Datenbank Spektrum (2021) 21:225–236
https://doi.org/10.1007/s13222-021-00389-5

FACHBEITRAG

Season- and Trend-aware Symbolic Approximation for Accurate and Efficient Time Series Matching

Lars Kegel[1] · Claudio Hartmann[1] · Maik Thiele[1] · Wolfgang Lehner[1]

Received: 1 June 2021 / Accepted: 7 September 2021 / Published online: 6 October 2021

Abstract
Processing and analyzing time series datasets have become a central issue in many domains requiring data management systems to support time series as a native data type. A core access primitive of time series is matching, which requires efficient algorithms on-top of appropriate representations like the symbolic aggregate approximation (SAX) representing the current state of the art. This technique reduces a time series to a low-dimensional space by segmenting it and discretizing each segment into a small symbolic alphabet. Unfortunately, SAX ignores the deterministic behavior of time series such as cyclical repeating patterns or a trend component affecting all segments, which may lead to a sub-optimal representation accuracy. We therefore introduce a novel season- and a trend-aware symbolic approximation and demonstrate an improved representation accuracy without increasing the memory footprint. Most importantly, our techniques also enable a more efficient time series matching by providing a match up to three orders of magnitude faster than SAX.

Keywords Time series databases · Time series analysis · SAX · PAA

1 Introduction

Time series are the prime data source for data-mining tasks in many domains requiring efficient and thus usually native support by an underlying data management platform [18]. In order to devise complex analytical scenarios, time series data types require a storage model, a query language, and optimization mechanisms. One of the most relevant access primitives in time series system is the retrieval of similar time series which is commonly referred to as *time series matching* [18]. Due to the high number of measure values, time series are considered a high dimensional data type. This makes the distance calculation required for matching very time and memory consuming. Therefore, time series are not directly matched against each other but represented and compared in a low-dimensional space allowing the approximation of their true (Euclidean) distance [1].

Based on this idea, researchers have been developing representation techniques for time series for the past three decades. Among these techniques, the symbolic aggregate approximation (SAX) from Lin et al. is of particular interest [5]: First, this technique segments a time series into intervals represented by their mean value, the so-called Piecewise Aggregate Approximation (PAA). Second, it discretizes each mean value by mapping it to a discrete symbol. Thus, SAX provides a small representation and a fast distance measure enabling it for time series matching. Moreover, its distance measure has the important property to lower-bound the Euclidean distance measure, i.e., it allows for pruning observations based on the representation, without the need to load all high-dimensional time series into memory and calculate their Euclidean distance.

However, SAX suffers from two shortcomings. First, it assumes that the PAA of a normalized time series is normally distributed with the same standard deviation, which is over-simplistic [2] and negatively impacts the representation accuracy. Second, SAX ignores the deterministic behavior of a time series like a *season*, i.e., a cyclical repeated behavior, or a *trend*, i.e., a long-term change in the mean level. For example, production or consumption time

✉ Lars Kegel
lars.kegel@tu-dresden.de

✉ Claudio Hartmann
claudio.hartmann@tu-dresden.de

✉ Maik Thiele
maik.thiele@tu-dresden.de

✉ Wolfgang Lehner
wolfgang.lehner@tu-dresden.de

[1] Dresden Database Research Group, TU Dresden, Dresden, Germany

series from the energy domain exhibit daily, weekly, or yearly seasons, while sales or price time series within the economy domain often exhibit an increasing or decreasing trend. SAX does not take these features into account with again negative consequences for the representation accuracy. While several SAX extensions including more features have been proposed in the literature, no approach considers global features before segmentation, i.e., features that arise from the deterministic behavior of a time series.

Therefore, this work focuses on representation techniques based on SAX that aim to solve these shortcomings and significantly improve time series matching.

In the following, we introduce the season- and trend-aware symbolic approximations, *sSAX* and *tSAX* natively considering then season and trend of recorded time series. Both techniques provide a *more accurate representation* with the same representation size as SAX allowing a more *accurate* and *efficient* time series matching. In summary, we make the following contributions:

- We start with an overview of SAX and review existing extensions. While all extensions provide lower-bounding distance measures, most of them increase the representation size (Sec. 2).
- We introduce sSAX and tSAX providing a higher matching accuracy with the same memory footprint as SAX (Sec. 3).
- We evaluate the techniques of time series matching. We summarize our experimental setting, present, and discuss our results (Sec. 4). The most remarkable result to emerge from this evaluation is that on large datasets (100 GB), sSAX returns exact matches up to three orders of magnitude faster than SAX.
- Finally, we summarize and provide insights into future work in Sec. 5.

