Short Paper

Frequent Component Analysis for Large Time Series Databases with Gaussian Processes

Jan David Hüwel
Department of Mathematics and Computer Science,
University in Hagen
Hagen, Germany
jan.huewel@fernuni-hagen.de

Christian Beecks
Department of Mathematics and Computer Science,
University in Hagen
Hagen, Germany
christian.beecks@fernuni-hagen.de

ABSTRACT

Time series naturally comprise multiple inherent patterns of behavior that create the information we perceive. While these patterns frequently serve as a starting point for the analysis of large time series databases, they often overwhelm data scientists due to their volume and complexity. Extracting the most prevalent of these patterns thus still remains a challenge in many analytical methods. In this short paper, we propose an efficient approach to determine the most frequent inherent patterns of common behavior in large time series databases. To this end, we formally model time series by means of Gaussian processes and show how to find the most representative components corresponding to local time series patterns by introducing the CATGP (Component Analysis in Time series with Gaussian Processes) algorithm. We examine our proposal on different benchmark times series databases and show that the CATGP algorithm efficiently discovers representative components in the underlying time series databases.

1 INTRODUCTION

Time series epitomize a complex data type which appears across numerous application domains including industry, medicine, meteorology, economics or marketing, to name just a few. This data type comprises time-dependent measurements, such as temperature or air pressure measurements [4], stock prices [18] or quality control indicators [3] and is frequently encountered in combination with physical or virtual sensory. In this way, the observed measurements captured by means of time series provide an approximation of real-world processes which are typically unknown from the database perspective. In practice, these processes are often the result of several phenomena appearing and interacting simultaneously. For instance, meteorological temperature measurements contain daily and yearly periodicity, multiple linear increases and decreases as well as additional noise. Similarly, measurements from the manufacturing domain might contain specific patterns corresponding to the state of a machine indicating its oscillation or deterioration. In general, such patterns reflect multiple underlying process components which are intuitively to grasp for a human but difficult to analyze for a machine.

As indicated above, these process components correspond to real world phenomena and are thus difficult to infer from time series databases without further expert and domain knowledge. For example, two additive processes with logarithmic increases would result in a time series that contains one logarithmic trend. To separate this trend into the actual process components, prior domain knowledge is required. To streamline the user's input process and reduce the need for an exhaustive specification of domain knowledge, we assume that the measurable time series components adequately represent the underlying process components. In this way, our proposal becomes universally applicable across diverse domains, eliminating the necessity for pre-existing domain-specific knowledge.

In order to generate insights into time series databases, one might want to examine the correlation between multiple components by finding frequently appearing combinations, inferring potential rules and identifying anomalous behavior. This paper will present the necessary methodology to enable such examinations.

A necessary prerequisite is the extraction of components hidden in time series databases. Automatically extracting such time series components still poses a difficult problem, where some approaches have been proposed in the past: Statistical time series decomposition is for instance used in economics to separate the data into seasonality, trend and noise [19]. While this approach befits the intended use-case of analysing sales data, it's difficult to use it for inferring other types of components, such as behavior that follows a known differential equation [8].

As a domain-agnostic alternative, the Automatic Bayesian Covariance Discovery (ABCD) [17] method uses Gaussian processs (GPs) to construct a probabilistic model fitting the data's behavior that can be translated into natural language. This process will be described in more detail in Section 3. The possibility to include prior knowledge in the GP model via its kernel ensures that even complex components can be modelled [8]. While this approach is already a powerful tool in the right setting, it only delivers descriptions for individual time series and doesn't directly facilitate the comparison of multiple time series. Here, we propose Component Analysis in Time series with Gaussian Processes (CATGP), a method to expand this form of analysis by comparing the components of multiple time series and finding frequent components and component combinations.

To do so, we use the principle behind ABCD to link time series components to GP kernel components and treat these kernel components as items in frequent item set mining. The concept has been introduced before, using trees [13]. In this paper, we want to define a precise mathematical basis for both the extraction of components from time series and the analysis of said components. We define the necessary operations and relations in the kernel space in Section 4. Overall, our approach brings the problem of extracting components and identifying frequent combinations from the data space into the GP model space, which brings us additional flexibility and interpretability from GP kernels. This process is depicted in Figure 1.

