Anomaly Detection on Financial Time Series by Principal Component Analysis and Neural Networks

Stéphane Crépey, Lehdili Noureddine, Nisrine Madhar, Maud Thomas

Anomaly Detection on Financial Time Series by Principal Component Analysis and Neural Networks*

S. Crépey†, N. Lehdili‡, N. Madhar§, M. Thomas¶

September 14, 2022

Abstract

We consider time series representing a wide variety of risk factors in the context of financial risk management. A major issue of these data is the presence of anomalies that induce a miscalibration of the models used to quantify and manage risk, whence potentially erroneous risk measures on their basis. Therefore, the detection of anomalies is of utmost importance in financial risk management. We propose an approach that aims at improving anomaly detection on financial time series, overcoming most of the inherent difficulties. One first concern is to extract from the time series valuable features that ease the anomaly detection task. This step is ensured through a compression and reconstruction of the data with the application of principal component analysis. We define an anomaly score using a feed-forward neural network. A time series is deemed contaminated when its anomaly score exceeds a given cut-off. This cutoff value is not a hand-set parameter, instead it is calibrated as a parameter of the neural network throughout the minimisation of a customized loss function. The efficiency of the proposed model with respect to several well-known anomaly detection algorithms is numerically demonstrated. We show on a practical case of value-at-risk estimation, that the estimation errors are reduced when the proposed anomaly detection model is used, together with a naive imputation approach to correct the anomaly.

Key words: anomaly detection, financial time series, principal component analysis, neural network, density estimation, missing data, market risk, value at risk

1 Introduction

In the context of financial risk management, financial risk models are of utmost importance in order to quantify and manage financial risk. Their outputs, risk measurements, can either help in the process of decision making or ensure that the regulatory requirements are met [Basel Committee on Banking Supervision, 2013]. Financial management thus heavily relies on financial risk models and the interpretation of their outputs. The data usually consist of time series representing a wide variety of risk factors. A major issue of such data is the presence of anomalies. A time series is considered to be abnormal whenever its behaviour is significantly different from the behaviour of the rest of the time series [Hawkins, 1980]. In this work, we focus on the detection of abnormal observations in a risk factor time series used to calibrate financial risk models. Indeed, financial risk models may be sensitive to anomalies, when erroneous

*Python notebooks reproducing the results of this paper are available on `https://github.com/MadharNisrine/PCANN`. The authors would like to thank Pascal Oswald, Leader Market & Counterparty Risks Modelling, from Natixis, for insightful discussions.

†`stephane.crepey@lpsm.paris`. LPSM, Université Paris Cité, France.

‡`noureddine.lehdili@natixis.com`. Expert Leader Market & Counterparty Risks Modelling

§Email: `nisrine.madhar@lpsm.paris`. PhD student, Université Paris Cité, France. The research of N. Madhar is funded by a CIFRE grant from Natixis.

¶`maud.thomas@sorbonne-universite.fr`. Sorbonne Université, CNRS, Laboratoire de Probabilités, Statistique et Modélisation, LPSM, 4 place Jussieu, 75005 Paris, France

input data wrongly impact the calibrated model parameters. A typical example could be the estimation of the covariance matrix of a bank risk factors. The covariance matrix is involved, for instance, in the computation of Value-at Risks (VaR) or Expected Shortfalls, that is expected losses above the VaR level. Since the true covariance is unknown, it has to be estimated from the data. However, the presence of anomalies in the data might have an impact on the estimation. For this specific case, robust methods, which are less sensitive to anomalies, can be used. Yet, existing robust estimators are computationally expensive, with a polynomial or even exponential time complexity in terms of the number of risk factors. A faster approach was suggested by [Cheng et al., 2019]. Nonetheless, this algorithm only applies to the estimation of the covariance matrix in the case of a high-dimensional Gaussian distribution. Financial risk models used by banks are widespread and various. Therefore, instead of seeking for a robust version for each of them, we propose to detect anomalies directly on the time series. This way we are able to bypass the problem of sensitivity of all the models to anomalies.

Baseline Algorithms Anomaly detection aims at finding an *"observation that deviates so much from other observations as to arouse suspicion that it was generated by a different mechanism"* [Hawkins, 1980]. The baseline anomaly detection algorithms, described in [Chandola, 2009], struggle to identify anomalies in time series, mainly because their assumptions are invalidated. Indeed, first, if we consider models built for spatial data, a major assumption of these models is that observations are independent, whereas for time series, high dependency exists between different time stamps. Clustering-based approaches, like the density based spatial clustering with noise (DBSCAN) method, are particularly impacted by this aspect: if an anomaly occurs at a given time stamp and is followed by incorrect values, clustering-based approaches consider that the observations of the time series belong to two different clusters and thus fail to identify the anomaly. Another limitation when considering this type of techniques is the choice of the similarity metric used for data clustering. This task, although being a crucial pillar of these approaches, is not trivial and becomes even more complex for high dimensional problems.

Statistical Approach to Anomaly Detection The statistical techniques for anomaly detection can be split in two families: statistical tests and predictive models. Both suffer from the curse of dimensionality and model/data mismatch. When anomaly detection relies on hypothesis tests, usually they test whether the observations are drawn from a known distribution [Zhang and Paschalidis, 2017], which supposes that the user knows the probability distribution of the normal observations. This parametric framework narrows down the scope of applicability of hypothesis tests, as the data does not always coincide with the assumed distribution. Moreover, the tests provided in the literature are not suitable in high dimensional settings since they were originally built for univariate data as explained in [Kurt et al., 2020]. The statistical technique relying on fitting a predictive model to each time series also requires strong assumptions on the data. Predictive models are usually autoregressive (AR), moving average (MA), or ARMA models. Anomalies are then detected relatively to the forecasts suggested by the model [Chandola, 2009]. However, in this parametric approach, some parameters have to be specified, starting with the order of the models. Selecting the optimal model parameters with respect to an information criterion is not always possible [Sedman, 2018]. Additionally, these models assume that the time series are homogeneous, i.e. drawn from the same distribution [Laptev et al., 2015]. This is not always satisfied in the financial risk management case where several types of risk factors are treated simultaneously.

Score based Anomaly Detection Models Additional anomaly detection challenges are of general concern. Most of anomaly detection algorithms are score-based, in the sense that these approaches return an anomaly score reflecting to which extent the observation is considered to be abnormal by the model. In order to decide whether an observation is abnormal or not, a

cut-off value of the score has to be selected. Empirical approaches are often used, consisting in either setting the cut-off value as a quantile of the distribution of the anomaly score, or in considering the cut-off value as the elbow point of this distribution. However, the selected cut-off value according to such methods remains arbitrary. An advanced approach is proposed by [Gao and Tan, 2006] who propose to rely on the cost of misclassification. In most cases, the latter is taken as a weighted classification accuracy. Approaches to calculate "optimal" weights are described in [Lu et al., 2019], but they involve a grid search, technique that may draw near the optimum but may also diverge. Another alternative is to determine the cut-off value by cross validation on the training data as in [Saha et al., 2009]. Finally, some methods do not select any cut-off value, but are based instead on a contamination rate. However, in a sense, fixing a cut-off value or deciding on a contamination rate is equivalent.

Scarcity of Anomalies and Data Augmentation The scarcity of anomalies within the data sets is another typical problem in anomaly detection. Anomalies are, by definition, rare events, therefore they are underrepresented in the data set used to fit the models. This under representation is not helping in the design of a reliable model able to identify anomalies. Classical methods to overcome this issue consider data augmentation. These techniques aims at producing new synthetic samples that will ultimately enhance model performance since a better representation of the feature space is allowed. As reported in [Wen et al., 2020], time series can be augmented by using a simple transformation (in time domain [Cui et al., 2016] or frequency domain [Gao et al., 2020]) or more advanced approaches as generative models, which involve deep learning techniques (such as recurrent generative adversarial networks [Esteban et al., 2017]). In practice, the use of generative models for data augmentation of time series with anomalies presents two limitations. First, training such models requires a large number of samples to guarantee a satisfactory performance. While restricted Boltzman machines do not exhibit this problem, [Kondratyev et al., 2020] have shown that they fail into fitting multivariate complex distributions with non linear dependence structure. A more fundamental limitation affects the very idea behind generative models. Such networks are trained to learn a given distribution. However, by definition, anomalies are different from each other and therefore there is not a distribution that characterises them.

Supervised vs. Unsupervised learning Since anomalies are the realisations of atypical events for which the distribution is unknown, it seems quite natural to use unsupervised algorithms. However, these approaches are deemed more suitable for learning complex patterns and are task specific. Moreover, [Görnitz et al., 2013] pointed out that they often do not present a high prediction performance, in particular in high dimensional settings [Ruff et al., 2019]. Indeed, the performance of unsupervised shallow anomaly detection algorithms depend upon a feature engineering step. [Akyildirim et al., 2022] proposed to use signatures as feature extractors which are fed to algorithms such as isolation forest. This combination of techniques is shown to over-perform benchmark approaches. However, the designed model is task specific (detection of pump and dumps attacks) and the feature extraction step is only efficient when at least one one explanatory variable is considered in the analysis. Moreover, the unavailability of labelled data makes the model building and its evaluation even more complex. As for the supervised methods the only limit on which the literature tends to agree is their incapacity to generalize the learned patterns to new samples, which is the consequence of misrepresentation of anomalies among the training samples [Zhao and Hryniewicki, 2018]. However, even a small fraction of labelled data represents an essential information, that should be integrated during the model elaboration. By contrast, under the unsupervised learning framework, the use of this scarce but precious information is not possible [Ruff et al., 2019]. The lack of labelled data can be sidestepped through the use of data augmentation techniques on the fraction of available labelled data. Hence, for these reasons the supervised learning framework is to be preferred even when

only a small set of labelled data is available.

Anomaly detection on Time Series Usually, anomaly detection models on time series have two main components. The first component aims at extracting a parsimonious yet expressive representation of the time series. Several approaches are suggested in the literature to deal with such feature extraction. Recently, deep neural networks have been shown to suffer from overparametrization and to be often computationally expensive [Dempster et al., 2020, Akyildirim et al., 2022]. Path signatures are also computationally demanding. This could perhaps be alleviated by the random signatures [Compagnoni et al., 2022]. However, the information extracted with signatures is of most interest when the considered paths are characterized by several variables. The resulting representation is then transformed into an anomaly score. The second component aims at converting the anomaly score, which is usually a continuous variable, into a binary label [Braei and Wagner, 2020].

PCA NN Anomaly Detection In this paper, we propose a methodology to identify anomalies on time series using a principal component analysis (PCA) as underlying model, challenging current methods by providing a competitive alternative overcoming the above described pitfalls. PCA is involved in our approach as feature extractor. With a relatively low number of features given by PCA, we are able to accurately describe the dynamics of risk factors represented by times series. The reason standing behind that is the high correlation structure displayed by financial risk factors. The particular power-fullness of auto-encoder, a non-linear PCA, as data compressor is not desirable herein, since auto-encoders compress all patterns including abnormal ones. In the literature, PCA is usually used in anomaly detection for its dimension reduction properties. Anomalies are identified on the latent space, either by applying some anomaly detection algorithm or by assuming a given distribution on the principal component and identifying the anomalies relatively to a quantile as in [Shyu et al., 2006]. Once PCA is involved in anomaly detection, the user assumes that a normal subspace representation of the data set can be constructed with the first k-components [Ringberg et al., 2007]. Anomaly detection is then achieved by looking at the observations that cannot be expressed in terms of the first k-components [Bin et al., 2016]. While [Ding and Tian, 2016] claimed that the PCA-based models are stable with respect to their parameters, such as the number of principal components k spanning the subspace or the cut-off level, [Ringberg et al., 2007] found instead that PCA-based anomaly detection is sensitive to these parameters and to the amplitude of the anomalies. Indeed, the latter may undermine the construction of the normal subspace representation, in turn leading to mis-identification of anomalies. In light of that, we take an extra care regarding these aspects, and appropriate tests are conducted to show that the suggested approach is not subject to such issues. In addition, we introduce a neural network approach to calibrate the anomaly score threshold.

Outline The paper is organised as follows. In Section 2, we present the PCA NN approach with a particular focus on each step of the technique. A brief description of PCA is also provided to motivate its use as an efficient feature extractor tool. In Section 3, we present the numerical framework adopted to generate time series in Section 3.1, then contaminate them (Section 3.2) and the process of data augmentation used to overcome the scaricity of anomalies. Our main numerical results are provided in Section 4, with the main performance metrics that were considered to assess the suggested anomaly detection model (Section 4.1). The calibration of the latent space dimension is described in Section 4.3 followed by the model evaluation in Sections 4.4 and 4.5. We numerically demonstrate the efficiency of our approach over baseline anomaly detection models in Section 4.6. In Section 4.7, we tackle the issue of anomalies imputation once detected, comparing the imputation approach suggested by our approach to comparable imputation techniques (in terms of computational complexity). The calibration of the anomaly score cut-off value is a main contribution of the approach, therefore its robustness was tested in

Section 4.8. Finally, we show on a practical case of value-at-risk estimation in Section 5, that the estimation errors are reduced when the proposed anomaly detection model is used, together with a naive imputation approach to correct the anomaly. A conclusion of the paper with further discussions are presented in Section 6.

2 Method: PCA NN Anomaly Detection

After setting the key notations that are used in the remainder of the paper we proceed with the a general description of our two step method for anomaly detection on time series while briefly recalling tenets of principle components analysis. Then the first step of the approach i.e. the contaminated time series identification is introduced whilst explaining the motivation behind the model suggested for this step. The second step i.e. anomaly localization step is described in further details at the end of this section.

2.1 Notations

In this section, we introduce the notations that we use in the following sections to describe our two-step PCA methodology for anomaly detection on financial time series. Let

$$\boldsymbol{X} = \left(X^1, X^2, \ldots, X^i, \ldots, X^n\right)^\top, \tag{1}$$

where the column-vector $X^i = \left(x^i_{t_1}, x^i_{t_2}, \ldots, x^i_{t_j} \ldots, x^i_{t_p}\right) \in \mathbb{R}^p$ corresponds to the i-th observed time series, and $x^i_{t_j}$ to the value observed at time t_j in the i-th time series.

Our method fits into the supervised framework. We thus assume that the data matrix $\boldsymbol{X}$ comes along with two label matrices. The first label matrix $\boldsymbol{A} = \left(A^1, \ldots, A^n\right) \in \{0,1\}^n$ identifies the time series containing anomalies referred to as contaminated time series. For $i = 1, \ldots, n$, we define the identification labels as

$$A^i = \begin{cases} 1 & \text{if there exists } j \text{ such that } x^i_{t_j} \text{ is an anomaly.} \\ 0 & \text{otherwise.} \end{cases} \tag{2}$$

The second label matrix $\boldsymbol{L}$ concerns solely the contaminated time series. Its coefficients correspond to the location labels, that is the time stamps at which an anomaly occurs. For $j = 1, \ldots, p$ and $i \in I_c = \{i : A^i = 1\}$ (the set of contaminated time series),

$$L^i = \jmath \quad \text{when the anomaly occurs at time } t_\jmath \text{ for the } i\text{-th contaminated time series,}$$

$\boldsymbol{X}, \boldsymbol{A}$ and $\boldsymbol{L}$ are general notations. As the model involves a learning phase, we denote by $\boldsymbol{X}^{Train}$, $\boldsymbol{A}^{Train}$ and $\boldsymbol{L}^{Train}$ the data used for calibration and by $\boldsymbol{X}^{Test}$, $\boldsymbol{A}^{Test}$ and $\boldsymbol{L}^{Test}$ the independent data set on which the model performance is evaluated.

2.2 Model Description

The anomaly detection model we propose is a two step supervised-learning approach. The first step, namely *Contaminated Time Series Identification Step*, aims at identifying among the observed time series the ones with potential anomalies. The second step, called *Anomaly Localization Step*, consists in finding the location of the anomaly in each contaminated time series identified during the first step.

