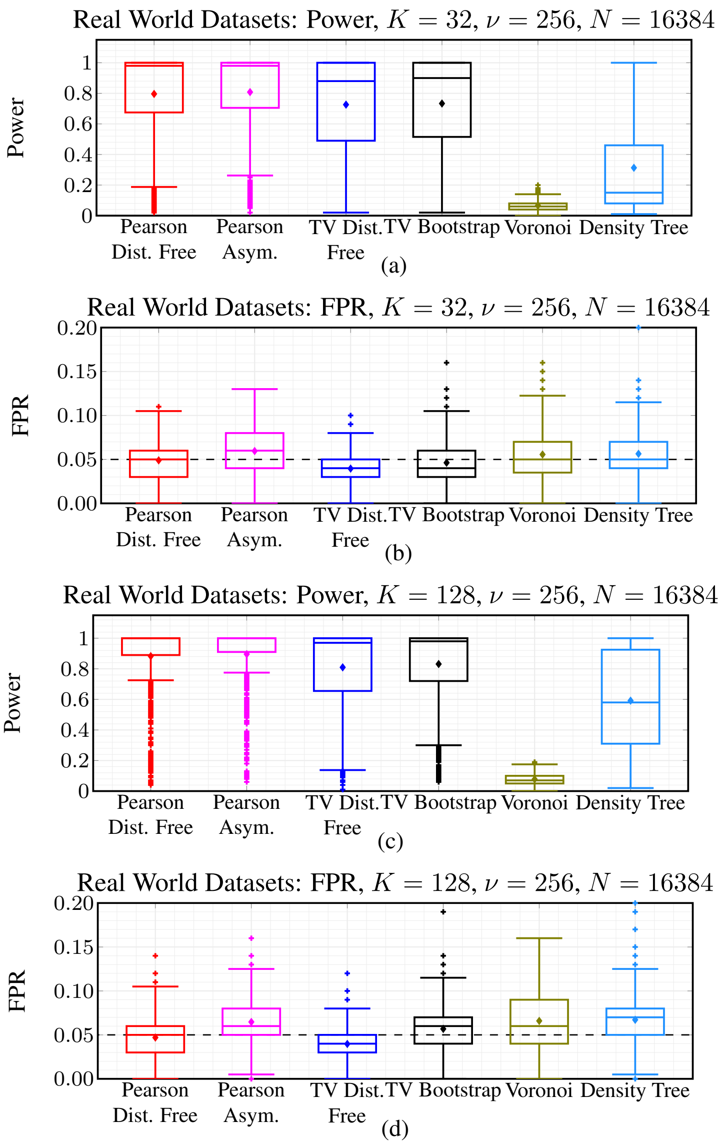


**Figure A.2:** *Results on real world datasets using the large configuration. As in case of the synthetic datasets, the large configuration guarantees higher power and a better control of the FPR than in small one (see Figure 6.4).*

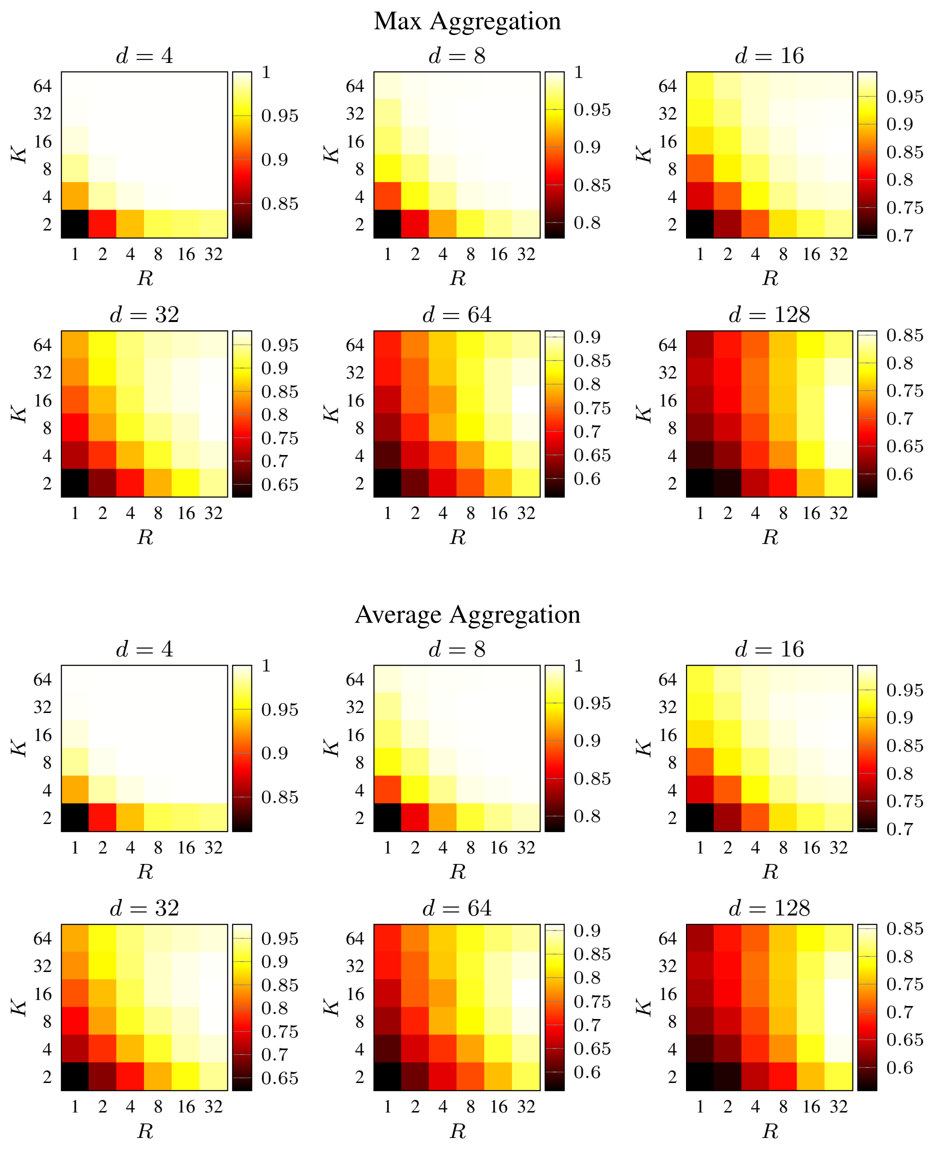
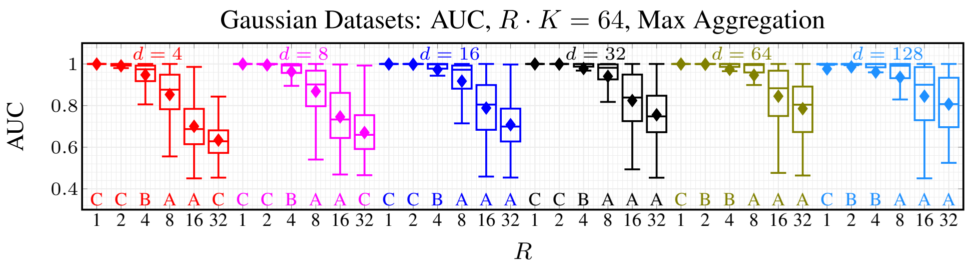
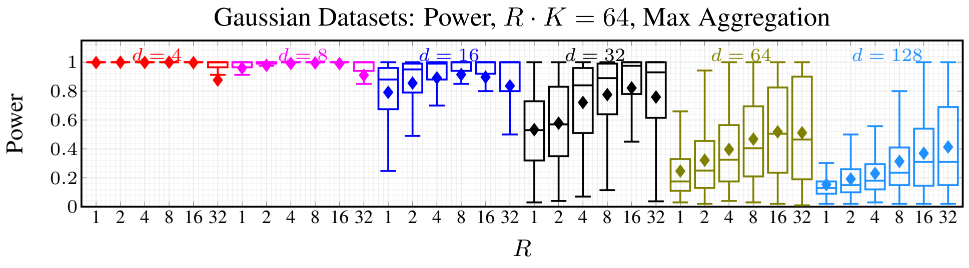