2 State of the Art

In the following, we compare the original SAX with some relevant extensions by formally defining the terms *time series dataset* and *time series matching* along with its constraints listed in [5].

2.1 Preliminaries

Throughout the paper, we use the following definition of a time series. A *time series* $\underline{x}$ is a vector of values x which are measured at discrete time instances t:

$$\underline{x}^{\mathsf{T}} = (x_1, ..., x_t, ..., x_T) \text{ where } \underline{x} \in \mathbb{R}^T, t \in \mathbb{N}_{>0}. \quad (1)$$

A time series is therefore (1) finite with a fixed length T, (2) complete, i.e., without null values, and (3) equidistant, i.e., the distance between two time instances is constant. Moreover, (4) it is normalized, i.e., its values have a sample mean of zero and a sample variance of one.

The goal of time series matching is to retrieve the most similar series out of a *time series dataset* compared to a query time series. A time series dataset is a set of I time series with the same length: $X = \{\underline{x}_1, ..., \underline{x}_i, ..., \underline{x}_I\}$ where $i \in \mathbb{N}_{>0}, i \leq I$.
The most similar time series exhibits the lowest *Euclidean distance* to the query which is: $d_{\text{ED}}(\underline{x}, \underline{x}') = \sqrt{\sum_{t=1}^{T}(x_t - x_t')^2}$. Usually, time series are not directly compared using the Euclidean distance, since a time series dataset may be large (many individual values) and calculating the Euclidean distances would require substantial effort.

2.2 Original SAX

The original SAX [5] reduces the dimensionality of a time series (number of data points) in two steps: First, the *segmentation* into mean values by *piecewise aggregate approximation* (PAA) reduces the dimensionality in the time domain. Second, the *discretization* into symbols by *symbolic aggregate approximation* (SAX) reduces the dimensionality in the value domain. As such, PAA is a prerequisite of SAX, which is defined as follows.

Let $W \in \mathbb{N}_{>0}$ be the number of segments per time series, and W divides T. The PAA $\underline{\bar{x}}$ is the vector of mean values of a time series:

$$\underline{\bar{x}}^{\mathsf{T}} = (\bar{x}_1, ..., \bar{x}_w, ..., \bar{x}_W), \text{ where} \quad (2)$$

$$\bar{x}_w = \frac{W}{T} \sum_{t=\frac{T}{W}(w-1)+1}^{\frac{T}{W}w} x_t. \quad (3)$$

Although PAA reduces a time series in the time domain, it still contains real values taking a considerable amount of memory. Therefore, the time series is further reduced utilizing SAX.

Let A be the size of an alphabet ($A \in \mathbb{N}_{>0}$) and let $\underline{b}^{\mathsf{T}} = (b_1, ..., b_a, ..., b_{A-1})$ be a vector of increasingly sorted *breakpoints* that split the original domain into A intervals $]-\infty, b_1[, ..., [b_{a-1}, b_a[, ..., [b_{A-1}, \infty[$.

SAX $\underline{\hat{x}}$ is the vector of symbols, i.e. the mean values discretized into the alphabet A:

$$\underline{\hat{x}}^{\mathsf{T}} = (\hat{x}_1, \hat{x}_2, ..., \hat{x}_w, ..., \hat{x}_W), \text{ where} \quad (4)$$

$$\hat{x}_w = \begin{cases} 1 & -\infty < \bar{x}_w < b_1 \\ a & \exists a : b_{a-1} \leq \bar{x}_w < b_a \\ A & b_{A-1} \leq \bar{x}_w < \infty \end{cases}. \quad (5)$$

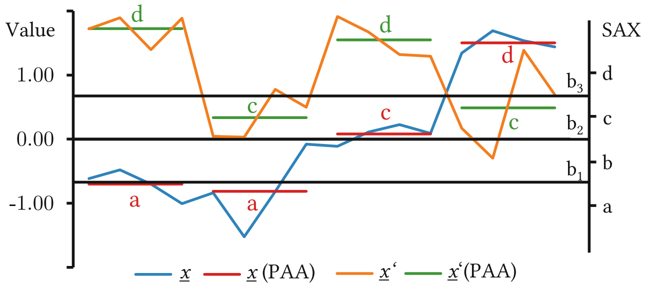

Fig. 1 Time series with PAA and SAX representations

SAX reduces each mean value to a discrete symbol usually visualized with alphabetic characters in order to stress their discrete nature. Ideally, the symbols of a dataset are equiprobable s.t. they make full use of the alphabet capacity. To achieve this, mean values should be $\mathcal{N}(0,1)$-distributed [5]. Consequently, breakpoints are set s.t. the area under the normal distribution $\mathcal{N}(0,1)$ from $[b_{a-1}, b_a[$ equals $1/A$.