The model used in the first step falls under the scope of binary classification models, as it will assign to each time series a predicted label $\widehat{A^i}$ in the following way:

$$\widehat{A^i} = \begin{cases} 1 & \text{if } X^i \text{ is considered as contaminated by the identification model.} \\ 0 & \text{otherwise.} \end{cases}$$

The second step uses a multi-classification model. For each contaminated time series identified in the first step, the model predicts a unique time stamp $\widehat{L}^i$ at which the anomaly has occurred. Formally, the observed value at the time stamp $t_{\hat{\jmath}^i}$ of the i-th contaminated time series is considered abnormal, meaning that the observation $x^i_{t_{\hat{\jmath}^i}}$ is abnormal. Thus,

$$\widehat{L}^i = \hat{\jmath}^i \quad \text{with} \quad \hat{\jmath}^i \in \{1, \ldots, p\}.$$

Our two-step anomaly detection model locates only one anomaly per run, if any. Hence, several iterations allow removing all anomalies. Indeed, as long as the time series is identified as contaminated by the first step of the model, the time series can go through the second step of the model. Once, all anomalies have been located the final step of the approach would be to remove the anomalous values and suggest imputation values.

2.3 Theoretical Basics of Principal Component Analysis

In this section, we recall the theoretical tenets of the principal component analysis (PCA) useful for our purpose. The idea of PCA is to project the data from the observation space of dimension p into a latent space of dimension k, with $k < p$. This latent space is generated by the k directions for which the variance retained under projection is maximal.

The first principal component U is given by the equation

$$U = w^\top \boldsymbol{X} \quad \text{with} \quad \text{var}(U) = w^\top \Sigma w,$$

with w a vector of weights to be determined. Since the PCA aims at maximizing the variance along its principal components, w is sought for as

$$\arg \max_{\|w\|=1} w^\top \Sigma w \,.$$

The optimal solution by the Lagrangian method is given by

$$\Sigma w = \lambda w \,,$$

where $\lambda > 0$ is a Lagrangian multiplier. Multiplying both sides by $w^\top$ results in

$$w^\top \Sigma w = \lambda.$$

Following the same process with an additional constraint regarding the orthogonality between the principal components, one can show that the k first principal components correspond to the eigenvectors associated with the k dominant eigenvalues of Σ.

The transfer matrix $\Omega_{\boldsymbol{X}} \in \mathbb{R}^{k \times p}$ is defined by the eigenvectors of Σ associated with its k largest eigenvalues. This transfer matrix $\Omega_{\boldsymbol{X}}$ applied to any observation in the original observation space projects it into the latent space. Since the transformation is linear, reconstructing the initial observations from their equivalent in the latent space is straightforward. Using PCA on the data matrix $\boldsymbol{X}$, the transfer matrix $\Omega_{\boldsymbol{X}}$ can be inferred. The projection of the observations into the latent space $\boldsymbol{Z} \in \mathbb{R}^{n \times k}$ is then given by $\boldsymbol{Z} = \boldsymbol{X}\Omega_{\boldsymbol{X}}^\top$, and their reconstructed values by $\widehat{\boldsymbol{X}} = \boldsymbol{Z}\Omega_{\boldsymbol{X}}$.

The reconstruction errors ε^i for each time series defined by

$$\varepsilon^i = \widehat{X}^i - X^i, \quad i = 1, \ldots, n, \tag{3}$$

are our new representation of data, that are given as inputs to our models, referred in the sequel as the new features. It is worth stressing that the two steps of the model use the same inputs, namely these reconstruction errors. Nevertheless, the processing these inputs undergo in each step differ.

A fundamental intuition behind the anomaly detection algorithm we suggest is the existence of a lower dimensional subspace where normal and abnormal observations are easily distinguishable. Indeed, since the selected principal components explain a given level of the variance of the data, they represent the common and essential characteristics to all observations. As one might reasonably expect, these characteristics mostly represent the normal observations. Therefore, when the inverse transformation is applied, the model successfully reconstructs the normal observations with the help of the extracted characteristics, while it fails into reconstructing the anomalies.

Following this intuition, we expect normal observations to have low reconstruction errors, and higher errors for anomalies. Since this error reflects the degree to which the instance is considered as abnormal, it can be used to propose an anomaly score according to which each observation will be labelled as either normal or abnormal. A decision rule thus needs to be defined on the anomaly score to convert it into a binary label: classically in anomaly detection, a time series is labelled as abnormal if its anomaly score exceeds a previously determined cut-off value s. This cut-off value is, in most of anomaly detection algorithms, a hyper-parameter that needs to be tuned. In this paper, we propose to calibrate s during the learning phase as described in Section 2.4. Algorithm 1 summarizes how to compute the reconstruction errors.

Algorithm 1 Features extraction with PCA

inputs : number of principal components k
out of sample data $\boldsymbol{X}^{Test}$
training data if not trained yet $\boldsymbol{X}^{Train}$

if *not trained yet* **then**
  $\Omega_{\boldsymbol{X}^{Train}} \leftarrow \text{PCA}(\boldsymbol{X}^{Train}, k)$
end
$\boldsymbol{\varepsilon}^{Train} = \boldsymbol{X}^{Train}\left(\Omega^{\top}_{\boldsymbol{X}^{Train}}\Omega_{\boldsymbol{X}^{Train}} - I_p\right)$
$\boldsymbol{\varepsilon}^{Test} = \boldsymbol{X}^{Test}\left(\Omega^{\top}_{\boldsymbol{X}^{Train}}\Omega_{\boldsymbol{X}^{Train}} - I_p\right)$

outputs : $\boldsymbol{\varepsilon}^{Train}$ Reconstruction errors for the train set
$\boldsymbol{\varepsilon}^{Test}$ Reconstruction errors for the test set

In the next sections, we introduce the two steps of our approach, while the first step identifies the contaminated time series, the second step focuses on the localization of the anomaly in the time series. Both steps use the reconstruction errors as inputs. The main difference is the processing undergone by these inputs, while for the first step we use a neural network to compute an anomaly score which reflects the propensity of the time series of being contaminated (cf. Section 2.4), a more shallow approach is applied on the reconstruction errors to localise the abnormal observation in the second step (cf. Section 2.5).

2.4 Contaminated Time Series Identification

Recall that the first step of the approach, that is the *Contaminated time series identification* step, aims at identifying the time series with anomalies based on the reconstruction errors that we obtain using PCA and Algorithm 1. In this section, we detail the process these new features undergo. In fact, we assign to each reconstruction error an anomaly score which computation is handled by a neural network (NN). We also numerically demonstrate that our NN have an advantage over the naive approach in terms of being able to assess more accurately the propensity of a time series of being contaminated.

Naive approach A naive and natural approach to define the anomaly score is to consider the ℓ^2-norm of the reconstruction errors $\widetilde{\varepsilon}_i = \|\varepsilon^i\|_2$, $i = 1, \ldots, n$. This anomaly score represents the propensity of a time series to be contaminated or not. The scores are first split into two sets

depending on whether the corresponding time series is identified as contaminated or not. We denote by $\widetilde{\boldsymbol{\varepsilon}}^c$ and $\widetilde{\boldsymbol{\varepsilon}}^u$ the set of contaminated and uncontaminated time series, i.e.

$$\widetilde{\boldsymbol{\varepsilon}}^c := \{\|\varepsilon^i\|_2;\ A^i = 1\} = \{\widetilde{\varepsilon}_i,\ i \in I_c\}$$
$$\widetilde{\boldsymbol{\varepsilon}}^u := \{\|\varepsilon^i\|_2;\ A^i = 0\} = \{\widetilde{\varepsilon}_i,\ i \notin I_c\}.$$

The density distribution function of each class of time series f^u and f^c, i.e. for both $\widetilde{\boldsymbol{\varepsilon}}^c$ and $\widetilde{\boldsymbol{\varepsilon}}^u$, is then estimated using kernel density estimation [Węglarczyk, 2018]. For $l = \{c, u\}$, let $\boldsymbol{\varepsilon}^l$ be an i.i.d sample of n^l observations from a population with unknown density f^l. The corresponding kernel estimator is given by

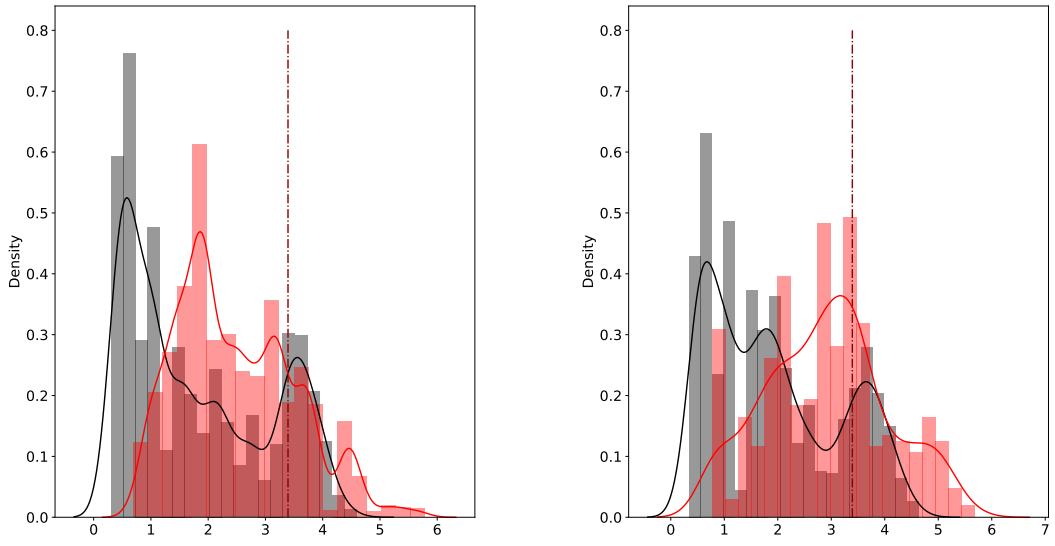
$$\hat{f}^l(s) = \frac{1}{n^l \mathfrak{h}} \sum_{k=1}^{n^l} \mathcal{K}\left(\frac{s - \widetilde{\varepsilon}_k^l}{\mathfrak{h}}\right), \tag{4}$$

where $\mathcal{K}$ is a kernel function and $\mathfrak{h}$ is a smoothing parameter.

The cut-off value s is then chosen as the intersection between the two empirical density functions, i.e.

$$\hat{s} = \arg\min_s \left\{ |\hat{f}^u(s) - \hat{f}^c(s)| < \eta \right\},$$

with η a precision level to be tuned. The selected cut-off value $\hat{s}$ represents the value of the score for which the area under the curve of the density of uncontaminated time series above $\hat{s}$ and the area under the curve of contaminated time series below $\hat{s}$ are as small as possible. We expect it to correspond to a value exceeded by a relative low amount of $\widetilde{\varepsilon}_i$ with $i \notin I_c$ as $\hat{f}^u(s)$ is expected to be flat around the cut-off value exceedance region, and exceeded only by few $\widetilde{\varepsilon}_i$ with $i \in I_c$.



a) b)

Figure 1: Empirical densities of anomaly scores given by the naive approach for uncontaminated time series in black and contaminated time series in red, a) on the train set and b) on a test set. The dotted dark red line represents the cut-off value.

As shown in Figure 1, the naive approach results in a non-negligible overlapping region between the two densities, both on the train set (see Figure 1 a)) and for the test set (see Figure 1 b)). This region represents the anomaly score associated with time series that could be either contaminated or uncontaminated. The uncertainty regarding the nature of the observation when its anomaly score belongs to this region is relatively high in comparison with observations whose anomaly scores lie on the extreme left-hand or right-hand side of the calibrated cut-off value. Moreover, because we are not able to provide a clear separation between the scores of uncontaminated and contaminated time series, we may expect the model to mislabel future observations, resulting a high rate of false positives and true negatives.

Neural network approach In view of getting a clearer separation of the densities, we propose an alternative approach for the computation of the anomaly scores built upon the reconstructions errors. Let F denote the function that associates with each reconstruction error ε^i its anomaly score. In the naive approach, F corresponds to the ℓ^2-norm. Hereafter, F instead represents the outputs of a feed-forward Neural Network (NN) [see e.g. Goodfellow et al., 2016], whose architecture is depicted in Figure 3. The training of the corresponding NN aims at minimising a loss function that reflects our ambition to construct a function F that gives accurate anomaly scores. The network used for computing the anomaly scores is defined by

$$F(\varepsilon) = \left(h^H \circ h^{H-1} \circ \dots h^2 \circ h^1\right)(\varepsilon),$$

where $h^i(\varepsilon) = \left(W^i \cdot \varepsilon + b^i\right)^+$ represents the computation carried in the i-th layer of the neural network with n_h^i hidden units. H stands for the number of layers of the NN with ReLU activation applied element-wise. We finally estimate the labels from the outputs following

$$\hat{\boldsymbol{A}} = \mathbb{1}_{\{(F(\boldsymbol{\varepsilon})-s)^+>0\}}. \tag{5}$$

The idea of the learning is to calibrate the weights $\boldsymbol{W} := \left\{W^i \in \mathbb{R}^{n_h^i}, i = 1, \dots, H\right\}$ and the biases $\boldsymbol{b} := \left\{b^i \in \mathbb{R}, i = 1, \dots, H\right\}$ such that the NN is able to accurately assess to which extent a time series is contaminated, with a clear distinction between the anomaly scores assigned to uncontaminated time series and those assigned to the contaminated ones, while integrating the calibration of the cut-off value s as part of the learning. We denote by $\Theta := \{\boldsymbol{W}, \boldsymbol{b}, s\}$ the set of parameters to be calibrated through the NN training. To meet these needs, the loss we minimise during the learning is given by

$$\mathcal{L}_\Theta\left(\boldsymbol{A}, \hat{\boldsymbol{A}}\right) = \text{BCE}(\boldsymbol{A}, \hat{\boldsymbol{A}}) + \text{AUCDensity}^u_{\boldsymbol{A},F(\boldsymbol{\varepsilon})} + \text{AUCDensity}^c_{\boldsymbol{A},F(\boldsymbol{\varepsilon})} \tag{6}$$

where

$$BCE(\boldsymbol{A}, \hat{\boldsymbol{A}}) = -\frac{1}{n}\sum_i \left(A^i \log(\hat{A}^i) + (1 - A^i)\log(1 - \hat{A}^i)\right)$$

is the binary cross-entropy, a well-known loss function classically used for classification problems. To this first component of our loss function, we add two components that aim at downsizing the overlapping region between the density of anomaly score of both types of observations. In order to have a control on the latter, we consider AUCDensity^u and AUCDentsity^c, which correspond to the area under the curve of the probability density function of anomaly scores the model assigns to contaminated and uncontaminated observations, i.e

$$\text{AUCDensity}^u(s) = \int_s^\infty \hat{f}^u_{\boldsymbol{A},F(\boldsymbol{\varepsilon})}(\omega)d\omega \qquad \text{AUCDensity}^c(s) = \int_{-\infty}^s \hat{f}^c_{\boldsymbol{A},F(\boldsymbol{\varepsilon})}(\omega)d\omega\,. \tag{7}$$

Note that the bounds of these integrals depend on the cut-off value s and define the region for which we want the probability density function to be as small as possible, which allows us to

estimate $\hat{s}$. The estimated probability density functions $\hat{f}^u_{A,F(\boldsymbol{\varepsilon})}$ and $\hat{f}^c_{A,F(\boldsymbol{\varepsilon})}$ depend on $F(\boldsymbol{\varepsilon})$, the scores assigned by contaminated time series identification model, and of the identification labels $\boldsymbol{A}$. We describe with Algorithm 2, the scoring and cut-off value calibration that we achieve throughout the calibration of a feed-forward network with the customized loss (6). To this end, the update of Θ, AdamStep in Algorithm 2, is carried following the Adam optimization algorithm of [Kingma and Ba, 2014].