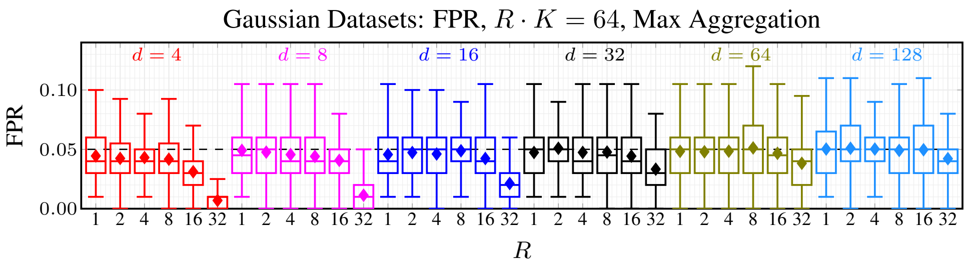
**Figure A.3:** *Results of the exhaustive tests over a mesh of $(R, K)$ pairs for the synthetic datasets, suing the large configuration. Qualitatively, these plots are similar to those obtained using the small configuration (Figure 6.5), although in this case we obtain a better AUC.*



(a)



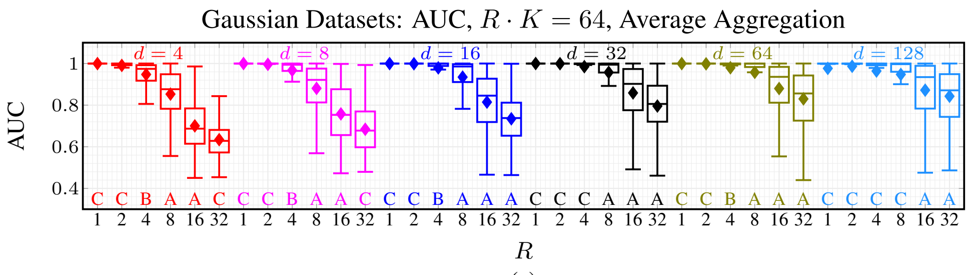
(b)



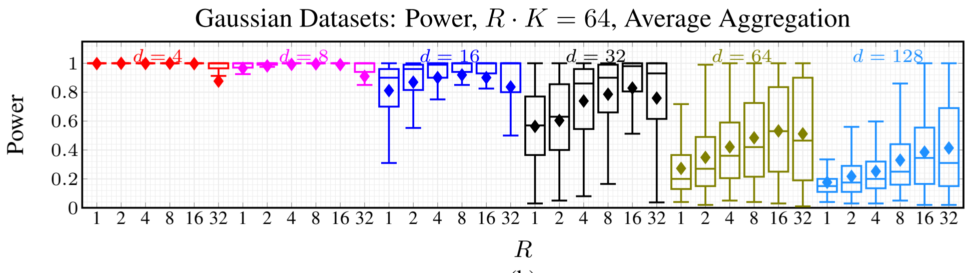
(c)

**Figure A.4:** *Results on the constrained budget tests for the ensemble of histograms with the synthetic datasets in all the tested dimension* $d$*, using the large configuration and the Max Aggregation scheme. Results are qualitatively similar to those in Figures 6.6*

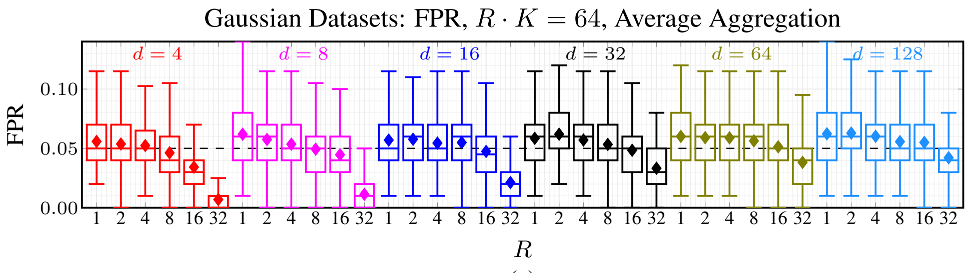
.



(a)



(b)



(c)

**Figure A.5:** *Results on the constrained budget tests for the ensemble of histograms with the synthetic datasets in all the tested dimension d, using the large configuration and the Average Aggregation scheme.Results are qualitatively similar to those in Figures 6.7*

.

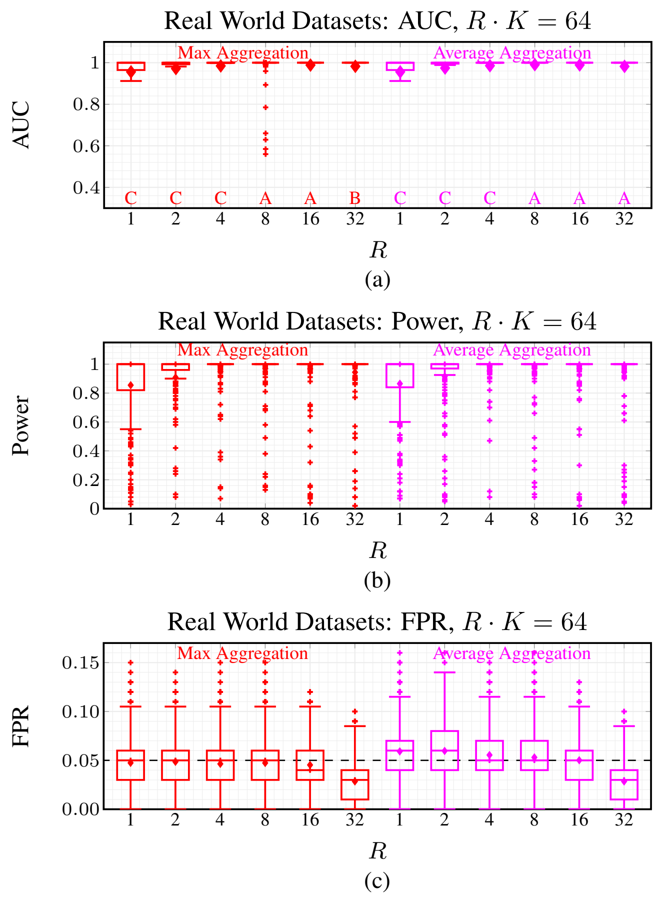


**Figure A.6:** *Results on the constrained budget tests with the real world datasets, using the large configuration and $R \cdot K = 64$ using both Max and Aggregation schemes.*

# Bibliography

[1] Cesare Alippi, Giacomo Boracchi, Diego Carrera, and Manuel Roveri, “Change detection in multivariate datastreams: Likelihood and detectability loss,” in *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, 2016, vol. 2, pp. 1368–1374.