The rest of this paper is structured as follows: Section 2 presents relevant works on time series component analysis and frequent item set mining. In Section 3, we introduce the necessary basics of GPs and their kernels. In Section 4, we introduce our method

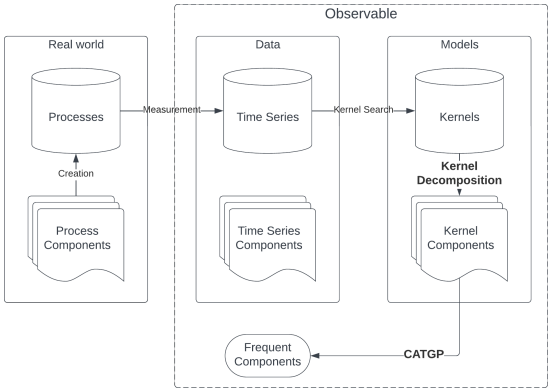
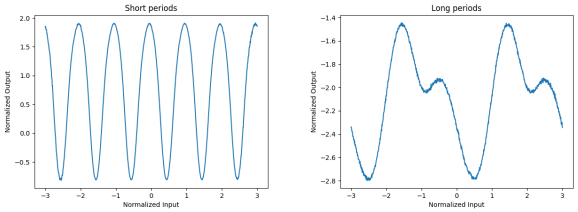


Figure 1: The process of CATGP. The goal is to identify correlations between time series components, as they are the best indicators for correlations between process components. For this examination, we model the data using a GP kernel search and perform the analysis on the resulting models. The separation of models into components and the subsequent frequent component mining are described in Section 4



(a) Low period length **(b) High period length**

Figure 2: Differences in GP models with the periodic kernel [9], using different parameters for the period length. The samples shown here are priors without any data.

CATGP, before we discuss the initial results in Section 5. Finally, Section 6 concludes this work with an outlook on future work.

2 RELATED WORK

Our proposed approach connects two different fields of analysis: The division of time series data into separate, interpretable components and the mining of frequent items in a transactional database.

Analysing the components of a time series is a common problem in areas such as economics and physics. Different domains developed different specialised solutions for the extraction of these components. In economics, the data to be analyzed is usually a singular time series depicting, for example, sales over time. Statistical analysis can be used to look for seasonal and linear trends [19]. However, this method can not detect components that have not been defined by the user. We are also not aware of any works that perform this method for components beyond seasonality, trend and noise.

In settings with physical sensors, users can often use multiple detectors to record data, resulting in multiple time series that contain identical components. A classic example are guests at a party, whose voices are recorded by multiple microphones. To isolate individual components, the Independent Component Analysis (ICA) [16] performs independence tests via high order statistical moments. This method requires the components to be present in multiple time series.

Neither of these methods facilitates the extraction of complex structural components from multiple independent time series for further comparisons. This type of analysis is mainly possible with GPs and the corresponding kernel search methods [7, 10, 14, 17]. Section 3 will briefly touch on some of these methods.

The field of frequent item set mining emerged with the Apriori algorithm [2]. This method identifies frequent items and combinations of items in a given database, where frequency is defined by a minimum rate of occurrence and items can be any categorical data. Later advances in this field, such as the FP-Growth algorithm [12], offer increased efficiency.

If the ordering of items is important for the use case, sequential pattern mining can be used to search for frequent and interesting sequences of items [1, 11]. This field is still developing today, as the steady increase in available data requires more and more efficient tools.

All of these methods require the database entries to be sets of items, where similarities between items are simply defined by the trivial metric, meaning items are either identical or unrelated. The goal of this paper is therefore, to define the necessary operations and relations on the infinite-dimensional space of GP kernels to apply item set mining on them. These definitions are derived in Section 4.

3 GAUSSIAN PROCESSES

In this section, we provide a short overview of Gaussian Processes.