Algorithm 2 Scoring and cut-off calibration

inputs : learning rate lr
number of maximum iterations K
kernel density estimator parameter $\mathcal{K}$,$\mathfrak{h}$
training data $\boldsymbol{\varepsilon}^{Train}$, $\boldsymbol{A}^{Train}$

Initialize parameter Θ, $\widehat{\mathcal{L}} = \infty$ and count index $k = 0$

while $k < K$ **do**

$scores \leftarrow F_\Theta\left(\boldsymbol{\varepsilon}^{Train}\right)$

$scores^c \leftarrow scores \mathbb{1}_{\boldsymbol{A}^{Train}=1}$

$scores^u \leftarrow scores \mathbb{1}_{\boldsymbol{A}^{Train}=0}$ `// Density estimation following` (4)

$\hat{f}^u_{\boldsymbol{A}^{Train},scores^u} = \text{KernelDensityEstimator}(\mathcal{K}, \mathfrak{h}, scores^u)$

$\hat{f}^c_{\boldsymbol{A}^{Train},scores^c} = \text{KernelDensityEstimator}(\mathcal{K}, \mathfrak{h}, scores^c)$

`// Loss evaluation following` (6) `and` (7)

$\text{AUCDensity}^u \leftarrow \text{NumericalIntegration}(\hat{f}^u_{\boldsymbol{A}^{Train},scores^u}, s)$

$\text{AUCDensity}^c \leftarrow \text{NumericalIntegration}(\hat{f}^c_{\boldsymbol{A}^{Train},scores^c}, s)$

$\hat{\boldsymbol{A}} \leftarrow \mathbb{1}_{scores>s}$

$\mathcal{L} \leftarrow \text{BCE}\left(\boldsymbol{A}^{Train}, \hat{\boldsymbol{A}}\right) + \text{AUCDensity}^u + \text{AUCDensity}^c$

if $\widehat{\mathcal{L}} > \mathcal{L}$ **then**

$\widehat{\mathcal{L}} \leftarrow \mathcal{L}$

$\widehat{\Theta} \leftarrow \Theta$

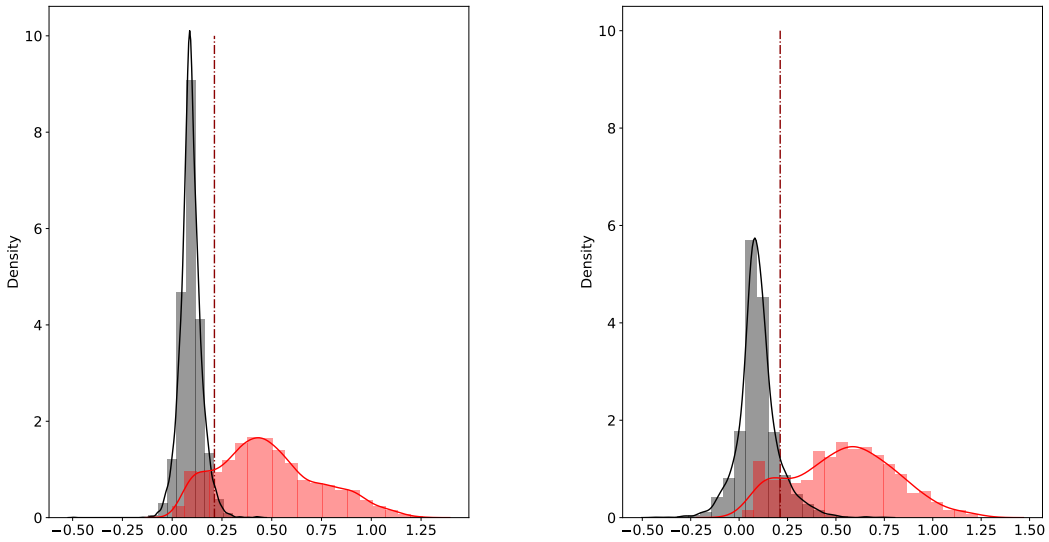
end

$\Theta \leftarrow \text{AdamStep}(\mathcal{L}, \Theta, lr)$

$k \leftarrow k + 1$

end

outputs : best calibrated parameter $\widehat{\Theta} = \{\widehat{\boldsymbol{W}}, \hat{\boldsymbol{b}}, \hat{s}\}$.



a) b)

Figure 2: Empirical densities of anomaly scores given by the NN approach for uncontaminated time series in black and contaminated time series in red, on the train set a) and on a test set b). The dotted dark red line represents the calibrated cut-off value $\hat{s}$.

Approach	AUCDensity^u	AUCDensity^c
Naive	0.1725	0.7898
NN	0.05153	0.1550

a)

Approach	AUCDensity^u	AUCDensity^c
Naive	0.1897	0.6354
NN	0.1290	0.1367

b)

Table 1: AUC obtained with the naive and the NN approaches for a) the train set and b) the test set.

Table 1 shows that with the NN approach the densities of scores assigned to each type of time series display the expected behaviours on the left-hand (right-hand) side of the cut-off value for contaminated (uncontaminated) time series, on the train and test sets. Actually, the lower AUCDensity^u (AUCDensity^c) is, the lower the number of uncontaminated (contaminated) time series to which are assigned anomaly scores above (below) the cut-off value, which prevents mislabelling.

Once the features $\boldsymbol{X}$ and the calibrated NN are provided, the contaminated times series identification model is ready for use. This step is described by Algorithm 3.

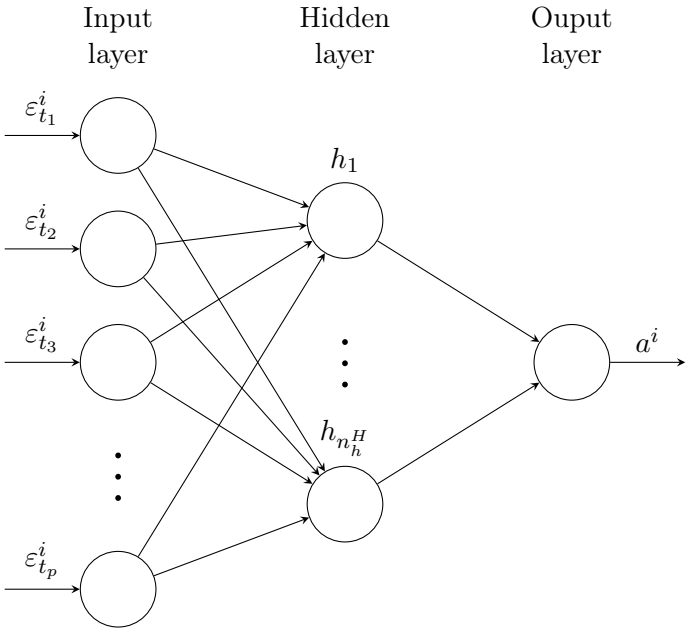


Figure 3: Architecture of the network computing the anomaly score a^i assuming $H = 1$, with the reconstruction error $\boldsymbol{\varepsilon}^i$ as input.

Algorithm 3 Contaminated time series identification model

inputs : Time series to analyze X^i,
Calibrated model parameters $\widehat{\Theta}$,
$\varepsilon^i \leftarrow \text{PCAFeaturesExtraction}(X^i)$; `// cf. Algorithm 1 in Section 2.3`
$score^i \leftarrow F_{\widehat{\Theta}}(\varepsilon^i)$
$\hat{A}^i \leftarrow \mathbb{1}_{score^i > \hat{s}}$
outputs : identification label $\hat{A}^i$.

2.5 Anomaly localization Step

Once the time series containing an anomaly have been identified, the second step of our approach aims at locating an abnormal observation (see Section 2.2) among each contaminated time series. Again we use the reconstruction errors defined in (3) as inputs of this second step. The difference lies on the transformation these reconstruction errors undergo before labelling the different time stamp observations. Namely, we now consider the following element-wise transformation of the reconstruction errors:

$$F(\varepsilon^i) = |\varepsilon^i| = |X^i - \widehat{X}^i|.$$

The model inputs $F(\varepsilon^i) \in \mathbb{R}^p_+$ is thus assigned to each time series $X^i \in \mathbb{R}^p$. The time stamp of occurrence of the anomaly is given by

$$\widehat{L}^i := \arg\max_j \left\{ F(\varepsilon^i_{t_j}) : j \in \{1, \ldots, p\} \right\}.$$

Algorithm 4 recaps the anomaly localization model.

Algorithm 4 Anomaly Localization Model

inputs : time series X^i to analyze.

$\varepsilon^i \leftarrow \text{PCAFeaturesExtraction}(X^i)$

$\widehat{L}^i \leftarrow \arg\max_{\jmath} \left\{ |\varepsilon^i_{t_\jmath}| : \jmath \in \{1, \ldots, p\} \right\}$

outputs : anomaly location $\widehat{L}^i$.

Note that anomalies are not necessarily extrema. For this reason the above PCA feature extraction step is necessary. Indeed, building on the reconstruction errors, the model identifies these extrema-anomalies, but also abnormal observations which are not necessarily extrema. This subtlety of the nature of the observations underlines the importance of going further than taking the index of the highest observed value of the contaminated time series X^i.

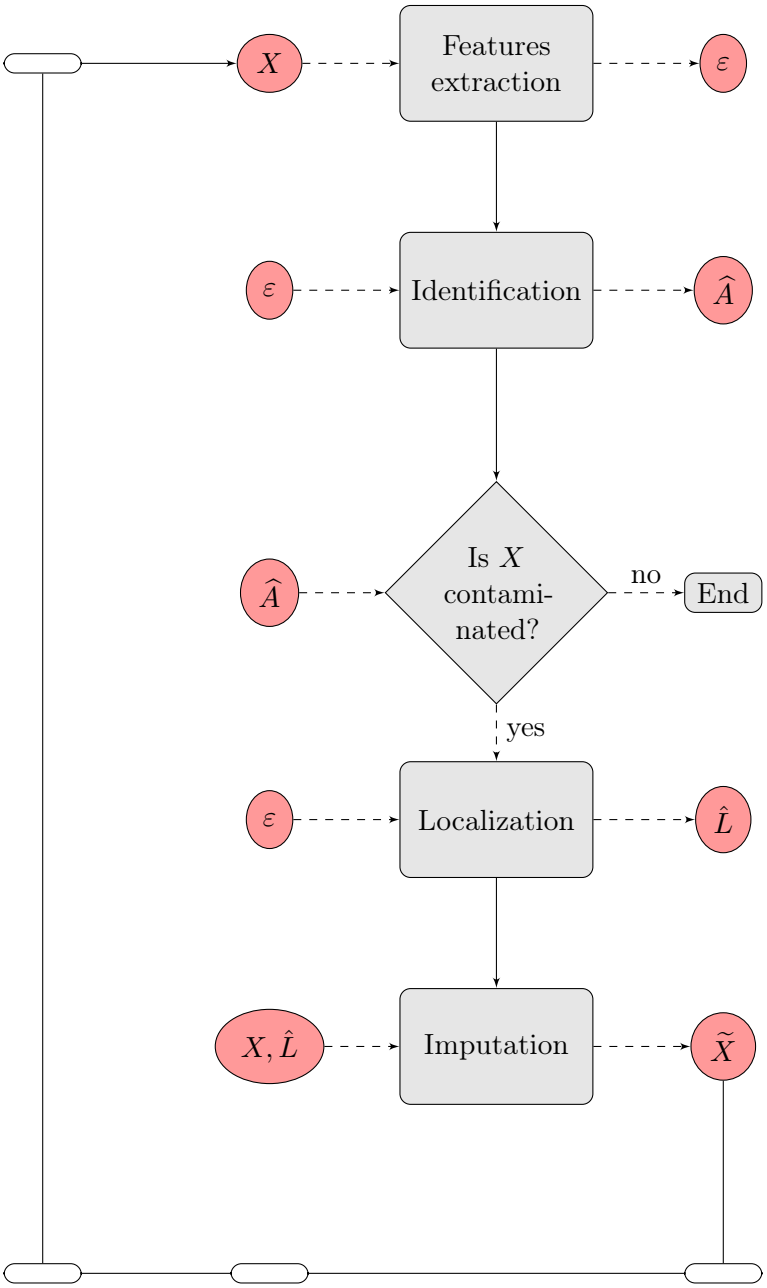


Figure 4: Flow chart of our two step anomaly detection model (PCA NN), depicting the process a time series X goes through.

The flow chart of Figure 4 along with Algorithm 5 summarize our approach. When a time series X is given to our model, we suggest a new representation of X, namely ε, through a features engineering step involving PCA. The resulting representation ε^i feeds the first component of the model, i.e. the identification step, which evaluates the time series likelihood of being contaminated. The optimal parameter $\widehat{\Theta}$ of the identification model solves

$$\widehat{\Theta} = \arg \min_{\Theta=(\boldsymbol{W},\boldsymbol{b},s)} \sum_{X \in \boldsymbol{X}} \mathcal{L}_\Theta \left(A, \hat{A}\right),$$

where $\hat{A} = \mathbb{1}_{F\left((\Omega\Omega^\top - I_p)X\right) > s}$ and $\mathcal{L}_\Theta$ is the loss function defined in (6). Then, if the model considers the time series as contaminated, the localization model takes over to localise the abnormal value in X. For this second step of the model the time stamp of occurrence of the anomaly is given by

$$\arg\max_j \quad \left\{ \left(X \left(\Omega^\top \Omega - I_p \right) \right)_j : j \in \{1, \dots, p\} \right\}$$

Finally, the anomaly is imputed. As our approach integrates the computation of a reconstruction of the time series, we could replace the anomaly with the corresponding reconstructed value. The model will be then able to detect the anomaly and suggest an imputation value. However, numerical tests described in Section 4.7 show that imputations with naive approaches perform better.

Algorithm 5 Anomaly Detection Model

inputs : time series X^i to analyze,
calibrated identification model parameters

$\widehat{A}^i \leftarrow \text{IdentificationModel}(X^i, \widehat{\Theta})$ `// Algorithm 3`

if $\widehat{A}^i = 1$ **then**

  $\widehat{L}^i \leftarrow \text{LocalizationModel}(X^i)$ `// Algorithm 4`

end

outputs : identification label $\widehat{A}^i$,
anomaly localization $\widehat{L}^i$.

3 Data

Since anomalies are rare by definition, real world data sets are very imbalanced, in the sense that the proportion of anomalies compare to normal observations is very small, making the learning phase of the model difficult. This has led us to consider synthetic data for the model calibration. Our data set is obtained through a three-step process including time series simulations, contamination and data augmentation. We point out that, in the data simulation, care was taken to ensure that the generated data sets stay realistic. Particularly, only few anomalies were added to time series as described in Section 3.2. To this extent, we still face the problem of scarcity of anomalies in this synthetic framework, and we provide some preprocessing steps to sidestep this issue as well.

3.1 Data Simulation

In this section, we describe the model used to simulate the data. Recall that our primary motivation is to detect anomalies in financial time series. For that purpose, we consider share price sample paths generated through the Black and Scholes model i.e. geometric Brownian motions. Under this framework, the share price $(S_t)_t$ is solution to the stochastic differential equation

$$dS_t = S_t \mu dt + S_t \sigma dW_t, \tag{8}$$

where W represents a standard Brownian motion, μ the drift and σ the volatility of the stock. Applying Itô's lemma to the process defined by $\log(S)$, yields

$$S_t = S_0 \exp\left(\left(\mu - \frac{1}{2}\sigma^2\right) t + \sigma W_t\right). \tag{9}$$

Let N stocks $S^1, \ldots, S^N$ simulated simultaneously from this model, with $S^i := \left\{S^j_{t_0}, S^j_{t_1}, \ldots, S^j_{t_T}\right\}$ the time series of length T representing the (time discretized) path diffusion of the j-th stock. Each stock S^i has its own drift μ^i, volatility σ^i and initial value $S^i_{t_0}$. The paths parameters are selected randomly according to

$$S^i_{t_0} \sim \mathcal{N}(100, 1), \qquad \mu^i \sim \mathcal{U}([0.01, 0.2]), \qquad \sigma^i \sim \mathcal{U}([0.01, 0.1]), \quad i = 1, \ldots, N. \tag{10}$$

For the sake of realism the Brownian motions driving the N-stocks are correlated. We assume that the time series obtained thanks to the paths diffusion do not contain any type of anomalies. The contamination of these time series by anomalies is described in the next section.

3.2 Time series Contamination

We apply a quite naive approach to introduce anomalies into our time series. We introduce the same fixed number of anomalies $n^{anomaly}$ to each time series by applying a shock on some original values of the observed time series. Formally, the $\jmath$-th added anomaly is characterized by its location $t_\jmath$ corresponding to the time stamp at which the anomaly has occurred, its shock $\delta_\jmath$, the amplitude of the shock is given by $|\delta_\jmath|$ and its sign by $\operatorname{sgn}(\delta_\jmath)$. We denote by $S^{a,i}$ the time series resulting from the contamination of the i-th clean time series S^i. For $i \in \{1, \ldots, N\}$, let $\mathcal{J}^i$ be the set of indices of the time stamps at which an anomaly occurs for the i-th time series. For $\jmath \in \mathcal{J}^i$, the abnormal values are

$$S^{a,i}_{t_\jmath} = S^i_{t_\jmath}(1 + \delta_\jmath).$$

The location, sign and amplitude of the shocks are generated randomly according to uniform distributions:

$$\mathcal{J} \sim \mathcal{U}_{N,n^{anomaly}}(\{1, \ldots, p\}), \qquad \operatorname{sign}(\delta) \sim \mathcal{U}_{N,n^{anomaly}}(\{-1, 1\}), \qquad |\delta| \sim \mathcal{U}_{N,n^{anomaly}}([0, \rho]),$$

where ρ is an upper bound on the shock amplitude.