[2] Cesare Alippi, Giacomo Boracchi, and Diego Carrera, “Ccm: Controlling the change magnitude in high dimensional data,” in *Proceedings of the INNS Conference on Big Data*, 2016, pp. 216–225.

[3] Diego Carrera and Giacomo Boracchi, “Generating high-dimensional datastreams for change detection,” *Big data research*, vol. 11, pp. 11–21, 2018.

[4] Giacomo Boracchi, Diego Carrera, Cristiano Cervellera, and Danilo Maccio, “Quanttree: Histograms for change detection in multivariate data streams,” in *Proceedings of the International Conference on Machine Learning (ICML)*, 2018, pp. 638–647.

[5] Diego Carrera, Beatrice Rossi, Daniele Zambon, Pasqualina Fragneto, and Giacomo Boracchi, “Ecg monitoring in wearable devices by sparse models,” in *Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases (ECML-PKDD)*, 2016, pp. 145–160.

[6] Diego Carrera, Beatrice Rossi, Pasqualina Fragneto, and Giacomo Boracchi, “Domain adaptation for online ecg monitoring,” in *Proceedings of the IEEE International Conference on Data Mining (ICDM)*, 2017, pp. 775–780.

[7] Diego Carrera, Beatrice Rossi, Pasqualina Fragneto, and Giacomo Boracchi, “Online anomaly detection for long-term ecg monitoring using wearable devices,” *Pattern Recognition*, vol. 88, pp. 482–492, 2019.

[8] Marco Longoni, Diego Carrera, Beatrice Rossi, Pasqualina Fragneto, Marco Pessione, and Giacomo Boracchi, “A wearable device for online and long-term ecg monitoring.,” in *IJCAI*, 2018, pp. 5838–5840.

[9] Beatrice Rossi, Pasqualina Fragneto, Daniele Zambon, Diego Carrera, and Giacomo Boracchi, “Method for the detecting electrocardiogram anomalies and corresponding system,” Nov. 30 2017, US Patent App. 15/169,184.

[10] Beatrice Rossi, Pasqualina Fragneto, Diego Carrera, Giacomo Boracchi, and Daniele Zambon, “Method for the detecting electrocardiogram anomalies and corresponding system,” Dec. 21 2017, US Patent App. 15/696,051.

[11] Diego Carrera, Fabio Manganini, Giacomo Boracchi, and Ettore Lanzarone, “Defect detection in sem images of nanofibrous materials,” *IEEE Transactions on Industrial Informatics*, vol. 13, no. 2, pp. 551–561, 2017.

[12] Diego Carrera, Giacomo Boracchi, Alessandro Foi, and Brendt Wohlberg, “Scale-invariant anomaly detection with multiscale group-sparse models,” in *Proceedings of the IEEE International Conference on Image Processing (ICIP)*, 2016, pp. 3892–3896.

[13] Diego Carrera, Giacomo Boracchi, Alessandro Foi, and Brendt Wohlberg, “Sparse overcomplete denoising: aggregation versus global optimization,” *IEEE Signal Processing Letters*, vol. 24, no. 10, pp. 1468–1472, 2017.

[14] Ludmila I Kuncheva, “Change detection in streaming multivariate data using likelihood detectors,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 25, no. 5, pp. 1175–1180, 2013.

[15] Xiuyao Song, Mingxi Wu, Christopher Jermaine, and Sanjay Ranka, “Statistical change detection for multi-dimensional data,” in *Proceedings of the International Conference on Knowledge Discovery and Data Mining (KDD)*, 2007, pp. 667–676.

[16] Albert Bifet and Ricard Gavalda, “Learning from time-changing data with adaptive windowing,” in *Proceedings of the SIAM International Conference on Data Mining*, 2007, vol. 7, pp. 2007–2023.

[17] Gordon J Ross, Dimitris K Tasoulis, and Niall M Adams, “Nonparametric monitoring of data streams for changes in location and scale,” *Technometrics*, vol. 53, no. 4, pp. 379–389, 2011.

[18] Michael Elad, *Sparse and redundant representations: from theory to applications in signal and image processing*, Springer, 2010.

[19] Philip de Chazal and Richard B Reilly, “A patient-adapting heartbeat classifier using ecg morphology and heartbeat interval features,” *IEEE Transactions on Biomedical Engineering*, vol. 53, no. 12, pp. 2535–2543, 2006.

[20] Jiapu Pan and Willis J Tompkins, “A real-time qrs detection algorithm,” *IEEE Transactions on Biomedical Engineering*, vol. BME-32, no. 3, pp. 230–236, 1985.

[21] Rudi Hoekema, Gérard JH Uijen, and Adriaan Van Oosterom, “Geometrical aspects of the interindividual variability of multilead ecg recordings,” *IEEE Transactions on Biomedical Engineering*, vol. 48, no. 5, pp. 551–559, 2001.

[22] Yu Hen Hu, Surekha Palreddy, and Willis J Tompkins, “A patient-adaptable ecg beat classifier using a mixture of experts approach,” *IEEE Transactions on Biomedical Engineering*, vol. 44, no. 9, pp. 891–900, 1997.

[23] Serkan Kiranyaz, Turker Ince, and Moncef Gabbouj, “Real-time patient-specific ecg classification by 1-d convolutional neural networks,” *IEEE Transactions on Biomedical Engineering*, vol. 63, no. 3, pp. 664–675, 2016.

[24] Douglas L Mann, Douglas P Zipes, Peter Libby, and Robert O Bonow, *Braunwald’s heart disease: a textbook of cardiovascular medicine*, Elsevier Health Sciences, 2014.

[25] Douglas M Hawkins, PeiHua Qiu, and Wook Kang Chang, “The changepoint model for statistical process control,” *Journal of quality technology*, vol. 35, no. 4, pp. 355–366, 2003.

[26] E. S. Page, “Continuous inspection schemes,” *Biometrika*, vol. 41, no. 1/2, pp. 100–115, 1954.

[27] SW Roberts, “Control chart tests based on geometric moving averages,” *Technometrics*, vol. 1, no. 3, pp. 239–250, 1959.

[28] Walter Andrew Shewhart, *Economic control of quality of manufactured product*, ASQ Quality Press, 1931.

[29] Cesar A Acosta-Mejia, Joseph J Pignatiello, and B Venkateshwara Rao, “A comparison of control charting procedures for monitoring process dispersion,” *IIE Transactions*, vol. 31, no. 6, pp. 569–579, 1999.

[30] Douglas M Hawkins and KD Zamba, “A change-point model for a shift in variance,” *Journal of Quality Technology*, vol. 37, no. 1, pp. 21–31, 2005.