Inferring the underlying process from recorded data points is a challenge that is still subject to current research. This includes inferring unknown data points via interpolation, forecasting to not yet recorded data or fitting a modelling function onto the known data.

GPs are among the most commonly used types of models for these purposes. They are very proficient at fitting a distribution to the given data points and allow for the incorporation of prior knowledge into the modelling process.

GPs are probabilistic, non-parametric machine learning frameworks [21]. A model $GP(m, k)$ depends on a mean function $m : \mathbb{R} \to \mathbb{R}$ and a covariance function, also called kernel, $k : \mathbb{R} \times \mathbb{R} \times \Theta \to \mathbb{R}$ to infer an unknown function from data samples. The space Θ contains the kernel's parameters and depends on the specific kernel in use. An exemplary showcase of the effect of these parameters on the kernel's behavior is shown in Figure 2. In many cases, the kernel contains all necessary information and the mean function can be set to zero [21]. As such, this work also only regards zero-mean GPs.

The kernel describes, how the modelled data depend on each other. For example, there are popular kernels that represent periodic or linear behavior within the data [9]. Such kernels can also be added or multiplied to create more complex and more descriptive kernels [10]. The quality of predictions or interpolations using a GP directly depends on how well the model's kernel describes the data's behavior. Consequently, optimal model selection would require extensive prior knowledge about the underlying process that the data represent. In settings where

this knowledge is not given, one can either use general-purpose kernels or apply an automatic kernel selection algorithm, that constructs a fitting kernel for the given data [6, 7, 10, 14, 17].

Recently, there has been more research into kernel search methods that split the time series at change points and find segment-based "local" kernels. Algorithms like the Concatenated Composite Covariance Search [7] or Event-Triggered Kernel Adjustments [15] can be used to segment and model long time series including multiple change points. In these settings, analyzing the resulting local models or generalizing to a global model requires comparisons between the kernels. In addition, it has been shown that kernel search methods can be employed to create natural-language descriptions of a time series [17]. This uses the property that the product of kernels is equivalent to inter-dependent structures, such as periodic patterns with linearly rising amplitudes, while the sum of kernels can be equated to independent processes that run in parallel to create the data.

In this paper, we consider products of kernels to represent statistically dependent behavior and sum of kernels to represent statistically independent behavior, cf. [7]. In this way, kernel products represent inseparable patterns which are the smallest units defining a kernel that we refer to as components.

In the next section, we will introduce a method to find the most commonly-occurring components.

4 PROPOSED METHOD

In this section, we introduce the CATGP method for finding frequent time series components modelled via Gaussian Processes. For this purpose, we focus on kernel components, which are inherently encountered in GPs. As a result, the problem of finding the most representative components in a time series is attributed to the ability of GPs to infer an interpretable kernel for the given data [9, 10, 17].

Prior to the application of CATGP, we make use of a kernel search method [6, 7, 10, 14, 17] to find the best fitting GP kernel for each time series in the database. These kernels are parameterized functions, that are constructed by adding or multiplying several base kernels together (cf. Section 3) and can therefore become arbitrarily complicated. Our analysis is primarily concentrated on the types of patterns that a kernel corresponds to, independent of the parameters' initialization. Therefore, we categorize the kernels that only differ in parameterization into equivalence classes.

Definition 4.1 (Kernel equivalence class). The equivalence class $[k]$ of a kernel k is defined as the class of all kernels that only differ from k by values of their parameters:

$$[k] := \{k(\cdot, \cdot, \theta) : \theta \in \Theta\}$$

Furthermore, we define the sum and product of two classes as

$$[k_1] + [k_2] := [k_1 + k_2] = \{\hat{k}_1 + \hat{k}_2 : \hat{k}_1 \in [k_1], \hat{k}_2 \in [k_2]\}$$

$$[k_1] \times [k_2] := [k_1 \times k_2] = \{\hat{k}_1 \times \hat{k}_2 : \hat{k}_1 \in [k_1], \hat{k}_2 \in [k_2]\}$$