Distance measures for PAA and SAX are defined as follows:

$$d_{\text{PAA}}(\bar{\underline{x}}, \bar{\underline{x}}') = \sqrt{T/W}\sqrt{\sum_{w=1}^{W}(\bar{x}_w - \bar{x}'_w)^2} \tag{6}$$

$$d_{\text{SAX}}(\widehat{\underline{x}}, \widehat{\underline{x}}') = \sqrt{T/W}\sqrt{\sum_{w=1}^{W}\text{cell}(\widehat{x}_w, \widehat{x}'_w)^2}, \text{ with} \tag{7}$$

$$\text{cell}(a, a') = \begin{cases} 0 & |a - a'| \leq 1 \\ b_{\max(a,a')} - b_{\min(a,a')+1} & \text{otherwise} \end{cases}. \tag{8}$$

Figure 1 shows an example time series $\underline{x}$ (blue line, $T = 16$) and its PAA representation (red segments, $W = 4$). Given an alphabet $A = 4$ and respective breakpoints at 0.00 and ± 0.67 (black horizontal lines), its SAX representation is $\widehat{\underline{x}}^{\mathsf{T}} = (a, a, c, d)$. The figure also shows a second time series $\underline{x}'$ (orange line) whose SAX representations is $\widehat{\underline{x}}'^{\mathsf{T}} = (d, c, d, c)$. The Euclidean distance between $\underline{x}$ and $\underline{x}'$ is approx. 6.71, the PAA distance is approx. 6.44, and the SAX distance is approx. 3.02.

2.3 Properties

Efficient time series matching exhibits five properties. Subsequently, we describe and review these properties regarding SAX.

a) *Representation Size:* The representation size of a time series should be small compared to its original size to allow efficient processing: SAX reduces the time series to W segments that are further reduced to symbols of an alphabet of size A resulting in a representation size of $W \cdot \text{ld}(A)$. In Fig. 1, the SAX symbols of $\widehat{\underline{x}}$ need $4 \cdot \text{ld}(4) = 8$ bits, which is small compared to the original time series $\underline{x}$ needing $16 \cdot 32 = 512$ bits, assuming an original floating-point value is stored with 32 bits.
b) *Representation Time:* The transformation of a time series into its representation should be fast: SAX involves one pass over the data, carrying out segmentation and discretization simultaneously. Thus, SAX allows for a fast transformation into a low-dimensional space. The time series have to be already normalized.
c) *Distance Storage:* The distance calculation should not incur a substantial storage overhead: SAX distance calculation involves the pairwise comparison of SAX symbols. Instead of frequently recalculating these distances, Lin et al. store each symbol combination in a lookup table of size $A^2 \cdot 32$ bits [5]. For a typical alphabet size of $A = 256$, the size of the lookup table is 262 kB, which is only calculated once for the dataset.
d) *Distance Time:* Time series matching can only benefit from a representation technique if the comparison between representations is faster than in the high-dimensional space: The SAX distance calculation involves one lookup for each segment, i.e. in total W lookups for comparing two time series, which is faster than the Euclidean distance where the calculation requires a value-by-value comparison of the two time series of length $T \gg W$.
e) *Lower-bounding Distance:* A distance measure is lower-bounding if the distance of two representations is always smaller or equal to the true Euclidean distance of the original time series: This property allows for pruning during time series matching, because there is no need to evaluate the Euclidean distance, if the representation distance between a query and an observation is too large. PAA and SAX distance measures have been proven to lower-bound the Euclidean distance [6, 14].

A representation technique that competes with SAX should provide similar properties. Moreover, by including other features it should give a higher representation accuracy and a more efficient time series matching.

2.4 SAX Extensions

Several SAX extensions have been proposed in the literature. We briefly review them regarding their representation and distance properties and summarize them in Table 1 together with SAX.