[31] Arthur B Yeh, Richard N Mcgrath, Mark A Sembower, and Qi Shen, “Ewma control charts for monitoring high-yield processes based on non-transformed observations,” *International Journal of Production Research*, vol. 46, no. 20, pp. 5679–5699, 2008.

[32] João Gama, Indrė Žliobaitė, Albert Bifet, Mykola Pechenizkiy, and Abdelhamid Bouchachia, “A survey on concept drift adaptation,” *ACM Computing Surveys (CSUR)*, vol. 46, no. 4, pp. 1–44, 2014.

[33] Joao Gama, Pedro Medas, Gladys Castillo, and Pedro Rodrigues, “Learning with drift detection,” in *Proceedings of the Brazilian Symposium on Artificial Intelligence (SBIA)*, 2004, pp. 286–295.

[34] Cesare Alippi, Giacomo Boracchi, and Manuel Roveri, “Just-in-time classifiers for recurrent concepts,” *IEEE Transactions on Neural Networks and Learning Systems*, vol. 24, no. 4, pp. 620–634, 2013.

[35] Gordon J Ross, Niall M Adams, Dimitris K Tasoulis, and David J Hand, “Exponentially weighted moving average charts for detecting concept drift,” *Pattern Recognition Letters*, vol. 33, no. 2, pp. 191–198, 2012.

[36] Cesare Alippi, Giacomo Boracchi, and Manuel Roveri, “A just-in-time adaptive classification system based on the intersection of confidence intervals rule,” *Neural Networks*, vol. 24, no. 8, pp. 791–800, 2011.

[37] George EP Box and David R Cox, “An analysis of transformations,” *Journal of the Royal Statistical Society. Series B (Methodological)*, pp. 211–252, 1964.

[38] Gordon J Ross and Niall M Adams, “Two nonparametric control charts for detecting arbitrary distribution changes,” *Journal of Quality Technology*, vol. 44, no. 2, pp. 102, 2012.

[39] Douglas M Hawkins and Qiqi Deng, “A nonparametric change-point control chart,” *Journal of Quality Technology*, vol. 42, no. 2, pp. 165–173, 2010.

[40] Yves Lepage, “A combination of wilcoxon’s and ansari-bradley’s statistics,” *Biometrika*, vol. 58, no. 1, pp. 213–217, 1971.

[41] Willem Albers and Wilbert CM Kallenberg, “Cumin charts,” *Metrika*, vol. 70, no. 1, pp. 111–130, 2009.

[42] Harold Hotelling, “Multivariate quality control,” *Techniques of statistical analysis*, 1947.

[43] Ronald B Crosier, “Multivariate generalizations of cumulative sum quality-control schemes,” *Technometrics*, vol. 30, no. 3, pp. 291–303, 1988.

[44] Cynthia A Lowry, William H Woodall, Charles W Champ, and Steven E Rigdon, “A multivariate exponentially weighted moving average control chart,” *Technometrics*, vol. 34, no. 1, pp. 46–53, 1992.

[45] KD Zamba and Douglas M Hawkins, “A multivariate change-point model for statistical process control,” *Technometrics*, vol. 48, no. 4, pp. 539–549, 2006.

[46] KD Zamba and Douglas M Hawkins, “A multivariate change-point model for change in mean vector and/or covariance structure,” *Journal of Quality Technology*, vol. 41, no. 3, pp. 285–303, 2009.

[47] Cesare Alippi, Stavros Ntalampiras, and Manuel Roveri, “An hmm-based change detection method for intelligent embedded sensors,” in *Neural Networks (IJCNN), The 2012 International Joint Conference on*, 2012, pp. 1–7.

[48] Yunus Saatçi, Ryan D Turner, and Carl E Rasmussen, “Gaussian process change point models,” in *Proceedings of the International Conference on Machine Learning (ICML)*, 2010, pp. 927–934.

[49] Georg Krempl, “The algorithm apt to classify in concurrence of latency and drift,” in *Proceedings of the Intelligent Data Analysis (IDA)*, 2011, pp. 222–233.

[50] Tai-Yue Wang and Long-Hui Chen, “Mean shifts detection and classification in multivariate process: a neural-fuzzy approach,” *Journal of Intelligent Manufacturing*, vol. 13, no. 3, pp. 211–221, 2002.

[51] Tuan Duong Nguyen, Marthinus Christoffel Du Plessis, Takafumi Kanamori, and Masashi Sugiyama, “Constrained least-squares density-difference estimation,” *IEICE Trandations on Information and Systems*, vol. 97, no. 7, pp. 1822–1829, 2014.

[52] Ana Justel, Daniel Peña, and Rubén Zamar, “A multivariate kolmogorov-smirnov test of goodness of fit,” *Statistics & Probability Letters*, vol. 35, no. 3, pp. 251–259, 1997.

[53] Alexandre Lung-Yut-Fong, Céline Lévy-Leduc, and Olivier Cappé, “Robust changepoint detection based on multivariate rank statistics,” in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2011, pp. 3608–3611.

[54] Mark F Schilling, “Multivariate two-sample tests based on nearest neighbors,” *Journal of the American Statistical Association*, vol. 81, no. 395, pp. 799–806, 1986.

[55] Gregory Ditzler and Robi Polikar, “Hellinger distance based drift detection for nonstationary environments,” in *Proceedings of the IEEE Symposium on Computational Intelligence in Dynamic and Uncertain Environments (CIDUE)*, 2011, pp. 41–48.

[56] Giacomo Boracchi, Cristiano Cervellera, and Danilo Macciò, “Uniform histograms for change detection in multivariate data,” in *Proceedings on the IEEE International Joint Conference on Neural Networks (IJCNN)*, 2017, pp. 1732–1739.

[57] Raquel Sebastião, João Gama, Pedro Pereira Rodrigues, and João Bernardes, “Monitoring incremental histogram distribution for change detection in data streams,” in *Knowledge discovery from sensor data*, pp. 25–42. Springer, 2010.

[58] Abdulhakim A Qahtan, Basma Alharbi, Suojin Wang, and Xiangliang Zhang, “A pca-based change detection framework for multidimensional data streams: Change detection in multidimensional data streams,” in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, 2015, pp. 935–944.

[59] Tamraparni Dasu, Shankar Krishnan, Suresh Venkatasubramanian, and Ke Yi, “An information-theoretic approach to detecting changes in multi-dimensional data streams,” in *Proceedings of the Symposium on the Interface of Statistics, Computing Science, and Applications*, 2006.

[60] Jon Louis Bentley, “Multidimensional binary search trees used for associative searching,” *Communications of the ACM*, vol. 18, no. 9, pp. 509–517, 1975.

[61] Jerome H Friedman, Jon Louis Bentley, and Raphael Ari Finkel, “An algorithm for finding best matches in logarithmic expected time,” *ACM Transactions on Mathematical Software*, vol. 3, no. 3, pp. 209–226, 1977.

[62] Leo Breiman, *Classification and regression trees*, Routledge, 2017.

[63] Parikshit Ram and Alexander G Gray, “Density estimation trees,” in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, 2011, pp. 627–635.