Consequently, a kernel class represents a structure on an abstract level, so that differences in amplitude, steepness, frequency and other properties don't affect the class label. Since all fundamental kernels that we use have a scaling parameter as a prefactor and no kernel is constant, this definition implies the following properties:

$$[k_1], [k_2] \subseteq [k_1 + k_2]$$

$$[k_1], [k_2] \nsubseteq [k_1 \times k_2]$$

These properties align with a very intuitive definition of a class's components: Since products of kernels correspond to inter-dependency [17], a product isn't separable for a component analysis. The set of all possible components for kernels can therefore be defined as follows.

Definition 4.2 (Set of possible components). Given a set of base kernel classes $\mathbb{B}$, we define the set of all possible component classes as $\mathbb{K} = \{\Pi_{i=1}^{n} [b_i] : n \in \mathbb{N}, [b_i] \in \mathbb{B}\}$

This set contains everything that we would consider components in kernels. It consequently corresponds to the potential time series components that we are able to identify with our approach. Every given GP kernel's class is a sum of (usually up to three) set elements. This makes the decomposition of such a class straight forward.

Definition 4.3 (Component decomposition). The set of components that make up a class $[k]$ is given by the multiset

$$C([k]) = (A, m)$$

where

$$A = \{[c] \in \mathbb{K} : [c] \subseteq [k]\} \text{ and}$$

$$m = c \mapsto max\left(\{n \in \mathbb{N} : \sum_{i=1}^{n} [c] \subseteq [k]\}\right).$$

We use a multiset instead of a regular set for this definition, so that in cases where one component is contained multiple times within the same kernel, these multiple occurrences are not ignored. Consequently, we can write each kernel class as the sum of its components:

$$(A, m) = C([k]) \Leftrightarrow [k] = \sum_{[c] \in A} \sum_{i=1}^{m([c])} [c]$$

Since our method of analysis is inspired by frequent item set mining, for example with the Apriori algorithm [2], we now require three more definitions before we formulate the CATGP algorithm: The length and support of a class and the join of two classes. Using the component decomposition, we can intuitively define the length of a class, or even an initialized kernel, in the following way:

Definition 4.4 (Length). A kernel k and its equivalence class $[k]$ have the following length:

$$\mathcal{L}(k) := \mathcal{L}([k]) = |C([k])|.$$

Our final goal regarding the GP components is to define, what constitutes as a "frequent" component and to identify all components in a given set of kernels, that match this definition. Consequently, we define the support of any class as the amount of given kernels that the class is contained in.

Definition 4.5 (Support). The support of a class $[c]$ given a sequence of kernels $K = (k_1, ..., k_N)$ is

$$supp([c]) = |\{i \in (1, ..., N) : [c] \subseteq [k_i]\}|$$

Note that this definition implies that multiple occurrences of a component in one kernel do not compensate the non-occurrence in other kernels.

The final required definition is an operator that joins two classes to their smallest common superset, that is also a viable kernel class.

Definition 4.6 (Join). The join of two classes $[k_1]$ and $[k_2]$ is equivalent to the join definition for item sets in the Apriori algorithm: the result is the smallest possible item set (here kernel class) that contains both the given sets.

$$[k_1] \sqcup [k_2] := min\left(\{[k] \in \mathbb{K} : [k_1], [k_2] \subseteq [k]\}\right)$$

We now have the necessary mathematical language to formulate our algorithm in detail (see Alg. 1). The CATGP algorithm takes a sequence of GP kernels and a minimum threshold for the support of components and delivers all component classes and sums thereof whose support exceeds the given threshold. To determine these classes, we first extract the fundamental components of the given kernels by using the kernel decomposition (Def. 4.3). The components with sufficient support are added to the output set and then used to create potentially frequent classes of length 2. Here, the Apriori principle is applied, since any class containing a non-frequent subclass can not be frequent itself [2]. The steps of calculating the support for the candidates and then creating the next generation of candidates is repeated until no further potentially frequent classes can be found.