We define the anomaly mask matrix $\mathbb{R}^{N \times T}$ by setting, for $i = 1, \ldots, 20$, and $j = 1, \ldots, T$,

$$\mathcal{T}_{i,j} = \begin{cases} 1 + \delta_j & \text{if } j \in \mathcal{J}^i. \\ 1 & \text{otherwise.} \end{cases} \tag{11}$$

Under this framework, where we incorporate the anomalies to the clean time series driven by the geometric Brownian motion, we assign labels to the time series observations according to whether the values correspond to anomalies or normal observations. We thus provide the labels $Y^i_{t_j}$ associated with each value $S^{a,i}_{t_j}$. Hence, for $i = 1, \ldots, N$,

$$Y^i_{t_j} = \begin{cases} 1 & \text{if } j \in \mathcal{J}^i. \\ 0 & \text{otherwise.} \end{cases} \tag{12}$$

Algorithm 6 recaps the time series contamination procedure, where AnomalyMask and GetLabels are names for the operators defined by (11) and (12).

Algorithm 6 Data Contamination

inputs : mplitude range ρ,
number of anomalies to add in time series $n^{anomaly}$,
sata $\boldsymbol{S}$

$\operatorname{sgn}(\delta) \leftarrow \mathcal{U}_{N,n^{anomaly}}(\{-1,1\})$
$|\delta| \leftarrow \mathcal{U}_{N,n^{anomaly}}(\{0,\rho\})$
$\mathcal{J} \leftarrow \mathcal{U}_{N,n^{anomaly}}(\{1,\ldots,p\})$
$\mathbb{R}^{N\times T} \ni \mathcal{T} \leftarrow \text{AnomalyMask}(\mathcal{J},\delta)$
$\boldsymbol{S}^a \leftarrow \boldsymbol{S} \circ \mathcal{T}$
$Y \leftarrow \text{GetLabels}(\mathcal{J})$

outputs : time series with anomalies $\boldsymbol{S}^a$,
labels associated with each value of the time series $\boldsymbol{L}$.

3.3 Data Augmentation

By definition, anomalies are rare events and thus represent only a low fraction of the data set. Yet, the suggested approach needs an important training set for an efficient learning. To overcome this issue, we apply a classical data augmentation technique called the sliding window method. This method not only extends the number of anomalies within the data set, but also allows the model to learn that anomalies could be located anywhere in the time series. The sliding window technique as described by [Le Guennec et al., 2016] consists in extracting $N_p = T - p + 1$ sub-time series of length p of the initial observed time series of length T.

Note that we have to split the data into train and test sets before augmentation to guarantee that time series considered in the training set do not share any observation with the ones we use for the model evaluation. In view of simplification, we introduce the data augmentation process, without loss of generality, for $\boldsymbol{S}$ and $\boldsymbol{Y}$, but one should keep in mind that this process has to be applied to $\boldsymbol{S^{Train}}, \boldsymbol{Y^{Train}}$ and $\boldsymbol{S^{Test}}, \boldsymbol{Y^{Test}}$ separately.

For $i = 1,\ldots,N$ and $q \in \{1,\ldots,N_p\}$, the sub-time series $S^{i,q}$ and the associated labels $Y^{i,q}$ are defined by (see Algorithm 7)

$$S^{i,q} = \left\{ S^i_{t_j}\,,\ j = q,\ldots,q+p-1 \right\}$$
$$Y^{i,q} = \left\{ Y^i_{t_j}\,,\ j = q,\ldots,q+p-1 \right\}.$$

Each sub-time series $S^{i,q+1}$ thus results from the shift forward in time of one observation of the previous sub-time series $S^{i,q}$. The final data set $\boldsymbol{X}$ and the labels $\boldsymbol{Y^s}$ are then defined as the matrices which rows correspond to the sub-time series $S^{i,q}_{q=1,\ldots,N_p;i=1,\ldots,N}$, and $Y^{i,q}_{q=1,\ldots,N_p;i=1,\ldots,N}$, respectively, i.e.

$$\boldsymbol{X} = \left(S^{1,1}, S^{1,2}, \ldots, S^{1,N_p}, S^{2,1}, \ldots, S^{i,q}, \ldots, S^{N,N_p}\right)^\top,$$

$$\boldsymbol{Y_s} = \left(Y^{1,1}, Y^{1,2}, \ldots, Y^{1,N_p}, Y^{2,1}, \ldots, Y^{i,q}, \ldots, Y^{20,N_p}\right)^\top.$$

With this data configuration, an observation refers to a time series $S^{i,q}$ obtained through the sliding window technique. Two observations may represent the same stock but on different time intervals.

Algorithm 7 Data Augmentation with sliding window technique

inputs : window size p,
data and labels $\boldsymbol{S}$,$\boldsymbol{Y}$;
Initialise empty slided time series and labels matrices $\boldsymbol{X}$, $\boldsymbol{Y_s}$;
for *i :=1 to N* **do**
  for *q :=1 to* N_p **do**
    $S^{i,q} \leftarrow \left\{ S^i_{t_j} | j \in \{q, \ldots, q+p-1\} \right\}$
    $Y^{i,q} \leftarrow \left\{ Y^i_{t_j} | j \in \{q, \ldots, q+p-1\} \right\}$
  end
  $\boldsymbol{X} \leftarrow$ Concatenate $\left(\boldsymbol{X}, S^{i,q}\right)$
  $\boldsymbol{Y_s} \leftarrow$ Concatenate $\left(\boldsymbol{Y_s}, Y^{i,q}\right)$
end
outputs : resulting slided time series and labels $\boldsymbol{X}, \boldsymbol{Y_s}$

While the fact that the observations share the same values may be argued to wrongly impact the learning process, we point out that a real benefit can be drawn from this situation. Indeed, thanks to this sliding window technique, the number of anomalies is considerably increased. This technique also allows the model to learn that anomalies could be located anywhere in the time series, reducing the dependency on the event location [Um et al., 2017].

However, once we apply the sliding window technique, we do not only extend the number of contaminated time series: the number of uncontaminated time series is also increased. But the minority class (contaminated time series) has, this time, a significant number of instances denoted by N^c. Therefore, in order to get a balanced data set for the training set, we perform an undersampling, selecting randomly N^c observations from the N^u uncontaminated time series without any anomalies (RandomSampling in Algorithm 8). It is important to point that the resulting retained number of observations $2N^c$ is more than enough to train the model. The test set, in turn is umbalanced. To sharpen the imbalanced characteristic of the test set we specify a contamination rate r_c which corresponds to the rate of contaminated time series in the data set. The construction of the test set is described in Algorithm 8.

Algorithm 8 Time series selection

Inputs : slided data and labels $\boldsymbol{X}$,$\boldsymbol{Y_s}$,
contamination rate r_c (for test set)

$\boldsymbol{X}, \boldsymbol{Y_s} \leftarrow \boldsymbol{X}\mathbb{1}_{\text{sum}(\boldsymbol{Y_s})\leq 1}, \boldsymbol{Y_s}\mathbb{1}_{\text{sum}(\boldsymbol{Y_s})\leq 1}$; `// Keep time series with at most one anomaly.`
if *Train set* **then**
  $N^c = \text{card}\left(\boldsymbol{Y_s}\mathbb{1}_{\text{sum}(\boldsymbol{Y_s})=1}\right)$
  `// Randomly select the indexes of` N^c `uncontaminated time series`
  $\text{index}^u = \text{RandomSampling}\left(\left\{i;\ \text{sum}(Y_s^i) = 0, i \in \{1, \ldots, NN_p\}\right\}, N^c\right)$
end
if *Test set* **then**
  $N^c = \text{card}\left(\boldsymbol{Y_s}\mathbb{1}_{\text{sum}(\boldsymbol{Y_s})=1}\right)$
  $N^u = \left\lceil \frac{N^c(1-r_c)}{r_c} \right\rceil$
  `// Randomly select the indexes of` N^c `uncontaminated time series`
  $\text{index}^u = \text{RandomSampling}\left(\left\{i;\ \text{sum}(Y_s^i) = 0, i \in \{1, \ldots, NN_p\}\right\}, N^u\right)$
end

$\text{index}^c = \left\{i;\ \text{sum}(Y_s^i) = 1, i \in \{1, \ldots, NN_p\}\right\}$
$\boldsymbol{X} \leftarrow \left(X^i, \text{for } i \in \text{index}^u \cup \text{index}^c\right)$
$\boldsymbol{Y_s} \leftarrow \left(Y_s^i, \text{for } i \in \text{index}^u \cup \text{index}^c\right)$

outputs : time series $\boldsymbol{X}$ with at most one anomaly,
corresponding labels for Identification task $\boldsymbol{A}$,
corresponding labels for localization task $\boldsymbol{L}$.

As mentioned in Section 2.2, the model is designed to predict the localization of only one anomaly. If there is more than one anomaly in the time series, this will wrongly impact the evaluation of the anomaly localization model. Therefore at this stage we only consider time series with at most one anomaly.

Hence before introducing the labels transformation, it is worth stressing that the following assumption is being made:

Assumption 1 *A time series $S^{i,q}$ contains at most one anomaly among all its observed values.*

We assign the identification label $A^{i,q}$ (see Section 2.1) to each $S^{i,q}$ following the rule

$$A^{i,q} = \sum_{j=q}^{q+p-1} Y_{t_j}^i = \begin{cases} 1 & \text{if there is an anomaly among the observed values of } S^{i,q} \\ 0 & \text{otherwise.} \end{cases}$$

Regarding the location labels, we recall that they only concern the time series with an anomaly, therefore $L^{i,q}$ is defined following

$$L^{i,q} = \arg\max_j Y^{i,q} = \arg\max_j \left\{Y_{t_j}^i,\ j = q, \ldots, q+p-1\right\} \tag{13}$$

With Algorithm 9, we give a rundown of the construction process of the identification and localization labels namely $\boldsymbol{A}$ and $\boldsymbol{L}$ departing from $\boldsymbol{Y_s}$.

The supervised learning framework is adopted herein, since we have at our disposal labelled data.

The resulting matrices with the observed values of the time series in $\boldsymbol{X}$ the associated identification labels in $\boldsymbol{A}$ and location labels in $\boldsymbol{L}$ constitute the data set used for our model calibration and evaluation.

Algorithm 9 Time series labelling

Inputs : slided labels $\boldsymbol{Y_s}$;
$\boldsymbol{A} \leftarrow \text{sum}(\boldsymbol{Y_s})$

$\boldsymbol{L} \leftarrow \arg\max \left\{ \boldsymbol{Y_s} \mathbb{1}_{\text{sum}(\boldsymbol{Y_s})=1} \right\}$

outputs : corresponding labels for identification task $\boldsymbol{A}$,
corresponding labels for localization task $\boldsymbol{L}$.

4 Model Evaluation

We briefly describe the process of the latent space dimension calibration. Then we evaluate the performance of the identification and the localization models on synthetic data, using appropriate performance indicators. And we demonstrate numerically the efficiency of the PCA NN over baseline anomaly detection algorithms. To take our approach a leap beyond classical anomaly detection algorithms, we evaluate the anomalies imputation suggest by the PCA NN model. Lastly, we test the robustness of the calibrated cut-off value.

4.1 Performance Metrics

Common methods to assess the performance of binary classifiers include true positive and true negative rates, and ROC (Receiver Operating Characteristics) curves, which display the true positive rate against the false positive rate. These methods, however, are uninformative when the classes are severely imbalanced. In this context, F_1-score and Precision-Recall curves (PRC) have been shown to be more informative [Brownlee, 2020, Saito and Rehmsmeier, 2015]. They are both based on the values of

$$\text{Precision}(s) = \frac{\text{true positives}}{\text{true positives} + \text{false positives}}$$

against the values of

$$\text{Recall}(s) = \frac{\text{true positives}}{\text{true positives} + \text{false negatives}}$$

where s is a cut-off probability varying between 0 and 1. Precision quantifies the number of correct positive predictions out all positive predictions made; and Recall (often also called Sensitivity) quantifies the number of correct positive predictions out of all positive predictions that could have been made. Both focus on the Positives class (the minority class, anomalies) and are unconcerned with the Negatives (the majority class, normal observations).

The F_1-score combines these two measures in a single index by taking the harmonic mean of those two values. It is derived from the F-Measure introduced in [Chinchor and Sundheim, 1993, Rijsbergen, 1979] and defined as

$$F_1 = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}. \tag{14}$$

To closer the F_1-score is to 1, the better the prediction model is.

PRC display the values of Precision and Recall as the cut-off s varies from 0 to 1. The PR curve of a skill-ful model bows towards the point with coordinates $(1, 1)$. The curve of a no-skill classifier will be a horizontal line on the plot with a y-coordinate proportional to the number of Positives in the data set. For a balanced data set this will be 0.5 Brownlee [2020].

PRC and F_1-score are complementary in our approach. The PRC is used on the anomaly scores outcomes of the models, it gives the best configuration and the best model, whereas the F_1-score is used to select the best cut-off value used in the prediction of the two classes, for each model.

4.2 Synthetic Data Set

The anomaly detection task is performed on $N = 20$ stocks simultaneously. The stock prices are diffused according to the Black and Scholes model, each stock has its own drift and volatility and the 20 stocks are correlated, as described in Section 3. Each time series represents $T = 1,500$ daily stock prices, split into two sets: $1,000$ observations, corresponding to the train set, are used to learn the model parameters, that is the PCA transfer matrix and the NN weights and biases. The last 500 observations, corresponding to the test set, are used to assess the quality of the estimated parameters when applied to unseen samples. Hence, for the application of the sliding window technique, we consider that the length of the resulting time series p is 206. Finally, we obtained $\boldsymbol{X}^{Train} \in \mathbb{R}^{12,000\times 206}$ and $\boldsymbol{X}^{Test} \in \mathbb{R}^{2,500\times 206}$, with $6,000$ contaminated time series in the train set and 400 in the test set.

Table 2 sums up the composition of each data set before and after data augmentation. Thanks to data augmentation, we are able to extend the number of time series in both train and test sets, going from 20 time series of length 1,000 with 4 anomalies each to 12,000 time series of length 206, among which 6,000 time series are contaminated for the train set for example.

	Nb of Time Series	Nb of Observed values per time series	Nb of anomalies
Train set	20	1,000	4×20
Test set	20	500	2×20

a)

	Nb of Time Series	Nb of Observed values per time series	Nb of anomalies
Train set	12,000	206	6,000
Test set	2,500	206	400

b)

Table 2: Data set composition a) before and b) after data augmentation

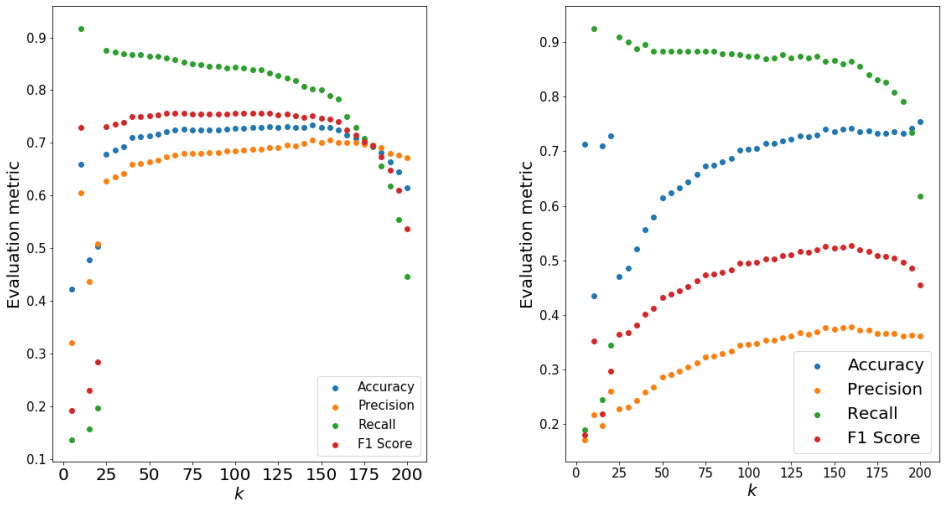
We recall that both steps of the model are preceeded by a feature extraction step, for which the latent space dimension calibration is described in Section 4.3. The features extraction guarantees the stationarity of the time series used for the anomaly detection task, namely $\boldsymbol{\varepsilon}$, as shown by the numerical results provided in Appendix B.