[64] Erich L Lehmann and Joseph P Romano, *Testing statistical hypotheses*, Springer, 2006.

[65] Jerome Friedman, Trevor Hastie, and Robert Tibshirani, *The elements of statistical learning*, Springer, 2001.

[66] Leo Breiman, “Bagging predictors,” *Machine learning*, vol. 24, no. 2, pp. 123–140, 1996.

[67] Yoav Freund and Robert E Schapire, “A decision-theoretic generalization of on-line learning and an application to boosting,” *Journal of computer and system sciences*, vol. 55, no. 1, pp. 119–139, 1997.

[68] Leo Breiman, “Random forests,” *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.

[69] Chanop Silpa-Anan and Richard Hartley, “Optimised kd-trees for fast image descriptor matching,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008, pp. 1–8.

[70] Manuel Roveri and Francesco Trovò, “An ensemble of hmms for cognitive fault detection in distributed sensor networks,” in *Proceedings of the International Conference on Artificial Intelligence Applications and Innovations (IFIP)*, 2014, pp. 90–100.

[71] William J Faithfull, Juan J Rodríguez, and Ludmila I Kuncheva, “Combining univariate approaches for ensemble change detection in multivariate data,” *Information Fusion*, vol. 45, pp. 202–214, 2019.

[72] Joe H Sullivan and William H Woodall, “Change-point detection of mean vector or covariance matrix shifts using multivariate individual observations,” *IIE Transactions*, vol. 32, no. 6, pp. 537–549, 2000.

[73] Thomas M Cover and Joy A Thomas, *Elements of information theory*, John Wiley & Sons, 2012.

[74] Bernard L Welch, “The generalization ofstudent’s’ problem when several different population variances are involved,” *Biometrika*, vol. 34, no. 1/2, pp. 28–35, 1947.

[75] Arthur Zimek, Erich Schubert, and Hans-Peter Kriegel, “A survey on unsupervised outlier detection in high-dimensional numerical data,” *Statistical Analysis and Data Mining: The ASA Data Science Journal*, vol. 5, no. 5, pp. 363–387, 2012.

[76] M. Lichman, “UCI machine learning repository,” 2013.

[77] Ludmila I Kuncheva and William J Faithfull, “Pca feature extraction for change detection in multidimensional unlabeled data,” *IEEE Transactions on Neural Networks and Learning Systems*, vol. 25, no. 1, pp. 69–80, 2014.

[78] Giacomo Boracchi and Manuel Roveri, “Exploiting self-similarity for change detection,” in *Proceedings of the IEEE International Joint Conference onNeural Networks (IJCNN)*, 2014, pp. 3339–3346.

[79] Hoang-Vu Nguyen and Jilles Vreeken, “Linear-time detection of non-linear changes in massively high dimensional time series,” in *Proceedings of the SIAM International Conference on Data Mining*, 2016, pp. 828–836.

[80] Anand M Narasimhamurthy and Ludmila I Kuncheva, “A framework for generating data to simulate changing environments.,” in *Proceedings of the IASTED International Multi-Conference: artificial Intelligence and Applications (AIA)*, 2007, pp. 384–389.

[81] Leandro L Minku, Allan P White, and Xin Yao, “The impact of diversity on online ensemble learning in the presence of concept drift,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, no. 5, pp. 730–742, 2010.

[82] Patrick Lindstrom, Sarah Jane Delany, and Brian Mac Namee, “Autopilot: Simulating changing concepts in real data,” in *Proceedings of the Irish Conference on Artificial Intelligence and Cognitive Science (AICS)*, 2008, pp. 1–10.

[83] Geoffrey McLachlan and David Peel, *Finite mixture models*, John Wiley & Sons, 2004.

[84] Lloyd Trefethen and David Bau, *Numerical linear algebra*, Siam, 1997.

[85] Cesare Alippi, *Intelligence for Embedded Systems: A Methodological Approach*, Springer, 2014.

[86] Walter Rudin et al., *Principles of mathematical analysis*, McGraw-Hill New York, 1964.

[87] Richard L Burden and J Douglas Faires, *Numerical analysis. 2001*, Brooks/Cole, USA, 2001.

[88] Heinz Bauer, *Measure and integration theory*, Walter de Gruyter, 2001.

[89] Fernando Pérez-Cruz, “Estimation of information theoretic measures for continuous random variables,” in *Advances in Neural Information Processing Systems (NIPS)*, 2009, pp. 1257–1264.

[90] Nikolai Leonenko, Luc Pronzato, Vippal Savani, et al., “A class of rényi information estimators for multidimensional densities,” *The Annals of Statistics*, vol. 36, no. 5, pp. 2153–2182, 2008.

[91] XuanLong Nguyen, Martin J Wainwright, and Michael I Jordan, “Estimating divergence functionals and the likelihood ratio by convex risk minimization,” *IEEE Transactions on Information Theory*, vol. 56, no. 11, pp. 5847–5861, 2010.

[92] Qing Wang, Sanjeev R Kulkarni, and Sergio Verdú, “A nearest-neighbor approach to estimating divergence between continuous random vectors,” in *Proceedings of the IEEE International Symposium on Information Theory (ISIT)*, 2006, pp. 242–246.

[93] Narayanaswamy Balakrishnan and Calyampudi Radhakrishna Rao, *Order Statistics: Theory & Methods*, Elsevier Amsterdam, 1998.

[94] Laura Forsberg White, Marco Bonetti, and Marcello Pagano, “The choice of the number of bins for the m statistic,” *Computational statistics & data analysis*, vol. 53, no. 10, pp. 3640–3649, 2009.

[95] Arthur Zimek, Ricardo JGB Campello, and Jörg Sander, “Ensembles for unsupervised outlier detection: challenges and research questions a position paper,” *ACM SIGKDD Explorations Newsletter*, vol. 15, no. 1, pp. 11–22, 2014.

[96] Andrea Dal Pozzolo, Olivier Caelen, Reid A Johnson, and Gianluca Bontempi, “Calibrating probability with undersampling for unbalanced classification,” in *Proceedings of the IEEE Symposium Series on Computational Intelligence and Data Mining (CIDM)*, 2015, pp. 159–166.

[97] A Criminisi, J Shotton, and E Konukoglu, “Decision forests for classification, regression, density estimation, manifold learning and semi-supervised learning,” *Microsoft Research*, 2011.

[98] Janez Demšar, “Statistical comparisons of classifiers over multiple data sets,” *Journal of Machine learning research*, vol. 7, no. Jan, pp. 1–30, 2006.

[99] Yoshua Bengio, Aaron Courville, and Pierre Vincent, “Representation learning: A review and new perspectives,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 8, pp. 1798–1828, 2013.

[100] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton, “Imagenet classification with deep convolutional neural networks,” in *Advances in Neural Information Processing Systems (NIPS)*, 2012, pp. 1097–1105.

[101] Sebastian Mika, Bernhard Schölkopf, Alex J Smola, Klaus-Robert Müller, Matthias Scholz, and Gunnar Rätsch, "Kernel pca and de-noising in feature spaces," in *Advances in neural information processing systems (NIPS)*, 1999, pp. 536–542.