Algorithm 1: CATGP

Data: GP models $(k_1, ..., k_N)$, minimum support *minSupp*
Result: Frequent kernel component classes $\{[c_1], ..., [c_n]\}$

1-element candidates $C_1 = \bigcup_{i=1}^{N} \mathcal{C}([k_i])$
$i = 1$
Output $out = \{\}$
while $C_i \neq$ **do**
  Frequent sets $F_i = \{\}$
  for $[c] \in C_i$ **do**
    if $supp([c]) \geq minSupp$ **then**
      $out = out \cup \{[c]\}$
      $F_i = F_i \cup \{[c]\}$
  $C_{i+1} = \{\}$
  for $[c_1], [c_2] \in F_i$ **do**
    if $\mathcal{L}([c_1] \cap [c_2]) == i - 1$ **then**
      $[c] := [c_1] \sqcup [c_2]$
      if $\forall [c'] \subseteq [c] : \mathcal{L}([c']) == i - 1 \Rightarrow [c'] \in F_i$ **then**
        $C_{i+1} = C_{i+1} \cup \{[c]\}$
  $i = i + 1$
return out

CATGP's complexity is in $O\left(N \sum_i |C_i|\right)$, which is derived directly from Apriori [2]. The sum potentially scales exponentially with the number of unique component classes in the database. This is limited by the amount of base kernels $|\mathcal{B}|$ and the maximum complexity of a kernel, which is a parameter of the kernel search [10]. For most applications, these values will be rather low. For settings with a high variety of base kernels or a high kernel complexity, higher efficiency can be achieved when the presented operations and notations are used for a different frequent item set mining algorithm, like FP-Growth [12].

5 EXPERIMENTS

In this section, we present the functionality of the CATGP algorithm in two experiments. First, we analyze the *Plane* database [5], which contains time series depicting the outlines of different plane types, with two different configurations to show CATGP's ability to generate insights. Afterwards, we generate a GP database from a singular time series to demonstrate an alternative application of CATGP.

In our first analysis, we focus on showing that CATGP's results allow the user to derive insights into the given GP models and the modelling process. To do so, we first create GP models for all time series in the training set of *Plane*. This database contains 7 classes of plane types, which we analyze separately with CATGP. The results are shown in Tables 1 and 2. The chosen minimum support is set to 10%, but we chose to only list the two most frequent components for readability.

Class	1. component	2. component
1	RQ (53%)	PER (46%)
2	RQ (50%)	$RQ \times RQ$ (50%)
3	RQ (55%)	PER (22%)
4	$PER \times RQ$ (50%)	$RQ \times RQ$ (25%)
5	$RQ \times RQ \times RQ \times RQ$ (23%)	$RQ \times RQ \times RQ$ (23%)
6	$LIN \times PER$ (27%)	$PER \times RQ$ (27%)
7	$PER \times RQ$ (35%)	PER (30%)

Table 1: Experiment 1: Most frequent components per class in the *Plane* training set [5]. The GP models were created using Compositional Kernel Search (CKS) [10] with the base set of LIN, PER, RQ, describing the linear, periodic and rational quadratic kernel respectively [9]. CKS was run for 4 iterations. The support of the components is denoted as relative support within the given class.

Class	1. component	2. component
1	$PER \times PER$ (60%)	PER (33%)
2	$PER \times PER$ (92%)	$LIN \times PER$ (50%)
3	$PER \times PER$ (88%)	$PER \times PER + PER \times PER$ (44%)
4	$PER \times PER$ (81%)	$LIN \times PER$ (50%)
5	$PER \times PER$ (76%)	$PER \times PER \times PER$ (23%)
6	$PER \times PER$ (83%)	$PER \times PER + PER \times PER$ (33%)
7	$PER \times PER$ (65%)	$LIN \times PER$ (50%)

Table 2: The same experiment as Table 1, but the models were created from a smaller base set of PER, LIN, denoting the periodic and linear kernel respectively, and with 3 iterations of CKS.