4.3 Calibration of the Latent Space Dimension

The dimension k of the latent space in the PCA algorithm needs to be specified. When performing PCA, k is determined through a scree plot, which is the representation of the proportion of variance explained by each component. The optimal k corresponds to the number of principal components explaining a given level of the variance of the original data. However, this method has its limitation as stated in [Linting et al., 2007], and more importantly it is not suitable for our approach. Indeed, in our case, the number of selected principal components must achieve a given trade-off between information and noise in the latent space. If we consider a too low number of principal components, we may lose information regarding the normal observations,

leading to false alarms. If a too high number of principal component is retained, we may include components representing noise, which prevents the model from detecting some anomalies.

To select the optimal dimension of the latent space, we consider the distribution of anomaly score obtained through the application of naive approach Section 2.4, empirically calibrating a cut-off $\hat{s}$ through the non parametric estimation of the distribution of the anomaly scores of uncontaminated and contaminated time series. We chose to calibrate the dimension of the latent space with the naive approach, because using the NN to this end would be very costly. Hence, for each $k = 5, 10, \ldots, 200$, we construct a PCA model from which we infer reconstruction errors which are then converted into anomaly scores. Based on these anomaly scores, we tune the cut-off value $\hat{s}$ thanks to the distributions and finally convert the scores into labels. We evaluate the predictions of the naive approach for each value of k. The results on the evaluation metrics on the train set, as legitimate to choose k, are represented on Figure 5. The highest values are reached for $k \in \{40, \ldots, 145\}$. The performance seems to be stable in terms of F1-score and accuracy. Therefore, for computational reasons we choose the optimal k to be 40.



a) b)

Figure 5: Performance metrics on train a) and test b) sets with respect to the number of principal component k.

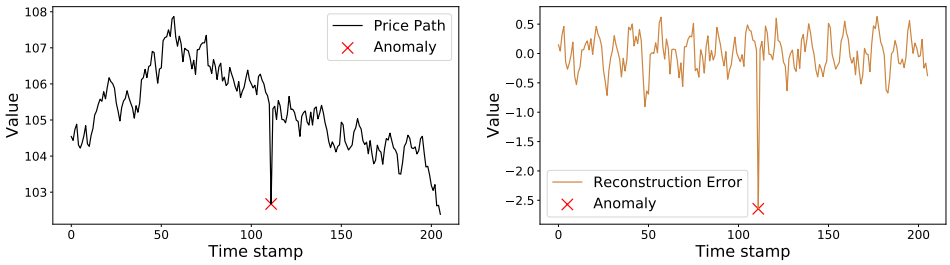
Figure 5 also shows, as expected, a downward trend of the scores when represented against the highest values of k. This demonstrates that when a high level of variance is explained, it become much harder to perform anomaly detection based on the reconstruction errors.

4.4 Contaminated Time Series Identification Step

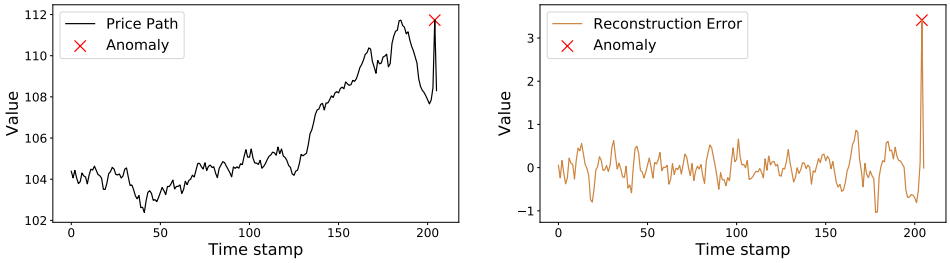
For the feature extraction step, we considered a latent space dimension $k = 40$. The NN built to compute the anomaly scores and convert them into labels was calibrated on the train set. The result of this calibration is shown in Table 3.

Data set	Accuracy	Precision	Recall	F1-score
Train set	90.97 %	97.36%	84.21%	90.31 %
Test set	88.58%	61.26%	85.27%	71.30%

Table 3: Performance evaluation of suggested model on synthetic data set for identification step



a)

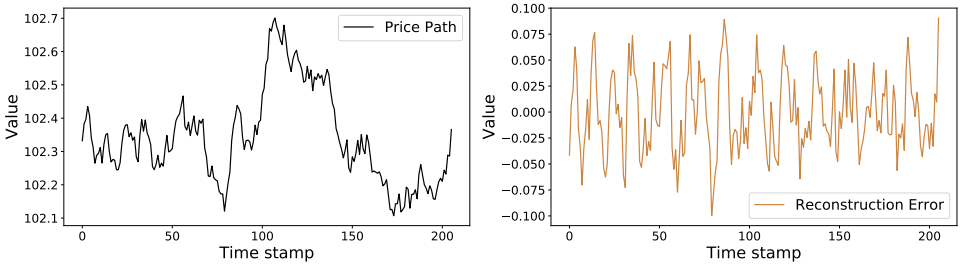


b)

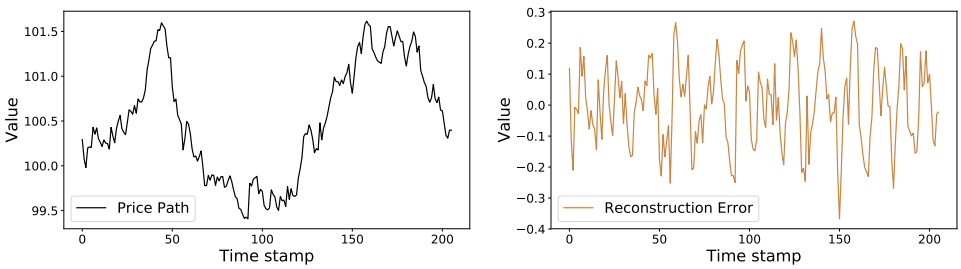
Figure 6: Two examples a) and b) of contaminated time series accurately identified by the model. The stock path and the reconstruction errors are represented in black and brown. The red cross shows the anomaly localization.

Figure 6 shows two contaminated time series identified as such by the model. Figure 7 displays two examples of time series without anomalies accurately identified by the model. Figure 8 displays two time series misidentified by the model.

When an observation deviates significantly from the rest of the time series values, the model is able to recognise that the concerned time series contains an abnormal observation.



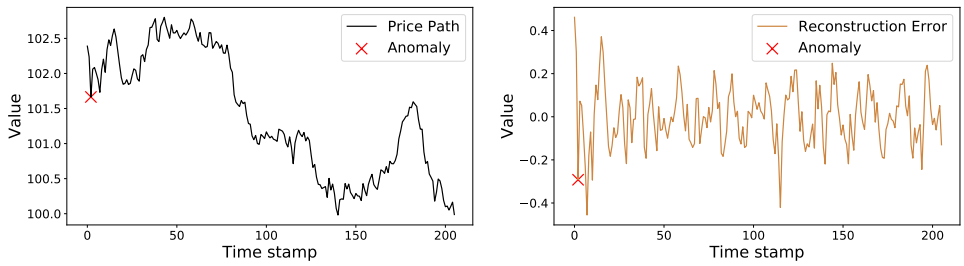
a)



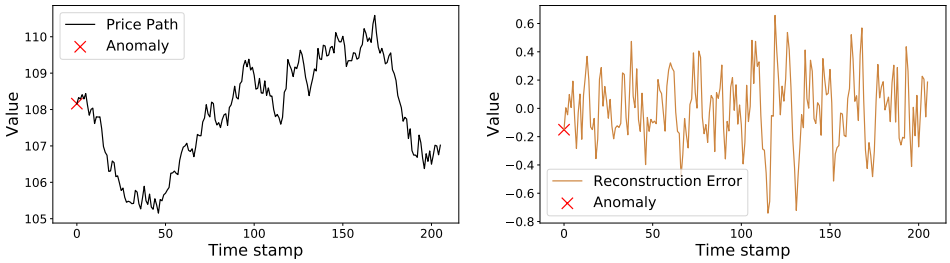
b)

Figure 7: Two examples a) and b) of uncontaminated time series accurately identified by the model. The stock path and the reconstruction errors are represented in black and brown.

In Figure 7 with uncontaminated time series, we see that even when there is a local upward trend in the time series values, the model is able to make the distinction between this market move and the occurrence of an anomaly.



a)



b)

Figure 8: Two examples a) and b) of time series misidentified by the model. The stock path and the reconstruction errors are represented in black and brown. The red cross shows the anomaly localization.

The last set of time series displays some limits of the model. The graphs in Figure 8a) represent the situation where the model did not manage to identify the contaminated time series. One explanation could be that the size of the anomaly is not significantly large enough to be spotted by the model. Indeed looking at the graph the time series does not seem to contain any anomaly. In contrast, the graphs in Figure 8b) shows a stock price path predicted to be contaminated whereas it is not.

We illustrate the robustness of the approach by assessing the model predictions on 100 distinct data sets. Table 4 shows the mean and standard deviation over the multiple runs.

Data set	Accuracy	Precision	Recall	F1-score
Train set	79.02% (2.4%)	78.69% (2.7%)	79.74% (4.6%)	79.13% (2.7%)
Test set	77.79% (3.9%)	41.87% (5.2%)	80.16% (6.4%)	54.82% (5.2%)

Table 4: Mean (standard deviation) of performance metrics of the suggested model over multiple runs for the identification step.

4.5 Anomaly Localization Step

Regarding the localization step, the dummy approach consists in taking the argmax of time series observations as the anomaly location. We distinguish between two cases, anomalies which are extrema and anomalies which are not. The quality of the detection of our model is assessed for both types of anomalies. The results are displayed in Tables 5 and 6.

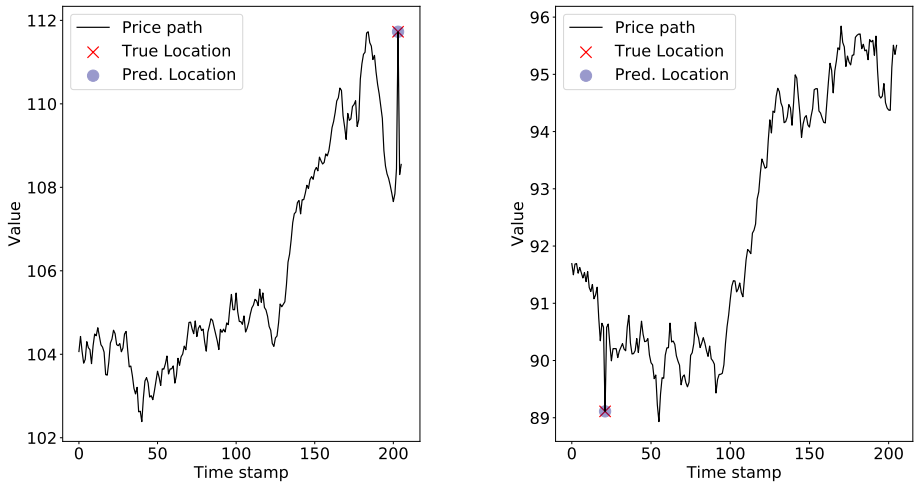
Data set	Accuracy	Precision	Recall	F1-score
Train set	89.65% (34.87%)	89.81% (40.53%)	89.65% (34.88%)	89.68% (36.56%)
Test set	94.39% (27.78%)	94.49% (31.52%)	94.39% (27.78%)	94.38% (28.94%)

Table 5: Performance evaluation of the suggested model and the dummy approach on synthetic data set for the location step on all type of anomalies.

Data set	Accuracy	Precision	Recall	F1-score
Train set	84.22% (0%)	84.55% (0%)	84.28% (0%)	89.68% (0%)
Test set	91.92% (0%)	92.10% (0%)	91.92% (0%)	91.90% (0%)

Table 6: Performance evaluation of the suggested model and the dummy approach on synthetic data set for the location step on non extrema anomalies.

Numerical tests show the necessity of the feature extraction step for the anomaly location task, as applying the dummy approach alone is not enough when the anomaly is not an extreme value. Figure 9 represents a stock price time series with the true and predicted anomaly location. In these examples, the locations are accurately predicted.



a) b)

Figure 9: Anomaly localization prediction on two distinct time series a) and b).

To guarantee the robustness of the model on the anomaly location, we evaluate the model prediction on 100 data sets. Table 7 shows the mean and standard deviation for the localization of all types of anomaly.

Data set	Accuracy	Precision	Recall	F1-score
Train set	89.58% (4.3%)	89.58% (4.3%)	89.96% (4.1%)	89.67% (4.3%)
Test set	89.49% (4.8%)	89.49% (4.8%)	90.00% (4.5%)	89.58% (4.7%)

Table 7: Mean (standard deviation) of performance metrics of suggested model over multiple runs for location step on all type of anomalies.

Table 8 displays the results on the localization of non extrema anomalies. These results show the importance of the feature extraction step. If we only consider the maximal observed value of

the time series to be the anomaly, we would not be able to localise any non-extrema anomaly. Applying Algorithm 4 instead leads to satisfying anomaly detection rates.

Data set	Accuracy	Precision	Recall	F1-score
Train set	85.23% (6.0%)	85.23% (6.0%)	85.87% (5.7%)	85.36% (6.0%)
Test set	85.17% (7.1%)	85.17% (7.1%)	86.04% (6.8%)	85.30% (7.1%)

Table 8: Mean (standard deviation) of performance metrics of suggested model over multiple runs for location step on non-extreme anomalies.

4.6 Numerical Results Against Benchmark Models

To assess the performance, the suggested approach is compared with well-known machine learning algorithms for anomaly detection, that is isolation forest (IF), local outlier factor (LOF), density based clustering of applications with noise (DBSCAN), k-nearest neighbors (KNN), support vector machine (SVM), reviewed in Section A. We assess their performance on the data set described in Section 3. Their results are given in this section.

Tables 9 and 11 summarize the performance on train and test sets of each model for both the contaminated time series identification and anomaly localization steps.

	Train		Test	
Model	Accuracy	F1-score	Accuracy	F1-score
IF	42.31%	42.31%	69.64%	7.022%
LOF	59.95%	59.95%	90.00%	62.41%
DBSCAN	50.00%	66.67%	16.65%	28.53%
KNN	94.68%	94.39%	64.82%	26.69%
SVM	81.96%	82.33%	44.39%	27.44%
PCA NN	90.97%	90.31%	88.58%	71.30%

Table 9: Performance evaluation of supervised (bottom) and unsupervised (top) models for contaminated time series identification step.

Algorithm	IF	LOF	DBSCAN	KNN	SVM	PCA NN
Exec. Time	0.6915	0.2015	0.1275	1.725	4.572	0.003523

Table 10: Execution time in seconds for identification of contaminated time series step.

The following paragraphs provide similar conclusions drawn from the results on the identification and location steps (see Tables 9 and 11).

For the unsupervised learning methods, since there is not a proper learning step, even though Tables 9 and 11 show the scores on the train and test set, these scores should be seen as the ones obtained by testing the models on independent data sets. For IF, LOF and DBSCAN models, the performance across the sets is not stable, as a significant difference could be observed between the scores on the train and test sets. The poor performance of unsupervised learning algorithms could be explained by the difficulty to estimate the contamination rate for IF and the LOF algorithms. For DBSCAN the poor performance is rather due to the high dimensionality of the data.

Regarding the supervised approaches, although KNN is outperforming the suggested approach on the training set, there is a non-negligible decrease of these scores on the test set. This may suggest an over-fitting on the training data, which makes KNN unable to generalize what it has learnt to unseen samples. The same trend is seen on the scores with our approach, however the loss is no as harsh as in the KNN case.