[102] Matthew Brand, "Charting a manifold," in *Advances in neural information processing systems (NIPS)*, 2003, pp. 985–992.

[103] Geoffrey E Hinton and Sam T Roweis, "Stochastic neighbor embedding," in *Advances in neural information processing systems (NIPS)*, 2003, pp. 857–864.

[104] Davide Eynard, Artiom Kovnatsky, Michael Bronstein, Klaus Glashoff, and Alexander Bronstein, "Multimodal manifold analysis by simultaneous diagonalization of laplacians," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 12, pp. 2505–2517, 2015.

[105] David L Donoho, Iain M Johnstone, Gérard Kerkyacharian, and Dominique Picard, "Wavelet shrinkage: asymptopia?," *Journal of the Royal Statistical Society*, vol. 57, no. 2, pp. 301–369, 1995.

[106] Michal Aharon, Michael Elad, and Alfred Bruckstein, "K-svd: An algorithm for designing overcomplete dictionaries for sparse representation," *IEEE Transactions on Signal Processing*, vol. 54, no. 11, pp. 4311–4322, 2006.

[107] Julien Mairal, Francis Bach, Jean Ponce, and Guillermo Sapiro, "Online dictionary learning for sparse coding," in *Proceedings of the International Conference on Machine Learning (ICML)*, 2009, pp. 689–696.

[108] Michael Elad and Michal Aharon, "Image denoising via sparse and redundant representations over learned dictionaries," *IEEE Transactions on Image Processing*, vol. 15, no. 12, pp. 3736–3745, 2006.

[109] Julien Mairal, Michael Elad, and Guillermo Sapiro, "Sparse representation for color image restoration," *IEEE Transactions on Image Processing*, vol. 17, no. 1, pp. 53–69, 2008.

[110] Julien Mairal, Francis Bach, Jean Ponce, Guillermo Sapiro, and Andrew Zisserman, "Non-local sparse models for image restoration," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2009, pp. 2272–2279.

[111] Julien Mairal, Francis Bach, and Jean Ponce, "Task-driven dictionary learning," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 4, pp. 791–804, 2012.

[112] Julien Mairal, Jean Ponce, Guillermo Sapiro, Andrew Zisserman, and Francis Bach, "Supervised dictionary learning," in *Advances in Neural Information Processing Systems (NIPS)*, 2009, pp. 1033–1040.

[113] Ignacio Ramirez, Pablo Sprechmann, and Guillermo Sapiro, "Classification and clustering via dictionary learning with structured incoherence and shared features," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010, pp. 3501–3508.

[114] Julien Mairal, Guillermo Sapiro, and Michael Elad, "Learning multiscale sparse representations for image and video restoration," *Multiscale Modeling & Simulation*, vol. 7, no. 1, pp. 214–241, 2008.

[115] Rémi Gribonval, "Fast matching pursuit with a multiscale dictionary of gaussian chirps," *IEEE Transactions on Signal Processing*, vol. 49, no. 5, pp. 994–1001, 2001.

[116] Bruno A Olshausen, Phil Sallee, and Michael S Lewicki, "Learning sparse image codes using a wavelet pyramid architecture," in *Advances in Neural Information Processing Systems (NIPS)*, 2001, pp. 887–893.

[117] Phil Sallee and Bruno A Olshausen, “Learning sparse multiscale image representations,” in *Advances in Neural Information Processing Systems (NIPS)*, 2002, pp. 1327–1334.

[118] James Michael Hughes, Daniel N Rockmore, and Yang Wang, “Bayesian learning of sparse multiscale image representations,” *IEEE Transactions on Image Processing*, vol. 22, no. 12, pp. 4972–4983, 2013.

[119] R. R. Coifman and D. L. Donoho, “Translation-invariant de-noising,” in *Wavelets and Statistics*, pp. 125–150. Springer, 1995.

[120] Brendt Wohlberg, “Efficient algorithms for convolutional sparse representations,” *IEEE Transactions on Image Processing*, vol. 25, no. 1, pp. 301–315, 2016.

[121] Shuhang Gu, Wangmeng Zuo, Qi Xie, Deyu Meng, Xiangchu Feng, and Lei Zhang, “Convolutional sparse coding for image super-resolution,” in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2015, pp. 1823–1831.

[122] Tran Minh Quan and Won-Ki Jeong, “Compressed sensing reconstruction of dynamic contrast enhanced MRI using GPU-accelerated convolutional sparse coding,” in *2016 IEEE 13th International Symposium on Biomedical Imaging (ISBI)*, 2016.

[123] He Zhang and Vishal Patel, “Convolutional sparse coding-based image decomposition,” in *British Machine Vision Conference (BMVC)*, 2016.

[124] Yu Liu, Xun Chen, Rabab K. Ward, and Z. Jane Wang, “Image fusion with convolutional sparse representation,” *IEEE Signal Processing Letters*, 2016.

[125] Kévin Degraux, Ulugbek S Kamilov, Petros T Boufounos, and Dehong Liu, “Online convolutional dictionary learning for multimodal imaging,” in *Proceedings of the IEEE International Conference on Image Processing (ICIP)*, 2017, pp. 1617–1621.

[126] Sinno Jialin Pan and Qiang Yang, “A survey on transfer learning,” *IEEE Transactions on knowledge and data engineering*, vol. 22, no. 10, pp. 1345–1359, 2010.

[127] Anastasia Krithara and Georgios Paliouras, “Tl-plsa: Transfer learning between domains with different classes,” in *Proceedings of the IEEE International Conference on Data Mining (ICDM)*, 2013, pp. 419–427.

[128] Jiayuan Huang, Arthur Gretton, Karsten M Borgwardt, Bernhard Schölkopf, and Alex J Smola, “Correcting sample selection bias by unlabeled data,” in *Advances in Neural Information Processing Systems (NIPS)*, 2007, pp. 601–608.

[129] Ming Shao, Carlos Castillo, Zhenghong Gu, and Yun Fu, “Low-rank transfer subspace learning,” in *Proceedings of the IEEE International Conference on Data Mining (ICDM)*, 2012, pp. 1104–1109.

[130] Rajat Raina, Alexis Battle, Honglak Lee, Benjamin Packer, and Andrew Y Ng, “Self-taught learning: transfer learning from unlabeled data,” in *Proceedings of the International Conference on Machine Learning (ICML)*, 2007, pp. 759–766.

[131] Qiang Qiu, Vishal M Patel, Pavan Turaga, and Rama Chellappa, “Domain adaptive dictionary learning,” in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2012, pp. 631–645.

[132] Jie Ni, Qiang Qiu, and Rama Chellappa, “Subspace interpolation via dictionary learning for unsupervised domain adaptation,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2013, pp. 692–699.

[133] Sumit Shekhar, Vishal M Patel, Hien V Nguyen, and Rama Chellappa, “Generalized domain-adaptive dictionaries,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2013, pp. 361–368.

[134] Baoyao Yang, Andy J Ma, and Pong C Yuen, "Learning domain-shared group-sparse representation for unsupervised domain adaptation," *Pattern Recognition*, vol. 81, pp. 615–632, 2018.