Table 1 shows some clear differences between the classes when using the three mentioned base kernels. Especially class 5 seems to favor highly adaptable models, like products of multiple rational quadratic kernels. In general, the rational quadratic kernel is very prominent, which is probably because it is the least specialized of the base kernels. The linear kernel is the most specialized and only appears in class 6 commonly. The variety of results suggests that the minimum support was chosen too low, or that the amount of instances per class was not sufficient to generate valuable insights into every class.

Table 2 shows another very interesting phenomenon. By removing the rational quadratic kernel from the set of base kernels, we limit the model selection's possibilities to construct highly adaptable kernels. As a result, all classes contain the product of two periodic kernels most frequently. The time series are all

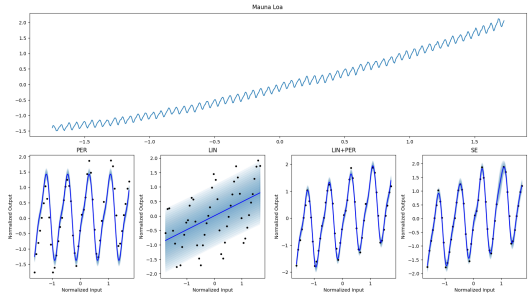


Figure 3: Analysis of the *Mauna Loa* time series data set [17, 20] using CATGP. The data is shown on top and the four most common components or sums of components, applied to arbitrary data segments, are shown on the bottom, ordered from most frequent to least. The exact support was 86%**,** 63%**,** 59% **and** 45% **respectively.**

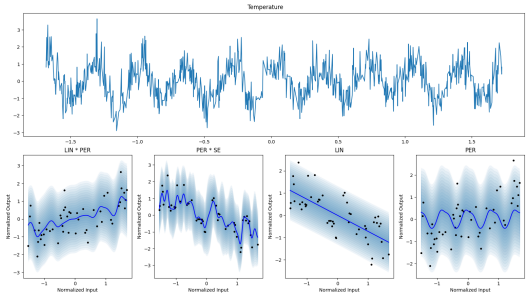


Figure 4: Analysis of the *Termperature* time series data set using CATGP. The data is shown on top and the four most common components or sums of components, applied to arbitrary data segments, are shown on the bottom, ordered from most frequent to least. The exact support was 40%**,** 25%**,** 21% **and** 18% **respectively.**

symmetrical on the Y-axis, which is why a periodic component is to be expected. The reason for why this didn't show up in the previous experiment is, that our model selection uses the GP likelihood for evaluation, which favors adaptable kernels.

In an alternative method of application, CATGP can be used to analyze patterns and components within a singular time series. To do so, we can extract subsequences as overlapping windows of equal length from the time series, perform a GP model selection on each subsequence and define our model database as the resulting GP models. Here, we demonstrate this with two different real-world data sets.

The subsequences for our experiments have a length of 50 and we use a stepsize of 30, so the first subsequence is from index 1 to 50, the second from 31 to 80 etc. All subsequences are Z-scaled to improve the GP performance [21]. The GP models are then created by applying the CKS algorithm with 5 iterations to the subsequences, which means that our kernels are made up of up to 5 base kernels. The set of base kernels contains the linear, periodic and squared-exponential kernel, denoted as *LIN*, *PER*, *SE* respectively. We set a minimum support of 10% for our experiments. With the subsequence sampling, the results of CATGP can be interpreted as frequencies and correlations between local patterns of behavior.

We are not aware of any existing methods that determine correlations of time series components, so we visually determined realistic components in the data sets and compared CATGP's results to these expectations.

The first iteration of this experiment analyzes the *Mauna Loa* time series data set [17, 20]. It shows a series of CO_2 measurements at the Mauna Loa weather station over a long period of time. Naturally, the most important components are periodic and linear trends. The results of the analysis are shown in Figure 3. CATGP has successfully identified periodic and linear trends with 86% and 63% support respectively. The combined structure of periodic and linear behavior has a support of 59%. Finally, the squared-exponential kernel, which is very adaptable and therefore often favored in the kernel selection, was the fourth most frequent component with a support of 45%.