Hence, according to the results, our PCA based approach seem to be the most suitable method for the problem at stake of anomaly detection on time series. Besides the relatively satisfying performance it shows in terms of accuracy and F1-score, its low computational cost (Tables 10 and 12) for both steps, makes the approach stand out.

	Train		Test	
Model	Accuracy	F1-score	Accuracy	F1-score
IF	89.79%	2.296%	70.94%	1.611%
LOF	99.51%	0%	2.066%	0.9816%
DBSCAN	N/A	NA	NA	NA
KNN	99.99%	99.99%	95.56%	2.794%
SVM	NA	NA	NA	NA
PCA NN	89.65%	89.68%	94.39%	94.38%

Table 11: Performance evaluation of supervised (bottom) and unsupervised (top) models for anomaly localization step. Results for DBSCAN and SVM are not provided due to high computational cost.

Algorithm	IF	LOF	DBSCAN	KNN	SVM	PCA NN
Exec. Time	14.50	2.287	N/A	14.19	N/A	0.002004

Table 12: Execution time in seconds for the anomaly localization step.Results for DBSCAN and SVM are not provided due to high computational cost.

4.7 Anomalies Imputation

To assess the imputation values suggested by our approach, we start from time series simulated without anomalies, then we randomly select a time stamp of the time series and add a noise to the corresponding value following the methodology described in Section 3.

The imputation value the PCA NN suggests is the reconstructed observation. This imputation technique is compared to naive methods of missing values imputation, namely backward fill (BF), consisting in replacing the anomaly by the previous value, and linear interpolation (LI). The choice of these imputation methods is motivated by their low computational cost. In fact, the PCA NN imputation comes at no additional cost as stated before. Therefore, we only challenge its performance with methods with similar complexity. To assess the quality of the imputation value suggested by each approach, we consider the following metrics. The imputation errors ImputationError^i are computed on each time series as

$$\text{ImputationError}^i = \sqrt{\sum_j \frac{\left(S^i_{t_j} - \widetilde{S}^i_{t_j}\right)^2}{n^{anomaly}}},$$

where $\widetilde{S}^i$ refers to as the i-th path price with imputed values. The error on the covariance matrix is computed as

$$\text{ErrCov} = \|\Sigma - \widetilde{\Sigma}\|_{\text{Frob}},$$

where $\|\cdot\|_{\text{Frob}}$ is the Frobenius norm, Σ the sample covariance matrix and $\widetilde{\Sigma}$ the covariance matrix estimated on the data after the anomalies have been replaced by their respective imputation values.

Each stock path was diffused and contaminated 100 times. The anomalies were then imputed following two baseline approach (BF and LI). Figure 11 clearly shows that imputation using the reconstructed values with PCA reduces the error on the estimation of the covariance matrix compared to the situation where the covariance matrix is estimated on the data with anomalies. However, Table 13 shows that the baseline imputation approaches achieve even lower errors on the estimation of the covariance matrix. Figure 12 shows a higher variance of the errors for the PCA-based imputation approach. In conclusion, it would be rather recommended to replace the flagged anomalies with approaches such as backward fill or linear interpolation.

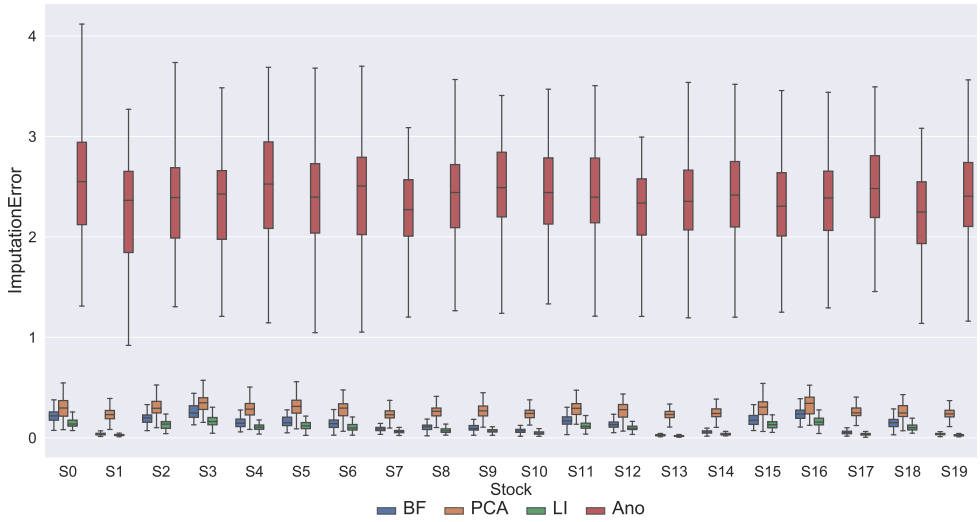


Figure 10: Boxplot representation of the distribution of the errors on time series imputation before the imputation of anomalies (Ano) and after replacing the anomalies; with the reconstructed values suggested by PCA (PCA), using linear interpolation (LI) and using backward fill (BF).

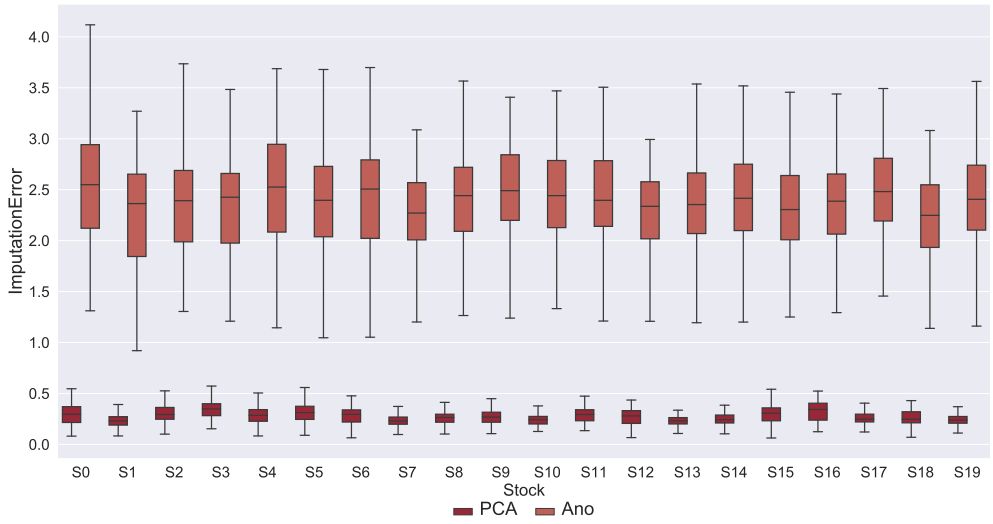


Figure 11: Boxplot representation of the distribution of the errors on time series imputation before the imputation of anomalies (Ano) and after replacing the anomalies with the reconstructed values suggested by PCA NN (PCA).

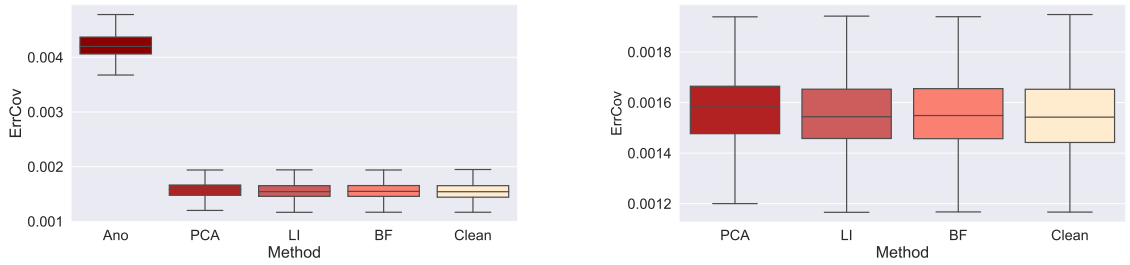


Figure 12: Boxplot representation of the distribution of the errors on covariance matrix (on several stocks paths samples). Ano, PCA NN, LI, BF and Clean refer to the errors on the covariance when estimated on time series with anomalies, time series after anomalies imputation following three approaches: imputation by the reconstructed values suggested by PCA (PCA), using linear interpolation (LI), with backward fill (BF) and time series without anomalies (Clean). The Ano, PCA NN, LI, BF and Clean errors on the covariance estimation are all represented in a). For better visualization we remove the Ano errors in b).

Method	Ano	PCA	LI	BF	Clean
Mean	0.004208	0.001575	0.001554	0.001554	0.001552
Standard deviation	0.000272	0.000165	0.000170	0.000172	0.000173

Table 13: Mean and standard deviation error on covariance matrix after imputation of anomalies with baseline methods.

A natural explanation to the relative inefficiency of the reconstruction value as imputation value comes from the fact that by definition the reconstructed value integrated in its computation the abnormal value, which is not the case when using other imputation techniques. Therefore even if the reconstructed value is closer to the true value than it is to the abnormal value, the spread between the imputation and the true value is still significant. Figures 13 and 14 illustrate the latter with a plot of a reconstructed and original price path.

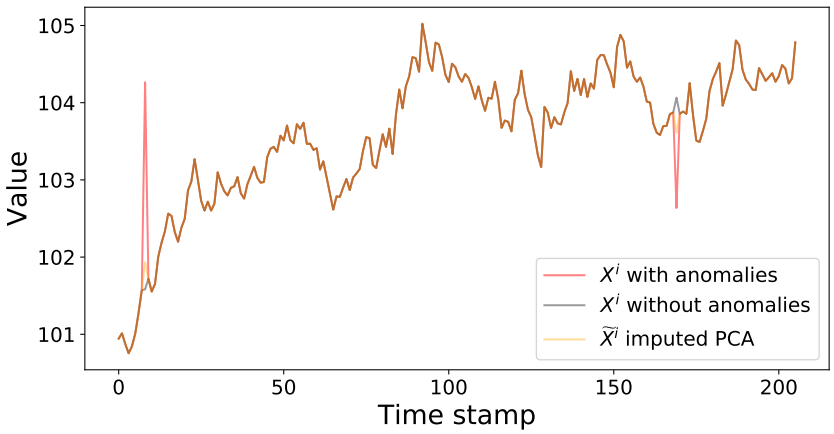


Figure 13: Original, contaminated and reconstructed stock price path.

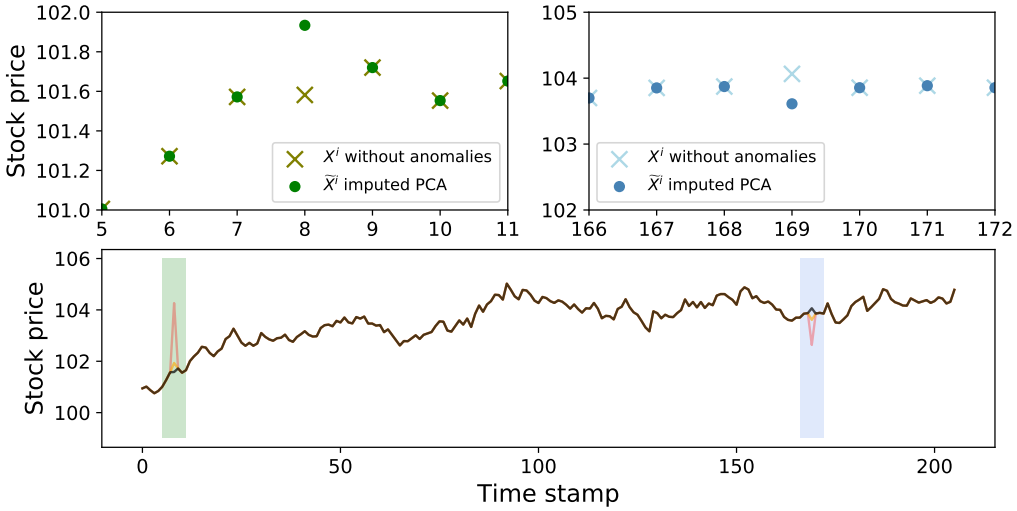


Figure 14: Original, contaminated and imputed stock prices path in black, red and orange. The two additional graphs represent the region around anomalies, where the cross and circle respectively show the true value of the stock and its value after imputation of the anomalies using the reconstructed value suggested by the PCA NN model.

4.8 Cut-off Value Robustness

In most anomaly detection models, the cut-off value is a hand-set parameter. In our approach, the cut-off value is a model parameter and as such it is calibrated through the learning. Testing the robustness of the cut-off given by the model is therefore a must. The identification step is the unique step of the approach being concerned by the robustness, since it is the only step involving the cut-off calibration. Robustness is checked by shocking the suggested cut-off with different level of noise and observing the impact of these shocks on the model performance. We consider several shocks amplitude $\gamma \in \pm\{10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, 1, 2\}$. Table 14 reports the mean and standard deviation of the scores of the model. The cut-off calibrated on the synthetic training data sample is shocked. Scores on both train and test sets are computed using the shocked cut-off values.

γ	Accuracy	Precision	Recall	F1
10^{-4}	0.9097	0.9736	0.8421	0.9031
-10^{-4}	0.9097	0.9736	0.8421	0.9031
10^{-3}	0.9097	0.9738	0.8419	0.9031
-10^{-3}	0.9098	0.9737	0.8424	0.9033
10^{-2}	0.9092	0.9747	0.8403	0.9025
-10^{-2}	0.9098	0.9720	0.8439	0.9035
10^{-1}	0.9042	0.9860	0.8201	0.8954
-10^{-1}	0.9103	0.9563	0.8599	0.9056
0	0.9097	0.9736	0.8421	0.9031
1	0.7771	0.9994	0.5546	0.7133
-1	0.5183	0.5093	0.9998	0.6749
2	0.6284	1.0000	0.2567	0.4086
-2	0.5001	0.5000	1.0000	0.6667

a)

γ	Accuracy	Precision	Recall	F1
10^{-4}	0.8858	0.6126	0.8527	0.7130
-10^{-4}	0.8858	0.6126	0.8527	0.7130
10^{-3}	0.8858	0.6126	0.8527	0.7130
-10^{-3}	0.8850	0.6105	0.8527	0.7116
10^{-2}	0.8877	0.6179	0.8527	0.7166
-10^{-2}	0.8838	0.6074	0.8527	0.7095
10^{-1}	0.8984	0.6502	0.8432	0.7342
-10^{-1}	0.8672	0.5653	0.8741	0.6866
0	0.8858	0.6126	0.8527	0.7130
1	0.9391	0.9435	0.6746	0.7867
-1	0.2660	0.1848	1.0000	0.3120
2	0.8949	0.9814	0.3753	0.5430
-2	0.1700	0.1670	1.0000	0.2862

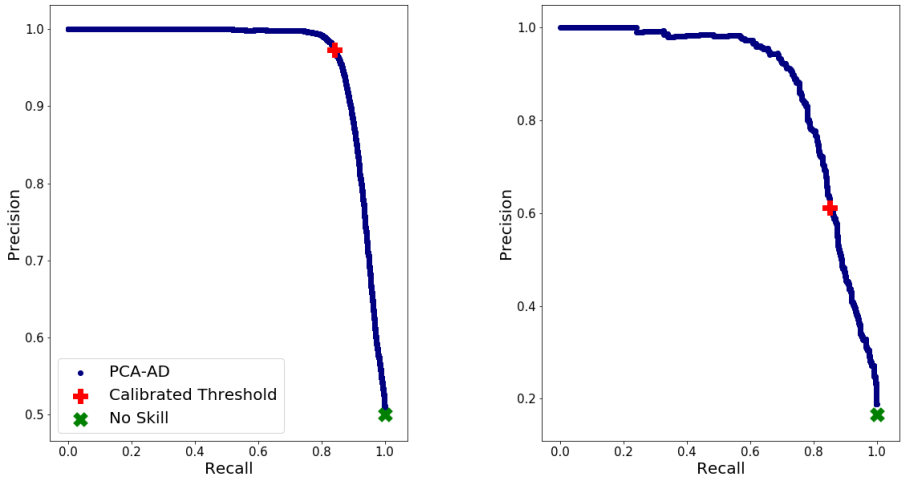
b)

Table 14: Performance evaluation after shocking the calibrated cut-off value for a) the training set and b) for the test set.

Excluding the extreme cases where the shock amplitude is $\pm\{1, 2\}$, one could see that the accuracy is barely impacted by the shocked cut-off values both on the train and test sets. The interpretation of the remaining results is split in two and is applicable for both the training and test sets.