ESAX [7] extends SAX by taking not only the mean value but also the extreme values of each segment into account, which triples the representation size. Consequently, a lookup table would have a size A^6, which is huge compared to SAX.

1d-SAX [9] (1) segments a time series, (2) applies a piecewise linear approximation (PLA) representing each

Table 1 Properties of Representation Techniques

Technique	Representation		Distance		
	Size (bit)	Time	Storage (32 bits)	Time	LB
SAX	$W \cdot \mathrm{ld}(A)$	1	A^2	W	✓
ESAX	$3 \cdot W \cdot \mathrm{ld}(A)$	1	A^6	W	✓
1d-SAX	$W \cdot \mathrm{ld}(A)$	2	$W \cdot A$	W	(✓)
TFSA	$W \cdot (\mathrm{ld}(T)+66)$	3	0	W	✓
SAX_SD	$W \cdot (\mathrm{ld}(A)+32)$	1	A^2	W	✓
sSAX	$W \cdot \mathrm{ld}(A)$	1	$A^2_{\mathrm{seas}} + A^2_{\mathrm{res}}$	$4WL$	✓
tSAX	$W \cdot \mathrm{ld}(A)$	2	$A^2_{\mathrm{tr}} + A^2_{\mathrm{res}}$	$W+1$	✓

segment by mean level and slope, (3) discretizes these features using SAX, and (4) interleaves them to one representation. This requires a lookup table of size $W \cdot A$ and makes an asymmetric comparison between the original-valued query and the discretized dataset.

TFSA [15] represents a time series by trends of segments. The approach also uses non-discretized features leading to an increased representation size: The segments have different lengths requiring algorithms to pass three times over a time series for the split detection. The distance computation is based on the representation without a lookup table.

SAX_SD [16] represents every segment by its mean value (discretized like SAX) and its standard deviation (non-discretized), leading to an increased representation size. The distance calculation is performed with a lookup table for the mean value and directly on the standard deviation feature.

Table 1 comprises the following observations. *a) Representation Size:* All SAX extensions except 1d-SAX increase the representation size. For an unbiased evaluation of representation accuracy, it should be equal. *b) Representation Time:* 1d-SAX and TFSA need several passes over the dataset for the representation, which may be considered an acceptable penalty because the calculation still has linear complexity. *c) Distance Storage:* The distance storage often uses a lookup table with a small size that provides a fast distance calculation. *d) Distance Time:* The distance calculation needs W lookups for all techniques. For features other than the mean value, the distance calculation has additional costs. An evaluation requires an optimized distance function of each technique which is not part of this work. *e) Lower-bounding Distance:* The distance measures of ESAX, TFSA, and SAX_SD lower-bound the Euclidean distance measure. Although PLA is lower-bounding [3], it is not clearly stated for 1d-SAX.

Subsequently, we propose our symbolic approximations including the lower-bounding distance measures. In contrast to the SAX extensions mentioned above, they provide a higher representation accuracy while having the same representation size as SAX, which is already represented in the lower part of Table 1.

3 Season- and Trend-aware Symbolic Approximation

Time series from many domains such as weather, energy, or economy, exhibit deterministic behavior. The wind speed is often stronger in winter, while the solar irradiation has a strong daily season. In energy consumption, human behavior comes into play, where weekly patterns may be observed. Finally, economic time series may exhibit a trend from increasing sales.

As mentioned in the previous section, SAX has been applied to datasets agnostic to the existence of trends and/or seasons, whether they were synthetic [5, 6] or real-world datasets [11]. As a consequence, ignoring the deterministic behavior reduces the representation accuracy. We propose sSAX and tSAX that are aware of the time series' season and trend, respectively. Each technique is described along with its time series model, representation, distance measure, and properties. The underlying representations can be efficiently computed within a preprocessing step which is also required by SAX to normalize the time series.

3.1 Season-aware Symbolic Approximation - sSAX

sSAX is aware of the season of a time series by assuming the existance of a *seasonal component* in the time series. The remaining part of the time series forms the *residuals* that are unstructured information.