While the support of the periodic and linear components ideally would have been higher, CATGP was still able to identify the most important features of the data set. It is also important to mention, that a GP kernel search is not a deterministic process and can occasionally result in non-optimal models.

Repeating the experiment with the *Temperature* time series data set [17] yields the results presented in Figure 4. Given the much noisier and broadly periodic nature of the data, we expect high frequencies for the periodic kernel and the squared-exponential kernel. The results only roughly match this expectation: The two most frequent components are products of base kernels, which can adapt to noisy data more effectively than single base kernels because they have more parameters. Even so, the overall support for the most frequent components is very low, showing more variance between the local models. This data set has 1000 data points, so the subsequences of length 50 don't contain enough data to show the primary periodic behavior. Instead, linear kernels are used to model the local ascend or descend. These results show that CATGP requires careful consideration of how to create the local kernels.

These experiments show, that the CATGP algorithm can successfully extract frequent components from a given set of GP models. It also became clear that the insights into a singular time series that CATGP generates via slicing heavily depend on the selection of subsequences. The code for these experiments can be found here: https://github.com/JanHuewel/CATGP/

6 CONCLUSIONS AND FUTURE WORK

In this paper, we have proposed an algorithmic approach for finding the most representative components from time series data. For this purpose, we have leveraged Gaussian processes and proposed the CATGP algorithm for determining frequent kernel component classes. The experimental evaluation indicates a great potential for CATGP to advance the field of time series analysis, but also possible causes of failure that will need to be examined further.

There are some open challenges, which are yet to be researched further: The choice for the minimum support has a great impact on the results, but valid values depend heavily on the data and the setting in question. For now, CATGP can easily be modified to find the most frequent sets instead, without a minimum threshold, at the cost of some efficiency. Additionally, the set of base kernel classes needs to be defined by the user, which restricts the methods ability to find unexpected patterns. However, this can also be used to incorporate prior knowledge about the origin of the data into the search.

As for future work, we plan to expand CATGP to incorporate the kernels' parameters into class definitions. This enables

users to differentiate between low-frequency and high-frequency periods, linear increases and decreases or noisy and noise-free segments. Furthermore, we will examine the ability of CATGP to find association rules between components and their meaning for large time series databases. Finally, the process of selecting subsequences for local GP models will be improved and directly included into the CATGP algorithm.

ACKNOWLEDGMENTS

This research was supported by the research training group "Dataninja" (Trustworthy AI for Seamless Problem Solving: Next Generation Intelligence Joins Robust Data Analysis) funded by the German federal state of North Rhine-Westphalia.

REFERENCES

[1] R. Agrawal and R. Srikant. 1995. Mining sequential patterns. In *Proceedings of the Eleventh International Conference on Data Engineering*. 3–14. https://doi.org/10.1109/ICDE.1995.380415

[2] Rakesh Agrawal, Ramakrishnan Srikant, et al. 1994. Fast algorithms for mining association rules. In *Proc. 20th int. conf. very large data bases, VLDB*, Vol. 1215. Santiago, Chile, 487–499.

[3] Layth C Alwan and Harry V Roberts. 1988. Time-series modeling for statistical process control. *Journal of business & economic statistics* 6, 1 (1988), 87–95.

[4] Ingeborg Auer, Reinhard Böhm, Anita Jurkovic, Wolfgang Lipa, Alexander Orlik, Roland Potzmann, Wolfgang Schöner, Markus Ungersböck, Christoph Matulla, Keith Briffa, et al. 2007. HISTALP—historical instrumental climatological surface time series of the Greater Alpine Region. *International Journal of Climatology: A Journal of the Royal Meteorological Society* 27, 1 (2007), 17–46.

[5] Anthony Bagnall, Jason Lines, Aaron Bostrom, James Large, and Eamonn Keogh. 2017. The great time series classification bake off: a review and experimental evaluation of recent algorithmic advances. *Data mining and knowledge discovery* 31 (2017), 606–660.