When negative shocks are applied, the model predicts more anomalies and less normal observations (compared to the predicted numbers with the calibrated cut-off value). Therefore, the model is able to identify anomalies which were missed initially. Hence, applying negative shocks increases the recall. As for the precision, i.e. the rate of correctly identified contaminated time series, since on left hand side of the cut-off value we are in the density region where contaminated time series represent the dominant class, the new position of the cut-off value leads to a misidentification of this type of observations. This entails a deterioration of the precision. Opposite behaviours of the precision and recall are observed when positive shocks are applied since some abnormal time series are missed by the model.

Figure 15 shows the precision and recall when cut-off values other than the one we calibrated are used in our model, on the train and test sets. These scores are also compared with the



a) b)

Figure 15: Precision-recall curve of the suggested model (in blue), scores with the calibrated threshold (red plus), no skill model scores (green cross) for the training set a) and a test set b).

performance of the no skill model[1]. We note that our approach edges out the no skill model on both sets. The area under the curve (AUC) score on the train set and the test set are respectively 0.97 and 0.87 (as a reminder, AUC score ranges from 0 to 1, 1 being the score associated with a perfect model). This shows that we are better performing on the train set, which was expected, but the performance on test set is just as satisfying. These results reinforce our conclusions regarding the robustness of our approach. One can see that the calibrated cut-off value represents, for the train set, the point where an equilibrium is being reached between precision and recall. The calibrated cut-off allows to achieve high precision and recall scores simultaneously. Thus we conclude that the cut-off given by the model is suitable for the training samples and for unseen samples as well.

To briefly sum up this section, although we mentioned that some shocks on the cut-off induce higher scores, the improvement over the scores with the calibrated cut-off is not significant (unless high amplitude shocks are considered). The numerical tests and the precision-recall curves are consistent with the robustness of the cut-off value suggested by the approach.

5 Application to a Downstream Task: Value-at-Risk Computations

In this section, we illustrate the benefit that could be drawn when applying the PCA NN approach as pre-processing step for value-at-risk computations.

Given a random variable ΔX representing the potential loss in portfolio position over a time horizon Δt, the value-at-risk quantile at level $1-\alpha$, $\mathrm{VaR}_{1-\alpha}$, is defined by

$$\mathrm{VaR}_{1-\alpha}(\Delta X) := -\inf\{x : \mathbb{P}(\Delta X \leq x) \geq 1-\alpha\}.$$

The $\mathrm{VaR}_{1-\alpha}$ over a time horizon Δt corresponds to the greatest loss that may be exceeded with probability α in the period Δt. Let $(S_t)_{t=t_0,\dots,t_T}$ be a path price diffusion distributed according to the Black-Scholes model (9). The logarithmic returns to maturity are distributed according to a Gaussian distribution, i.e.

[1]Here, no skill model being the model assigning to all observations the positive label

$$\ln\left(\frac{S_{t+dt}}{S_t}\right) \sim \mathcal{N}\left(\left(\mu - \frac{\sigma^2}{2}\right)dt, \sigma^2 dt\right), \tag{15}$$

with $\mu \in \mathbb{R}$ and $\sigma > 0$.

We assume the vector $\boldsymbol{R}$ of log-returns on our stocks to be joint-normal,

$$\boldsymbol{R} = \left(R^1, R^2, \ldots, R^i, \ldots, R^N\right)^\top \sim \mathcal{N}_N\left(\mu_R, \Sigma_R\right),$$

where $R^i = \log\left(\frac{S^i_{t+dt}}{S^i_t}\right) \sim \mathcal{N}\left(\mu_{i,R}, \sigma^2_{i,R}\right)$ and Σ_R is the covariance matrix.

We consider a portfolio on N stocks, which return is given by $R^{Portfolio} = \mathcal{Q}^\top \boldsymbol{R}$. Hence, $R^{Portfolio} \sim \mathcal{N}\left(\mathcal{Q}^\top \mu_R, \mathcal{Q}^\top \Sigma_R \mathcal{Q}\right)$. The VaR quantile for the time horizon Δt and at level α is

$$VaR_{1-\alpha}\left(\mu_R, \Sigma_R\right) = q_{1-\alpha}\sqrt{\mathcal{Q}^\top \Sigma_R \mathcal{Q}} + \mathcal{Q}^\top \mu_R, \tag{16}$$

where $q_{1-\alpha}$ is the $(1-\alpha)$-quantile of a standard normal distribution and the parameters μ_R and Σ_R are estimated from different types of time series to evaluate the impact of identifying and removing anomalies following our approach.

Under the adopted framework, the true $\mathrm{VaR}_{1-\alpha}\left(\mu_R, \Sigma_R\right)$, $\mathrm{VaR}^{Theoretical}$ is thus known and can be computed using the diffusion parameters. An estimation $\mathrm{VaR}_{1-\alpha}\left(\widehat{\mu_R}, \widehat{\Sigma_R}\right)$ can be obtained by replacing the parameters in (16) by their estimates $\widehat{\mu_R}$ and $\widehat{\sigma_R}$ computed from the time series with anomalies, or after imputation of anomalies. Then, absolute errors and relative errors are computed as

$$\mathrm{AbsoluteError}_{\mathrm{VaR}} = \left|\mathrm{VaR}_{1-\alpha}\left(\mu_R, \Sigma_R\right) - \mathrm{VaR}_{1-\alpha}\left(\widehat{\mu_R}, \widehat{\Sigma_R}\right)\right|$$

$$\mathrm{RelativeError}_{\mathrm{VaR}} = \frac{\left|\mathrm{VaR}_{1-\alpha}\left(\mu_R, \Sigma_R\right) - \mathrm{VaR}_{1-\alpha}\left(\widehat{\mu_R}, \widehat{\Sigma_R}\right)\right|}{\mathrm{VaR}_{1-\alpha}\left(\mu_R, \Sigma_R\right)}$$

Table 15 summarizes the four VaR estimations we consider in the sequel:

VaR estimation Name	$\widehat{\mu_R}, \widehat{\Sigma_R}$ estimated on
VaR^{Clean}	Time series without anomalies
VaR^{Ano}	Times series with anomalies
$\mathrm{VaR}^{Loc,True}$	Time series after anomalies imputation knowing their true localization
$\mathrm{VaR}^{Loc,Pred}$	Time series after anomalies imputation based on predicted localization

Table 15: Notations of VaR estimation based on the time series the VaR parameters have been estimated from.

To conduct this analysis we generated new stocks paths samples that we assume to be clean, fixing the diffusion parameters μ_R and Σ_R, then we added the anomalies following the procedure described in Section 3. We apply our model to locate the anomalies and replace the identified anomalies using the backward fill (BF) approach (shown to be the more efficient imputation technique in Section 4.7. For each stock and each run of simulation, we obtain four estimates of the distribution parameters of the associated log-returns.

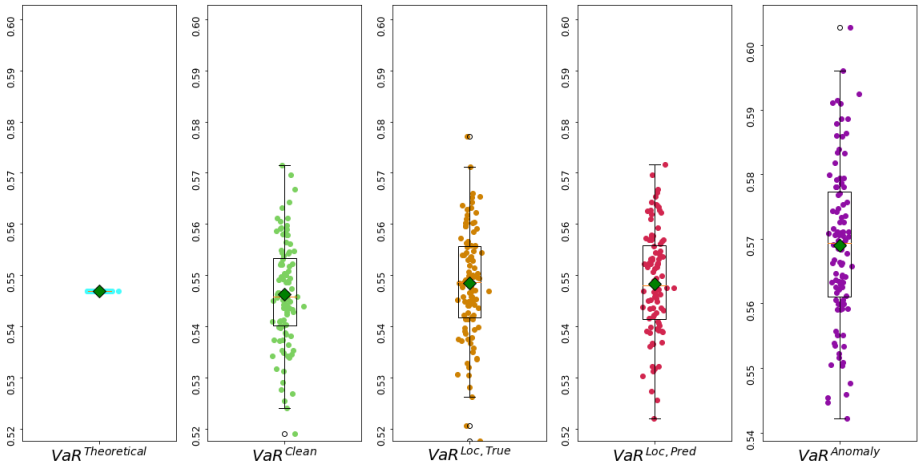


Figure 16: Boxplot representation of parametric VaR estimations for $\boldsymbol{R}^{Portfolio}$. The green square represents the mean of VaR estimation.

VaR	$\text{VaR}^{Theoretical}$	VaR^{Clean}	$\text{VaR}^{Loc,True}$	$\text{VaR}^{Loc,Pred}$	VaR^{Ano}
Mean	0.546851	0.546300	0.548392	0.548270	0.569015
Standard Deviation	0.0	0.010105	0.010739	0.010832	0.012268

Table 16: Summary of VaR estimations for $\boldsymbol{R}^{Portfolio}$.

Figure 16 and Table 16 summarize the distribution of the VaR estimates for $\alpha = 0.99$ and $\Delta t = 1(day)$, over several simulation runs. The boxplots show the dispersion of the portfolio VaR estimates on several diffusions. The green square represents the mean of VaR estimation. For the four first boxplots, the means are approximately on the same level, which is confirmed by the results of Table 16. The anomalies present among the time series observed values have a non-negligible impact on the distribution parameters estimation, which ultimately causes a wrong estimation of the VaR. Thanks to the localization of the anomalies by the suggested model and their imputation, we are able to get a more accurate estimation of the VaR. $\text{VaR}^{Loc,True}$ and $\text{VaR}^{Loc,Pred}$ are quite similar, which shows that the model accurately localizes the anomalies.

VaR	VaR^{Clean}	$\text{VaR}^{Loc,True}$	$\text{VaR}^{Loc,Pred}$	VaR^{Ano}
Absolute Error	0.007995	0.008596	0.008622	0.02235
Relative Error	0.014620	0.015720	0.015767	0.04087

Table 17: Mean Absolute and relative error on VaR estimations for $\boldsymbol{R}^{Portfolio}$.

We also evaluate the error on the VaR estimation using the mean absolute error and the mean relative error, taking the $\text{VaR}^{Theoretical}$ as our benchmark VaR. As one can tell from Table 17, even when the distribution parameters are estimated from the clean time series, the VaR computed with these parameters is not exactly the one computed with the theoretical parameters. This can be explained by the historical size of the observed values used to estimate the parameters. This table shows that by removing anomalies we can reduce by a factor two the error on VaR estimation.

Additionally, we assess the impact on $\text{VaR}^{Theoretical}, \text{VaR}^{Clean}, \text{VaR}^{Ano}, \text{VaR}^{Loc,True}$ and $\text{VaR}^{Loc,Pred}$ of increasing $n^{anomaly}$. To this end, we perform 50 simulation runs of stocks paths

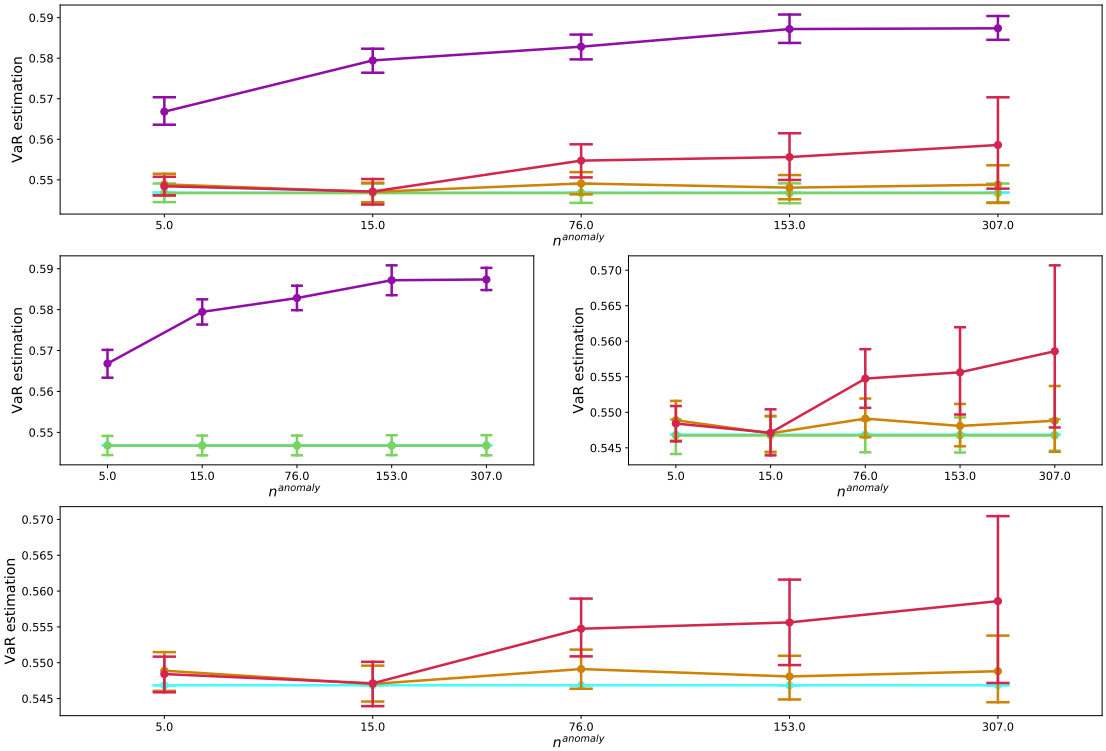


Figure 17: VaR estimation based on parameter estimation from time series with and without anomalies (purple and light blue curves), time series after identification and imputation of anomalies with the suggested approach (red curve) and time series imputed knowing the true location of the anomalies (brown curve), with respect to $n^{anomaly}$.

for $n^{anomaly}$ and for each of those scenarios we estimate the VaR on the portfolio. We summarize the results on Figure 17, where each curve represents the mean VaR estimation with respect to $n^{anomaly}$ along with a representation of the uncertainty around each evaluated point through a confidence interval.

When we compute the VaR using the abnormal time series, we notice that the difference between VaR^{Ano} and $\text{VaR}^{Theoretical}$ increases with $n^{anomaly}$, which is natural to expect. However, when the time series are cleaned prior to VaR estimation, the curve representing the VaR estimates are much closer to the ones representing $\text{VaR}^{Theoretical}$ and VaR^{Clean}, showing an undeniable improvement in the accuracy of the VaR estimation over the estimation based on abnormal time series. Furthermore, VaR estimation after the imputation following the model prediction or knowing the true location of the anomalies seem to be quite similar for low $n^{anomaly}$, while some discrepancies between the two become more significant as $n^{anomaly}$ increases. A natural explanation could be that when the number of anomalies increases and the model suggests wrong anomalies location, normal values are being replaced while true anomalies remain among the observed values, which wrongly impacts the VaR estimation. However, the results of Tables 18 and 19 indicate that the anomalies localization suggested by the model are overall correct and allow removing most of the anomalies, as the relative error of $\text{VaR}^{Loc,Pred}$ is, regardless of $n^{anomaly}$, always lower than the relative error of VaR^{Ano} (e.g a relative error of 0.0248 for $\text{VaR}^{Loc,Pred}$ against 0.0658 for VaR^{Ano} when there is 76 anomalies among the $1,500$ observed

values of the time series).

$n^{anomaly}$	VaR^{Clean}	$\text{VaR}^{Loc,True}$	$\text{VaR}^{Loc,Pred}$	VaR^{Ano}
5	0.012194	0.013796	0.013341	0.037392
15	0.012194	0.012623	0.016676	0.059604
76	0.012194	0.014831	0.024807	0.065791
153	0.012194	0.014644	0.034361	0.073750
307	0.012194	0.020537	0.052244	0.074091

Table 18: Mean relative error of parametric VaR estimations with respect to $n^{anomaly}$, for parameters estimated from clean time series, time series with anomalies, imputed time series following predicted location ($\text{VaR}^{Loc,Pred}$) and true anomalies location ($\text{VaR}^{Loc,True}$).

$n^{anomaly}$	VaR^{Clean}	$\text{VaR}^{Loc,True}$	$\text{VaR}^{Loc,Pred}$	VaR^{Ano}
5	0.010356	0.011511	0.009678	0.020674
15	0.010356	0.010514	0.012723	0.020172
76	0.010356	0.011842	0.018696	0.019930
153	0.010356	0.013683	0.026117	0.024243
307	0.010356	0.022526	0.057903	0.019207

Table 19: Standard deviation of relative error of parametric VaR estimations with respect to $n^{anomaly}$, for parameters estimated from clean time series, time series with anomalies, imputed time series following predicted location ($\text{VaR}^{Loc,Pred}$) and true anomalies location ($\text{VaR}^{Loc,True}$).