3.1.1 Time Series Model

A season-aware time series model is shown in Eq. 9, where $\underline{\mathrm{seas}}$ is the seasonal and $\underline{\mathrm{res}}$ is the residual component of the time series $\underline{x}$:

$$\underline{x} = \underline{\mathrm{seas}} + \underline{\mathrm{res}}. \tag{9}$$

A season repeats its behavior after L values (*season length*). We adopt the additive season model, which is a common assumption in many domains and can also be used to represent multiplicative seasons. The season is *extracted* by averaging all values at the same seasonal position l [4]. This technique allows us to provide a fast representation and a lower-bounding distance measure. A *seasonal feature* σ_l ($1 \le l \le L$) is given by:

$$\sigma_l = \frac{L}{T} \sum_{k=1}^{T/L} x_{(k-1)\cdot L+l} \tag{10}$$

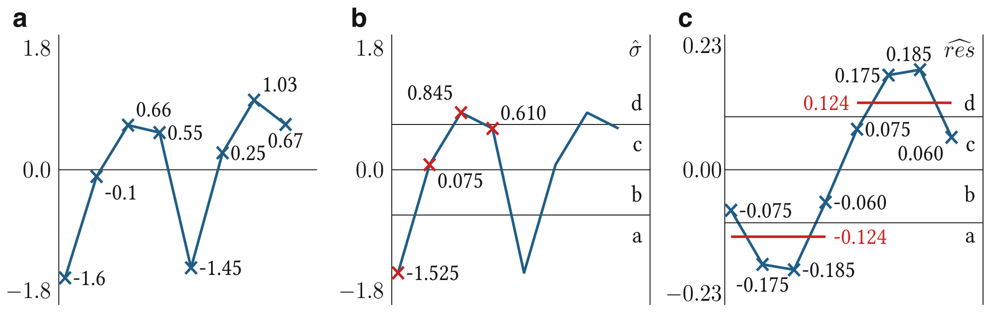

Fig. 2 Time series with season. **a** Time series. **b** Season. **c** Residuals

where T/L is the number of seasons in the time series that are iterated by k. The resulting features $\underline{\sigma}$ form the *season mask*.

3.1.2 Representation

Similar to SAX, the transformation in the low-dimensional space is carried out in two steps: the season-aware PAA reduces the time series in the time domain, the season-aware SAX subsequently reduces the value domain.

The *season-aware PAA* (sPAA) combines the season mask and the PAA of the residuals in a single representation. While PAA ignores the season of the time series by taking the mean value of a segment, sPAA explicitly extracts this season beforehand. Formally, the sPAA representation is the vector, where $W \cdot L$ divides T:

$$\underline{\bar{x}}_{\text{sPAA}}^{\mathsf{T}} = (\sigma_1, ..., \sigma_l, ..., \sigma_L, \overline{\text{res}}_1, ..., \overline{\text{res}}_w, ..., \overline{\text{res}}_W). \tag{11}$$

This representation is based on real numbers and is further reduced by discretization of sSAX. Let $A_{\text{seas}}, A_{\text{res}} \in \mathbb{N}_{>0}$ be the sizes of two alphabets and let $\underline{b}_{\text{seas}}$ and $\underline{b}_{\text{res}}$ be the respective vectors of breakpoints splitting the real numbers into A_{seas} and A_{res} intervals. Then, the sSAX representation is the vector $\underline{\hat{x}}_{\text{sSAX}}$:

$$\underline{\hat{x}}_{\text{sSAX}}^{\mathsf{T}} = (\widehat{\sigma}_1, ..., \widehat{\sigma}_l, ..., \widehat{\sigma}_L, \widehat{\text{res}}_1, ..., \widehat{\text{res}}_w, ..., \widehat{\text{res}}_W) \tag{12}$$

i.e., $\widehat{\sigma_l}$ / $\widehat{\text{res}}_w$ is σ_l / $\overline{\text{res}}_w$ discretized into A_{seas} / A_{res}.

The breakpoints are retrieved by two heuristics. We quantify the influence of the season on the time series by the *season strength* [13]: $R^2_{\text{seas}} = 1 - var(\underline{\text{res}})/var(\underline{x})$. Assuming, (1) the season strength of the dataset is known, (2) the time series are normalized, and (3) the residual and seasonal component are independent of each other, the following equations estimate the standard deviation of the season and the residuals:

$$sd(\underline{\text{res}}) = \sqrt{1 - R^2_{\text{seas}}} \tag{13}$$

$$sd(\underline{\text{seas}}) = \sqrt{1 - sd(\underline{\text{res}})^2} \tag{14}$$

where R^2_{seas} is the mean season strength of the dataset. Consequently, we set the breakpoints $\underline{b}_{\text{seas}}$ s.t. the area under normal distribution $\mathcal{N}(0, sd(\underline{\text{seas}}))$ is split into equiprobable regions $1/A_{\text{seas}}$. Regarding the residuals, we also assume normally distributed mean values. After season extraction, the residual component has less influence and its variance does not achieve 1 as assumed by Lin et al. [5]. Hence, we set the breakpoints $\underline{b}_{\text{res}}$ s.t. the area under normal distribution $\mathcal{N}(0, sd(\underline{\text{res}}))$ is split into equiprobable regions $1/A_{\text{res}}$.