[6] Fabian Berns, Jan Hüwel, and Christian Beecks. 2022. Automated Model Inference for Gaussian Processes: An Overview of State-of-the-Art Methods and Algorithms. *SN Computer Science* 3, 4 (2022), 300.

[7] Fabian Berns, Kjeld Schmidt, Ingolf Bracht, and Christian Beecks. 2021. 3cs algorithm for efficient gaussian process model retrieval. In *2020 25th International Conference on Pattern Recognition (ICPR)*. IEEE, 1773–1780.

[8] Andreas Besginow and Markus Lange-Hegermann. 2022. Constraining Gaussian Processes to Systems of Linear Ordinary Differential Equations. In *Advances in Neural Information Processing Systems*, S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (Eds.), Vol. 35. Curran Associates, Inc., 29386–29399. https://proceedings.neurips.cc/paper_files/paper/2022/file/bcef27c5825d1ed8757290f237b2d851-Paper-Conference.pdf

[9] David Duvenaud. 2014. *Automatic model construction with Gaussian processes*. Ph.D. Dissertation. University of Cambridge.

[10] David Duvenaud, James Lloyd, Roger Grosse, Joshua Tenenbaum, and Ghahramani Zoubin. 2013. Structure discovery in nonparametric regression through compositional kernel search. In *International Conference on Machine Learning*. PMLR, 1166–1174.

[11] Philippe Fournier-Viger, Jerry Chun-Wei Lin, Bay Vo, Tin Truong Chi, Ji Zhang, and Hoai Bac Le. 2017. A survey of itemset mining. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* 7, 4 (2017), e1207.

[12] Jiawei Han, Jian Pei, and Yiwen Yin. 2000. Mining frequent patterns without candidate generation. *ACM sigmod record* 29, 2 (2000), 1–12.

[13] Jan David Hüwel and Christian Beecks. 2023. Gaussian Process Component Mining with the Apriori Algorithm. In *International Conference on Database and Expert Systems Applications*. Springer, 423–429.

[14] Jan David Hüwel, Fabian Berns, and Christian Beecks. 2021. Automated kernel search for gaussian processes on data streams. In *2021 IEEE International Conference on Big Data (Big Data)*. IEEE, 3584–3588.

[15] Jan David Hüwel, Florian Haselbeck, Dominik G Grimm, and Christian Beecks. 2022. Dynamically Self-adjusting Gaussian Processes for Data Stream Modelling. In *KI 2022: Advances in Artificial Intelligence: 45th German Conference on AI, Trier, Germany, September 19–23, 2022, Proceedings*. Springer, 96–114.

[16] Christian Jutten and Jeanny Herault. 1991. Blind separation of sources, part I: An adaptive algorithm based on neuromimetic architecture. *Signal processing* 24, 1 (1991), 1–10.

[17] James Robert Lloyd, David Duvenaud, Roger B. Grosse, Joshua B. Tenenbaum, and Zoubin Ghahramani. 2014. Automatic Construction and Natural-Language Description of Nonparametric Regression Models. In *AAAI*. AAAI Press, 1242–1250.

[18] Prapanna Mondal, Labani Shit, and Saptarsi Goswami. 2014. Study of effectiveness of time series modeling (ARIMA) in forecasting stock prices. *International Journal of Computer Science, Engineering and Applications* 4, 2 (2014), 13.

[19] Julius Shiskin and Harry Eisenpress. 1957. Seasonal adjustments by electronic computer methods. *J. Amer. Statist. Assoc.* 52, 280 (1957), 415–449.

[20] Kirk W Thoning, Pieter P Tans, and Walter D Komhyr. 1989. Atmospheric carbon dioxide at Mauna Loa Observatory: 2. Analysis of the NOAA GMCC data, 1974–1985. *Journal of Geophysical Research: Atmospheres* 94, D6 (1989), 8549–8565.

[21] Christopher KI Williams and Carl Edward Rasmussen. 2006. *Gaussian processes for machine learning*. Vol. 2. MIT press Cambridge, MA.