6 Conclusion

The suggested approach targets anomaly detection on panel of time series possibly reflecting a wide variety of risk factors. We suggest a model able to detect anomalies in risk factor time series suitable for all risk factors. Anomaly detection is achieved in two steps. The first step aims at identifying the contaminated time series (time series with anomalies). The second step focuses on the localization of the anomaly among the observed values of the identified contaminated time series.

In addition, our methodology integrates feature extraction from the time series with PCA. This part of the method is proved to be an essential part of the model, as it provides the models with inputs on which the distinction between abnormal/contaminated and normal instances is eased, and also ensures the stationnarity of the model (time series) inputs. Another main focus of the approach is the calibration of the cut-off, the key parameter in the identification of contaminated time series, by means of feedforward network with a customized loss function.

The proposed approach suggests an imputation value, however this value is strongly impacted by the anomaly value. As a result, naive approaches with similar complexity are preferred.

Furthermore, our numerical experiments do not only show that our approach outperforms baseline anomaly detection models, it also shows the true benefit one could draw from cleaning time series with the suggested approach, as illustrated on the VaR computation task.

A State of the Art Anomaly Detection Models

There are mainly two categories of machine learning anomaly detection models : *density-based models*, where a distribution is used to fit the data and anomalies are defined relatively to this distribution, and *depth-based models*, which, instead of modeling the normal behaviour, isolate anomalies.

A.1 Density-Based Models

Density based spatial clustering with noise (DBSCAN) DBSCAN [Hinneburg, 1996] [Schubert et al., 2017] is an unsupervised clustering methodology that groups together comparable observations based on a similarity metric. Clusters are high density regions and are defined by the ε-neighbourhood of observations and by *MinPts*, the minimum number of points required to be in a radius of ε from an observation to form a dense region.

An anomaly is any observation which has not $MinPts$ in its ε-neighbourhood nor appears in the ε-neighbourhood of other observations.

K-nearest neighbours (KNN) Usually used for classification purposes, KNN [Hand, 2007] is a supervised algorithm that can be used for anomaly detection as well.For each observation it selects its closest k observations. Generally in terms of distance, but other similarity metrics can be considered. The anomaly score of an observation is computed as function of its distances to k-nearest neighbours, weighted average of the distances for instance. The points with the highest anomaly scores are considered as anomalies.

Support Vector Machines (SVM) The ultimate aim of SVM [Cortes and Vapnik, 1995] is to define a hyperplan that separates our data. The specificity of this hyperplan is that it maximizes the distances to the set of features representing each class. When the data is not linearly separable, a map ϕ is applied to the initial features vector so the data became linearly separable in the new space in which it was projected.

A.2 Depth-Based Models

Isolation Forest (IF) IF [Hariri et al., 2019] applies a depth approach to detect anomalies. The algorithm is based on the idea that anomalies are easier to isolate and thus will be isolated closer to the root while normal observations are isolated much further from the root. The algorithm uses random decision trees to separate the observations. The anomaly score is calculated as the path length to isolate the observation. As it is defined in the algorithm, the anomaly score will be closer to 1 for anomalies and $<< 1$ for normal observations. This allows ranking the observations, from the most abnormal observation the most regular one. However, the choice of an explicit cut-off between this two type of instances is not obvious.

Local Outlier Factor (LOF) The LOF algorithm [Breunig et al., 2000], [Alghushairy et al., 2020] tries to assess the isolation of one observation relatively to the rest of the Data set, before flagging it as an anomaly. This model relies on the concept of local density. The anomaly score of an observation x is its local outlier factor,which quantifies how dense is the location area of x compared to the one of its neighbors. Hence, for each observation, $LOF_k(x) \approx 1$ means that the density of observations around x is similar to the one of its neighbours, therefore x could not be considered as an isolated observation. $LOF_k(x) >> 1$, instead, shows that the density of x is lower compared to its neighbours, hence x should be flagged as anomaly

B Stationary time series

Assume that a process $\left(\varepsilon_t^i\right)$ satisfies the following representation

$$\Delta\varepsilon_t^i = \gamma\varepsilon_{t-1}^i + \theta_1\Delta\varepsilon_{t-1}^i + \ldots + \theta_{p-1}\Delta\varepsilon_{t-p+1}^i + z_t,$$

where p is the lag order, Δ is the difference operator,i.e $\Delta\varepsilon_t = \varepsilon_t - \varepsilon_{t-1}$ and z_t a white noise. The stationarity of (ε_t^i) is shown using the Augmented Dickey fuller test [Fuller, 2009]. Namely, the process ε^i is stationary if there is a unit root, i.e. $\gamma = 0$. Therefore, for each ε^i the test is carried under the null hypothesis

$$\mathcal{H}_0 : \gamma = 0 \quad \text{against} \quad \mathcal{H}_1 : \gamma < 0.$$

Set	Mean	Standard dev.	Min	25%	50%	75%	Max
Train	1.2578e-14	1.7389e-13	6.6310e-24	3.4635e-20	1.9166e-18	3.7363e-17	3.8829e-12
Test	1.5096e-14	4.8256e-13	6.2027e-30	4.2255e-19	8.4549e-18	1.6491e-16	4.3468e-11

Table 20: Descriptive statistics on p-values for train and test set time series $\boldsymbol{\varepsilon}$

If the p-values are lower than the significance level, then the null hypothesis is rejected for all the reconstruction errors and we conclude that the time series we work with do not suffer from the non-stationarity issue.

References

C. C. Aggarwal, A. Hinneburg, and D. A. Keim. On the surprising behavior of distance metrics in high dimensional space. In *International conference on database theory*, pages 420–434. Springer, 2001.

M. Ahmed, A. N. Mahmood, and J. Hu. A survey of network anomaly detection techniques. *Journal of Network and Computer Applications*, 60:19–31, 2016.

E. Akyildirim, M. Gambara, J. Teichmann, and S. Zhou. Applications of Signature Methods to Market Anomaly Detection. *arXiv preprint arXiv:2201.02441*, 2022.

O. Alghushairy, R. Alsini, T. Soule, and X. Ma. A review of local outlier factor algorithms for outlier detection in big data streams. *Big Data and Cognitive Computing*, 5(1):1, 2020.

H. Ali, M. N. M. Salleh, R. Saedudin, K. Hussain, and M. F. Mushtaq. Imbalance class problems in data mining: A review. *Indonesian Journal of Electrical Engineering and Computer Science*, 14(3):1560–1571, 2019.

M. Allouche, S. Girard, and E. Gobet. EV-GAN: Simulation of extreme events with ReLu neural networks. *Journal of Machine Learning Research*, 23(150):1–39, 2022.

Basel Committee on Banking Supervision. Consultative document: Fundamental Review of the Trading Book: A revised market risk framework, 2013. URL `http://www.bis.org/publ/bcbs265.pdf`.

Y. Bengio, I. Goodfellow, and A. Courville. *Deep learning*, volume 1. MIT press Cambridge, MA, USA, 2017.

X. Bin, Y. Zhao, and B. Shen. Abnormal Subspace Sparse PCA for Anomaly Detection and Interpretation. *arXiv preprint arXiv:1605.04644*, 2016.

M. Braei and S. Wagner. Anomaly detection in univariate time-series: A survey on the state-of-the-art. *arXiv preprint arXiv:2004.00433*, 2020.

M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander. Lof: identifying density-based local outliers. In *Proceedings of the 2000 ACM SIGMOD international conference on Management of data*, pages 93–104, 2000.

J. Brownlee. *Imbalanced Classification with Python: Better Metrics, Balance Skewed Classes, Cost-Sensitive Learning*. Machine Learning Mastery, 2020.

V. Chandola. *Anomaly detection for symbolic sequences and time series data*. University of Minnesota, 2009.

M. Chataigner, S. Crépey, and J. Pu. Nowcasting networks. *Journal of Computational Finance*, 24(3), 2020.

Y.-C. Chen. A tutorial on kernel density estimation and recent advances. *Biostatistics & Epidemiology*, 1(1):161–187, 2017.

Y. Cheng, I. Diakonikolas, R. Ge, and D. Woodruff. Faster Algorithms for High-dimensional Robust Covariance Estimation. *arXiv e-prints*, pages arXiv–1906, 2019.

N. Chinchor and B. M. Sundheim. Muc-5 evaluation metrics. In *Fifth Message Understanding Conference (MUC-5): Proceedings of a Conference Held in Baltimore, Maryland, August 25-27, 1993*, 1993. URL `https://aclanthology.org/M93-1007.pdf`.

E. M. Compagnoni, L. Biggio, A. Orvieto, T. Hofmann, and J. Teichmann. Randomized Signature Layers for Signal Extraction in Time Series Data. *arXiv preprint arXiv:2201.00384*, 2022.

C. Cortes and V. Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

Z. Cui, W. Chen, and Y. Chen. Multi-scale convolutional neural networks for time series classification. *arXiv preprint arXiv:1603.06995*, 2016.

A. Dempster, F. Petitjean, and G. I. Webb. Rocket: exceptionally fast and accurate time series classification using random convolutional kernels. *Data Mining and Knowledge Discovery*, 34 (5):1454–1495, 2020.

M. Ding and H. Tian. Pca-based network traffic anomaly detection. *Tsinghua Science and Technology*, 21(5):500–509, 2016.

C. Esteban, S. L. Hyland, and G. Rätsch. Real-valued (medical) time series generation with recurrent conditional gans. *arXiv preprint arXiv:1706.02633*, 2017.

W. A. Fuller. *Introduction to statistical time series*. John Wiley & Sons, 2009.

J. Gao and P.-N. Tan. Converting output scores from outlier detection algorithms into probability estimates. In *Sixth International Conference on Data Mining (ICDM'06)*, pages 212–221. IEEE, 2006.

J. Gao, X. Song, Q. Wen, P. Wang, L. Sun, and H. Xu. Robusttad: Robust time series anomaly detection via decomposition and convolutional neural networks. *arXiv preprint arXiv:2002.09545*, 2020.

I. Goodfellow, Y. Bengio, and A. Courville. *Deep learning*. MIT press, 2016.

N. Görnitz, M. Kloft, K. Rieck, and U. Brefeld. Toward supervised anomaly detection. *Journal of Artificial Intelligence Research*, 46:235–262, 2013.

N. Halko, P.-G. Martinsson, and J. A. Tropp. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM review*, 53(2):217–288, 2011.

D. J. Hand. Principles of data mining. *Drug safety*, 30(7):621–622, 2007.

S. Hariri, M. C. Kind, and R. J. Brunner. Extended isolation forest. *IEEE Transactions on Knowledge and Data Engineering*, 33(4):1479–1489, 2019.

D. Hawkins. Identification of outliers. springer, 1980.

P. Henry-Labordere. optimal transport and anomaly detection with neural networks: A primal-dual algorithm. *Available at SSRN*, 3370910, 2019.

A. Hinneburg. A density based algorithm for discovering clusters in large spatial databases with noise. In *KDD Conference, 1996*, 1996.

R. J. Hyndman, E. Wang, and N. Laptev. Large-scale unusual time series detection. In *2015 IEEE international conference on data mining workshop (ICDMW)*, pages 1616–1619. IEEE, 2015.

I. T. Jolliffe. Principal component analysis. *Technometrics*, 45(3):276, 2003.

D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

A. Kondratyev, C. Schwarz, and B. Horvath. Data anonymisation, outlier detection and fighting overfitting with Restricted Boltzmann Machines. *Outlier Detection and Fighting Overfitting with Restricted Boltzmann Machines (January 27, 2020)*, 2020.

H.-P. Kriegel, M. Schubert, and A. Zimek. Angle-based outlier detection in high-dimensional data. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 444–452, 2008.

M. N. Kurt, Y. Yılmaz, and X. Wang. Real-time nonparametric anomaly detection in high-dimensional settings. *IEEE transactions on pattern analysis and machine intelligence*, 43(7): 2463–2479, 2020.

D. Kwon, H. Kim, J. Kim, S. C. Suh, I. Kim, and K. J. Kim. A survey of deep learning-based network anomaly detection. *Cluster Computing*, 22(1):949–961, 2019.

N. Laptev, S. Amizadeh, and I. Flint. Generic and scalable framework for automated time-series anomaly detection. In *Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1939–1947, 2015.

A. Le Guennec, S. Malinowski, and R. Tavenard. Data augmentation for time series classification using convolutional neural networks. In *ECML/PKDD workshop on advanced analytics and learning on temporal data*, 2016.

M. Linting, J. J. Meulman, P. J. Groenen, and A. J. van der Koojj. Nonlinear principal components analysis: introduction and application. *Psychological methods*, 12(3):336, 2007.

H. Lu, Y. Xu, M. Ye, K. Yan, Z. Gao, and Q. Jin. Learning misclassification costs for imbalanced classification on gene expression data. *BMC bioinformatics*, 20(25):1–10, 2019.

A. Munawar, P. Vinayavekhin, and G. De Magistris. Limiting the reconstruction capability of generative neural network using negative learning. In *2017 IEEE 27th International Workshop on Machine Learning for Signal Processing (MLSP)*, pages 1–6. IEEE, 2017.

R. Mushtaq. Augmented dickey fuller test. *SSRN Electron. J.*, 2011.

C. Rijsbergen. Information retrieval 2nd ed buttersworth. *London*, 1979.

H. Ringberg, A. Soule, J. Rexford, and C. Diot. Sensitivity of PCA for traffic anomaly detection. *ACM SIGMETRICS Performance Evaluation Review*, 35(1):109–120, 2007.

L. Ruff, R. A. Vandermeulen, N. Görnitz, A. Binder, E. Müller, K.-R. Müller, and M. Kloft. Deep semi-supervised anomaly detection. *arXiv preprint arXiv:1906.02694*, 2019.

B. N. Saha, N. Ray, and H. Zhang. Snake validation: A PCA-based outlier detection method. *IEEE signal processing letters*, 16(6):549–552, 2009.

R. Sahoo, S. Zhao, A. Chen, and S. Ermon. Reliable Dcisions with Treshold Clibration. *Advances in Neural Information Processing Systems*, 34, 2021.

T. Saito and M. Rehmsmeier. The precision-recall plot is more informative than the roc plot when evaluating binary classifiers on imbalanced datasets. *PloS one*, 10(3):e0118432, 2015.

E. Schubert, J. Sander, M. Ester, H. P. Kriegel, and X. Xu. Dbscan revisited, revisited: why and how you should (still) use dbscan. *ACM Transactions on Database Systems (TODS)*, 42(3): 1–21, 2017.

R. Sedman. Online Outlier Detection in Financial Time Series, 2018.

M.-L. Shyu, S.-C. Chen, K. Sarinnapakorn, and L. Chang. Principal component-based anomaly detection scheme. In *Foundations and novel approaches in data mining*, pages 311–329. Springer, 2006.

T. T. Um, F. M. Pfister, D. Pichler, S. Endo, M. Lang, S. Hirche, U. Fietzek, and D. Kulić. Data augmentation of wearable sensor data for parkinson's disease monitoring using convolutional neural networks. In *Proceedings of the 19th ACM international conference on multimodal interaction*, pages 216–220, 2017.

S. Węglarczyk. Kernel density estimation and its application. In *ITM Web of Conferences*, volume 23. EDP Sciences, 2018.

Q. Wen, L. Sun, F. Yang, X. Song, J. Gao, X. Wang, and H. Xu. Time series data augmentation for deep learning: A survey. *arXiv preprint arXiv:2002.12478*, 2020.

Y. Yu, Y. Zhu, S. Li, and D. Wan. Time series outlier detection based on sliding window prediction. *Mathematical problems in Engineering*, 2014, 2014.

J. Zhang and I. C. Paschalidis. Statistical anomaly detection via composite hypothesis testing for markov models. *arXiv preprint arXiv:1702.08435*, 2017.

J. Zhang, J. Erway, X. Hu, Q. Zhang, and R. Plemmons. Randomized SVD methods in hyperspectral imaging. *Journal of Electrical and Computer Engineering*, 2012, 2012.

Y. Zhao and M. K. Hryniewicki. Xgbod: improving supervised outlier detection with unsupervised representation learning. In *2018 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2018.