Figure 2 shows a time series ($T = 8$) with $L = 4$ (Fig. 2a). Averaging the 1st and 5th value yields $\sigma_1 = -1.525$. Averaging all positions yields the season mask (Fig. 2b, red crosses). The residuals (Fig. 2c) remain after subtracting the seasonal component from the time series. With $W = 2$, the sPAA is $\underline{\bar{x}}_{\text{sPAA}}^{\mathsf{T}} \approx (-1.525, 0.075, 0.845, 0.610, -0.124, 0.124)$ (red crosses and bars). The season strength is $R^2_{\text{seas}} \approx 97.9\%$. With $A_{\text{seas}} = A_{\text{res}} = 4$, the breakpoints are at ± 0.67 (season) ± 0.098 (residuals) and 0.0 (both). Thus, $\underline{\hat{x}}_{\text{sSAX}}^{\mathsf{T}} = (a, c, d, c, a, d)$.

3.1.3 Distance

The distance measures of sPAA and sSAX are shown in Table 2. sSAX relies on a lookup table keeping the precalculated distances of the season and residual symbols, using $\underline{b}_{\text{seas}}$ and $\underline{b}_{\text{res}}$ as breakpoints. However, this look

Table 2 Distance Measures

Technique	$d_{*\text{PAA}}$	$d_{*\text{SAX}}$
sSAX	$\sqrt{\frac{T}{W\cdot L}}\sqrt{\sum_{l=1}^{L}\sum_{w=1}^{W}(\sigma_l - \sigma_l' + \overline{\text{res}}_w - \overline{\text{res}}'_w)^2}$	$\sqrt{\frac{T}{W\cdot L}}\sqrt{\sum_{l=1}^{L}\sum_{w=1}^{W}\text{cell}(\widehat{\sigma}_l, \widehat{\sigma}'_l, \widehat{\text{res}}_w, \widehat{\text{res}}'_w)^2}$
tSAX	$\sqrt{\sum_{t=1}^{T}(\Delta\theta_1 + \Delta\theta_2 \cdot (t-1) + \Delta\overline{\text{res}}_{\lfloor (t-1)/(T/W)\rfloor+1})^2}$	$\sqrt{c_t(\widehat{\phi}, \widehat{\phi}')^2 + \frac{T}{W}\sum_{w=1}^{W}\text{cell}(\widehat{\text{res}}_w, \widehat{\text{res}}'_w)^2}$

up table for four symbols may get large, which leads to an equivalent formulation for two smaller lookup tables. Let $c_s(a, a') = b_a - b_{a'+1}$ be a lookup table where $\underline{b}_a$ are the breakpoints of the given feature. Then $\text{cell}(\widehat{\sigma}, \widehat{\sigma}', \widehat{\text{res}}, \widehat{\text{res}}')$ can be calculated by:

$$\text{cell}(\widehat{\sigma}, \widehat{\sigma}', \widehat{\text{res}}, \widehat{\text{res}}') = \begin{cases} c_s(\widehat{\sigma}, \widehat{\sigma}') + c_s(\widehat{\text{res}}, \widehat{\text{res}}') & c_s(\widehat{\sigma}, \widehat{\sigma}') \geq -c_s(\widehat{\text{res}}, \widehat{\text{res}}') \\ c_s(\widehat{\sigma}', \widehat{\sigma}) + c_s(\widehat{\text{res}}', \widehat{\text{res}}) & c_s(\widehat{\sigma}', \widehat{\sigma}) \geq -c_s(\widehat{\text{res}}', \widehat{\text{res}}) \\ 0 & \text{otherwise} \end{cases} \tag{15}$$

3.2 Trend-aware Symbolic Approximation - tSAX

Similar to sSAX, tSAX is aware of the trend of a time series and captures this behavior in a *trend component*.