[135] Varun Chandola, Arindam Banerjee, and Vipin Kumar, "Anomaly detection: A survey," *ACM Computing Surveys (CSUR)*, vol. 41, no. 3, pp. 15, 2009.

[136] Marco AF Pimentel, David A Clifton, Lei Clifton, and Lionel Tarassenko, "A review of novelty detection," *Signal Processing*, vol. 99, pp. 215–249, 2014.

[137] Thibaud Ehret, Axel Davy, Jean-Michel Morel, and Mauricio Delbracio, "Image anomalies: a review and synthesis of detection methods," *arXiv preprint arXiv:1808.02564*, 2018.

[138] Bo Du and Liangpei Zhang, "Random-selection-based anomaly detector for hyperspectral imagery," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 49, no. 5, pp. 1578, 2011.

[139] Xianghua Xie and Majid Mirmehdi, "Texems: Texture exemplars for defect detection on random textured surfaces," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 8, pp. 1454–1464, 2007.

[140] Lionel Tarassenko, P Hayton, N Cerneaz, and M Brady, "Novelty detection for the identification of masses in mammograms," in *Proceedings of the International Conference on Artificial Neural Networks (ICANN)*, 1995, pp. 442–447.

[141] Antoni Buades, Bartomeu Coll, and Jean-Michel Morel, "A review of image denoising algorithms, with a new one," *Multiscale Modeling & Simulation*, vol. 4, no. 2, pp. 490–530, 2005.

[142] Stas Goferman, Lihi Zelnik-Manor, and Ayellet Tal, "Context-aware saliency detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 10, pp. 1915–1926, 2012.

[143] Maria Zontak and Israel Cohen, "Defect detection in patterned wafers using anisotropic kernels," *Machine Vision and Applications*, vol. 21, no. 2, pp. 129–141, 2010.

[144] Diederik P Kingma and Max Welling, "Auto-encoding variational bayes," in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2014, pp. 1–14.

[145] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio, "Generative adversarial nets," in *Advances in Neural Information Processing Systems (NIPS)*, 2014, pp. 2672–2680.

[146] Raymond A Yeh, Chen Chen, Teck Yian Lim, Alexander G Schwing, Mark Hasegawa-Johnson, and Minh N Do, "Semantic image inpainting with deep generative models," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 5485–5493.

[147] William Lotter, Gabriel Kreiman, and David Cox, "Deep predictive coding networks for video prediction and unsupervised learning," in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2017, pp. 1–18.

[148] Michael Mathieu, Camille Couprie, and Yann LeCun, "Deep multi-scale video prediction beyond mean square error," *arXiv preprint arXiv:1511.05440*, 2015.

[149] Xun Huang and Serge Belongie, "Arbitrary style transfer in real-time with adaptive instance normalization," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 1510–1519.

[150] Thomas Schlegl, Philipp Seeböck, Sebastian M Waldstein, Ursula Schmidt-Erfurth, and Georg Langs, "Unsupervised anomaly detection with generative adversarial networks to guide marker discovery," in *Proceedings of the International Conference on Information Processing in Medical Imaging (IPMI)*, 2017, pp. 146–157.

[151] Jinwon An and Sungzoon Cho, "Variational autoencoder based anomaly detection using reconstruction probability," *Special Lecture on IE*, vol. 2, pp. 1–18, 2015.

[152] Amir Adler, Michael Elad, Yacov Hel-Or, and Ehud Rivlin, "Sparse coding with anomaly detection," *Journal of Signal Processing Systems*, vol. 79, no. 2, pp. 179–188, 2015.

[153] Bin Zhao, Li Fei-Fei, and Eric P Xing, "Online detection of unusual events in videos via dynamic sparse coding," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011, pp. 3313–3320.

[154] Yang Cong, Junsong Yuan, and Ji Liu, "Sparse reconstruction cost for abnormal event detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011, pp. 3449–3456.

[155] Abdur Rahim Mohammad Forkan, Ibrahim Khalil, Zahir Tari, Sebti Foufou, and Abdelaziz Bouras, "A context-aware approach for long-term behavioural change detection and abnormality prediction in ambient assisted living," *Pattern Recognition*, vol. 48, no. 3, pp. 628–641, 2015.

[156] Leslie A Saxon, "Ubiquitous wireless ecg recording: a powerful tool physicians should embrace," *Journal of Cardiovascular Electrophysiology*, vol. 24, no. 4, pp. 480–483, 2013.

[157] Jenna Wiens and John V Guttag, "Active learning applied to patient-adaptive heartbeat classification," in *Advances in Neural Information Processing Systems (NIPS)*, 2010, pp. 2442–2450.

[158] Stanislaw Osowski, Linh Tran Hoai, and Tomasz Markiewicz, "Support vector machine-based expert system for reliable heartbeat recognition," *IEEE Transactions on Biomedical Engineering*, vol. 51, no. 4, pp. 582–589, 2004.

[159] Muqing Deng, Cong Wang, Min Tang, and Tongjia Zheng, "Extracting cardiac dynamics within ecg signal for human identification and cardiovascular diseases classification," *Neural Networks*, vol. 100, pp. 70–83, 2018.

[160] İnan Güler and Elif Derya Übeylı, "ECG beat classifier designed by combined neural network model," *Pattern recognition*, vol. 38, no. 2, pp. 199–208, 2005.

[161] Wei Jiang and Seong G Kong, "Block-based neural networks for personalized ecg signal classification," *IEEE Transactions Neural Networks*, vol. 18, no. 6, pp. 1750–1761, 2007.

[162] Rashid Ghorbani Afkhami, Ghanbar Azarnia, and Mohammad Ali Tinati, "Cardiac arrhythmia classification using statistical and mixture modeling features of ecg signals," *Pattern Recognition Letters*, vol. 70, pp. 45–51, 2016.

[163] Roshan Joy Martis, Chandan Chakraborty, and Ajoy K Ray, "A two-stage mechanism for registration and classification of ECG using gaussian mixture model," *Pattern Recognition*, vol. 42, no. 11, pp. 2979–2988, 2009.

[164] Eamonn Keogh, Jessica Lin, and Ada Fu, "Hot sax: Efficiently finding the most unusual time series subsequence," in *Proceedings of the IEEE International Conference on Data Mining (ICDM)*, 2005, pp. 226–233.

[165] Mooi Choo Chuah and Fen Fu, "Ecg anomaly detection via time series analysis," in *International Symposium on Parallel and Distributed Processing and Applications*, 2007, pp. 123–135.

[166] Haemwaan Sivaraks and Chotirat Ann Ratanamahatana, "Robust and accurate anomaly detection in ecg artifacts using time series motif discovery," *Computational and mathematical methods in medicine*, vol. 2015, 2015.

[167] Helena Aidos, André Lourenço, Diana Batista, Samuel Rota Bulò, and Ana Fred, "Semi-supervised consensus clustering for ecg pathology classification," in *Proceedings of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML-PKDD)*, 2015, pp. 150–164.

[168] Ruggero Donida Labati, Enrique Muñoz, Vincenzo Piuri, Roberto Sassi, and Fabio Scotti, "Deep-ecg: Convolutional neural networks for ecg biometric recognition," *Pattern Recognition Letters*, 2018.