3.2.1 Time Series Model

A trend-aware time series model is shown in Eq. 16, where $\underline{\text{tr}}$ is the trend and $\underline{\text{res}}$ is the residual component of the time series $\underline{x}$. Again we adopt an additive combination:

$$\underline{x} = \underline{\text{tr}} + \underline{\text{res}}. \tag{16}$$

Linear regression extracts the trend component of the time series. This technique is efficient and allows for the lower-bounding property of subsequent distance measures. It estimates two features θ_1 (*base value*) and θ_2 (*slope*) forming the trend-aware time series model:

$$\underline{x} = \theta_1 + \theta_2 \cdot (\underline{t} - 1) + \underline{\text{res}} \tag{17}$$

where $\underline{t}^\mathsf{T} = (1, ..., t, ..., T)$ is the vector of time instances. Linear regression selects these features s.t. they minimize the sum of squared residuals. Moreover, it yields two important properties: the sum of the residuals is always zero and the trend component and the residuals are uncorrelated. θ_1 and θ_2 are interdependent because the time series is normalized[1]. Therefore, θ_1 and θ_2 are combined to a single *trend feature* ϕ representing the angle between the x-axis and the trend component:

$$\phi = \arctan(\theta_2). \tag{18}$$

3.2.2 Representation

Similar to SAX, the trend-aware symbolic approximation transforms a time series in two steps, the trend-aware PAA reduces the time domain, the trend-aware SAX reduces the value domain.

The *trend-aware PAA* (tPAA) representation is the vector of the trend feature and the mean values of a residual component:

$$\overline{\underline{x}}_{\text{tPAA}}^\mathsf{T} = (\phi, \overline{\text{res}}_1, ..., \overline{\text{res}}_w, ..., \overline{\text{res}}_W). \tag{19}$$

Let $A_{\text{tr}}, A_{\text{res}} \in \mathbb{N}_{>0}$ be the sizes of two alphabets. Let $\underline{b}_{\text{tr}}$ and $\underline{b}_{\text{res}}$ be the respective vectors of breakpoints that split the real numbers into A_{tr} and A_{res} intervals. The tSAX representation is the vector $\widehat{\underline{x}}_{\text{tSAX}}$:

$$\widehat{\underline{x}}_{\text{tSAX}}^\mathsf{T} = (\widehat{\phi}, \widehat{\text{res}}_1, ..., \widehat{\text{res}}_w, ..., \widehat{\text{res}}_W) \tag{20}$$

where $\widehat{\phi}$ and $\widehat{\text{res}}_w$ denote the discretization of ϕ and $\overline{\text{res}}_w$ into the alphabets A_{tr} and A_{res}.

Due to normalization, ϕ is bounded by $\phi_{\max}$:

$$|\phi| \leq \phi_{\max}, \text{ where } \phi_{\max} = \tan^{-1}\sqrt{1/var(\underline{t})}. \tag{21}$$

$\phi_{\max}$ that is reached if the time series is a perfect trend with zero residuals. Using this observation and the assumption that each trend is equiprobable, we set the breakpoints $\underline{b}_{\text{tr}}$ s.t. the uniformly distributed area between $[-\phi_{\max}, \phi_{\max}]$ is split into regions of probability $1/A_{\text{tr}}$. Regarding the residuals, we adopt normally distributed mean values similar to SAX. After extracting the trend, the residual component has less influence. We quantify the influence of the trend on the time series by the *trend strength* [13]: $R_{\text{tr}}^2 = 1 - var(\underline{\text{res}})/var(\underline{x})$. Assuming that (1) the trend strength of the dataset is known and (2) the time series are normalized, the standard deviation of the residuals is estimated by:

$$sd(\underline{\text{res}}) = \sqrt{1 - R_{\text{tr}}^2} \tag{22}$$

where R_{tr}^2 is the mean trend strength of the dataset. Thus, we set $\underline{b}_{\text{res}}$ s.t. the area under normal distribution $\mathcal{N}(0, sd(\underline{\text{res}}))$ is split into equiprobable regions $1/A_{\text{res}}$.

Figure 3 shows a time series (Fig. 3a, $T = 8$) with a trend (Fig. 3b) and residuals (Fig. 3c). With $W = 2$, the tPAA is $\overline{\underline{x}}_{\text{tPAA}}^\mathsf{T} \approx (0.387, -0.02, 0.02)$ (ϕ and red bars). The trend strength is $R_{\text{seas}}^2 \approx 99.2\%$. With $A_{\text{tr}} = A_{\text{res}} = 4$, the breakpoints are at ± 0.19 (trend) ± 0.06 (residuals) and 0.0 (both). Thus, $\widehat{\underline{x}}_{\text{tSAX}}^\mathsf{T} = (d, b, c)$.