[169] Pankaj Malhotra, Lovekesh Vig, Gautam Shroff, and Puneet Agarwal, "Long short term memory networks for anomaly detection in time series," in *ESANN*, 2015, pp. 89–94.

[170] Ziad Sankari and Hojjat Adeli, "Heartsaver: a mobile cardiac monitoring system for auto-detection of atrial fibrillation, myocardial infarction, and atrio-ventricular block," *Computers in Biology and Medicine*, vol. 41, no. 4, pp. 211–220, 2011.

[171] Medina Hadjem, Osman Salem, and Farid Naït-Abdesselam, "An ecg monitoring system for prediction of cardiac anomalies using wban," in *Healthcom*, 2014, pp. 441–446.

[172] Yagyensh Chandra Pati, Ramin Rezaiifar, and Perinkulam Sambamurthy Krishnaprasad, "Orthogonal matching pursuit: Recursive function approximation with applications to wavelet decomposition," in *Proceedings of the Asilomar Conference on Signals, Systems and Computers*, 1993, pp. 40–44.

[173] Robert Tibshirani, "Regression shrinkage and selection via the lasso," *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 58, pp. 267–288, 1996.

[174] Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends® in Machine Learning*, vol. 3, no. 1, pp. 1–122, 2011.

[175] Bradley Efron, Trevor Hastie, Iain Johnstone, Robert Tibshirani, et al., "Least angle regression," *The Annals of statistics*, vol. 32, no. 2, pp. 407–499, 2004.

[176] Alfred M Bruckstein, David L Donoho, and Michael Elad, "From sparse solutions of systems of equations to sparse modeling of signals and images," *SIAM review*, vol. 51, no. 1, pp. 34–81, 2009.

[177] Brendt Wohlberg, "SParse Optimization Research COde (SPORCO)," Software library available from `http://purl.org/brendt/software/sporco`, 2017.

[178] Zdravko I Botev, Joseph F Grotowski, Dirk P Kroese, et al., "Kernel density estimation via diffusion," *The Annals of Statistics*, vol. 38, no. 5, pp. 2916–2957, 2010.

[179] Rafael C Gonzalez and Richard E Woods, *Digital Image Processing*, Prentice Hall, 2002.

[180] Ron Rubinstein, Michael Zibulevsky, and Michael Elad, "Efficient implementation of the k-svd algorithm using batch orthogonal matching pursuit," .

[181] George B Moody and Roger G Mark, "The impact of the mit-bih arrhythmia database," *IEEE Engineering in Medicine and Biology Magazine*, vol. 20, no. 3, pp. 45–50, 2001.

[182] Jana Zujovic, Thrasyvoulos N Pappas, and David L Neuhoff, "Structural texture similarity metrics for image analysis and retrieval," *IEEE Transactions on Image Processing*, vol. 22, no. 7, pp. 2545–2558, 2013.

[183] Eero P Simoncelli and William T Freeman, "The steerable pyramid: A flexible architecture for multi-scale derivative computation," in *Proceedings of the IEEE Internation Conference on Image Processing (ICIP)*, 1995, p. 3444.

[184] Scott Shaobing Chen, David L Donoho, and Michael A Saunders, "Atomic decomposition by basis pursuit," *SIAM journal on scientific computing*, vol. 20, no. 1, pp. 33–61, 1998.

[185] Ivana Tošić and Pascal Frossard, "Dictionary learning," *IEEE Signal Processing Magazine*, vol. 28, no. 2, pp. 27–38, 2011.

[186] Ming Yuan and Yi Lin, "Model selection and estimation in regression with grouped variables," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 68, no. 1, pp. 49–67, 2006.

[187] Ary L Goldberger, Luis AN Amaral, Leon Glass, Jeffrey M Hausdorff, Plamen Ch Ivanov, Roger G Mark, Joseph E Mietus, George B Moody, Chung-Kang Peng, and H Eugene Stanley, "Physiobank, physiotoolkit, and physionet," *Circulation*, vol. 101, no. 23, pp. e215–e220, 2000.

[188] Qiegen Liu, Shanshan Wang, Jianhua Luo, Yuemin Zhu, and Meng Ye, "An augmented lagrangian approach to general dictionary learning for image denoising," *Journal of Visual Communication and Image Representation*, vol. 23, no. 5, pp. 753–766, 2012.

[189] Yao-Liang Yu, "On decomposing the proximal map," in *Advances in Neural Information Processing Systems (NIPS)*, 2013, pp. 91–99.

[190] Hiroaki Sakoe and Seibi Chiba, "Dynamic programming algorithm optimization for spoken word recognition," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 26, no. 1, pp. 43–49, 1978.

[191] James Bergstra and Yoshua Bengio, "Random search for hyper-parameter optimization," *Journal of Machine Learning Research*, vol. 13, no. Feb, pp. 281–305, 2012.

[192] Paolo Gentile, Marco Pessione, Antonio Suppa, Alessandro Zampogna, and Fernanda Irrera, "Embedded wearable integrating real-time processing of electromyography signals," in *Proceedings of Eurosensor*, 2017.

[193] ST, "`http://www.st.com/en/evaluation-tools/nucleo-l476rg.html`," 2016.

[194] Neal Parikh, Stephen Boyd, et al., "Proximal algorithms," *Foundations and Trends® in Optimization*, vol. 1, no. 3, pp. 127–239, 2014.

[195] Matthew D Zeiler, Dilip Krishnan, Graham W Taylor, and Robert Fergus, "Deconvolutional networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010, pp. 2528–2535.

[196] Matthieu Kowalski, "Thresholding rules and iterative shrinkage/thresholding algorithm: A convergence study," in *Proceedings of the IEEE International Conference on Image Processing (ICIP)*, 2014, pp. 4151–4155.

[197] Ivan W Selesnick and Mario AT Figueiredo, "Signal restoration with overcomplete wavelet transforms: Comparison of analysis and synthesis priors," in *Proceedings of the SPIE Optical Engineering + Applications*, 2009.

[198] David L Donoho and Iain M Johnstone, "Adapting to unknown smoothness via wavelet shrinkage," *Journal of the American Statistical Association*, vol. 90, no. 432, pp. 1200–1224, 1995.

[199] Brendt Wohlberg, "Convolutional sparse representation of color images," in *Proceedings of the IEEE Southwest Symposium on Image Analysis and Interpretation (SSIAI)*, 2016, pp. 57–60.

[200] Brendt Wohlberg, "Efficient convolutional sparse coding," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2014, pp. 7173–7177.

[201] Onur G Guleryuz, "Weighted averaging for denoising with overcomplete dictionaries," *IEEE Transactions on Image Processing*, vol. 16, no. 12, pp. 3020–3034, 2007.

[202] Lukas Ruff, Nico Goernitz, Lucas Deecke, Shoaib Ahmed Siddiqui, Robert Vandermeulen, Alexander Binder, Emmanuel Müller, and Marius Kloft, "Deep one-class classification," in *Proceedings of the International Conference on Machine Learning (ICML)*, 2018, pp. 4390–4399.