3.2.3 Distance

The distance measures of tPAA and tSAX are again shown in Table 2. Let us note $\Delta f = f - f'$ as a shorthand for the difference between a feature of time series $\underline{x}$ and $\underline{x}'$. The tSAX distance measure relies on a lookup table

[1] The interested reader finds the proof in an extended version of this paper: https://arxiv.org/abs/2105.14867.

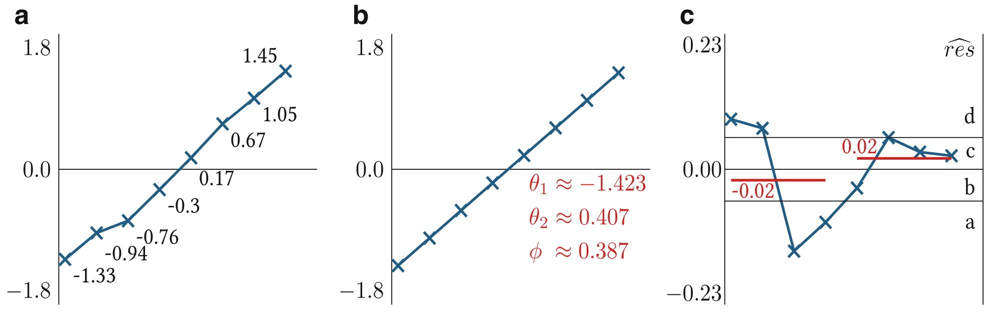

Fig. 3 Time series with trend. **a** Time series. **b** Trend. **c** Residuals

c_t for the trend feature using $\underline{b}_{\text{tr}}$ as breakpoints. The measure expresses the minimum distance of two trend components represented by $\widehat{\phi}$ and $\widehat{\phi}'$. For the residuals, tSAX relies on the lookup table cell from SAX using $\underline{b}_{\text{res}}$ as breakpoints (Eq. 8).

3.3 Properties of Representation Techniques

We review the properties of sSAX and tSAX (Table 1).

a) *Representation Size:* The alphabets A_{seas}, A_{tr}, and A_{res} are chosen s.t. the representation size of sSAX and tSAX equals the representation size of SAX. If they are not a power of 2, we allow interleaving [9].
b) *Representation Time:* For sSAX, the representation needs a single pass over the time series because season mask and residuals can be calculated simultaneously. tSAX needs an additional pass for the linear regression.
c) *Distance Storage:* Both representations need two lookup tables of size A_{res}^2 and $A_{\text{seas}}^2/A_{\text{tr}}^2$. The alphabet sizes determine the memory cost for the distance calculation.
d) *Distance Time:* The sSAX distance measure needs at most $4 \cdot W \cdot L$ lookups instead of W lookups due to the combinations of season and residual symbols. Although sSAX may use fewer segments for the residuals than SAX does for the time series, it leads to more lookups. The tSAX distance measure needs only a single lookup for the trend and W lookups for the residuals.
e) *Lower-bounding Distance:* Most importantly, d_{sPAA}, d_{tPAA}, d_{sSAX}, and d_{tSAX} are lower-bounding[1].

4 Experimental Evaluation

We compare our techniques sSAX and tSAX to the competitors SAX and 1d-SAX and evaluate their representation accuracy as well as their accuracy and efficiency for time series matching.

4.1 Matching Methods

Besides the evaluation of the representation accuracy, it is relevant to investigate the performance of time series matching, which we conduct with respect to *exact* and *approximate* matching.

Exact matching returns the time series of a dataset that has the minimum Euclidean distance to the query time series. We compute the representation distances between the query and all time series in the dataset and sort them in ascending order. Thereafter, the Euclidean distance from the observations is calculated in the order of their representation distance, keeping track of the "best-so-far" observation until it's distance is less than the representation distance of the next observation. Such an early stop is possible due to the lower-bounding property: subsequent observations never have a Euclidean distance below the "best-so-far" distance.

Approximate matching returns the time series with the minimum representation distance as an approximate match. If there are several observations with minimum representation distance, it returns the observation with the minimum Euclidean distance.

For the efficiency evaluation, we include the results of naive time series matching, i.e., the query is matched against the dataset with the Euclidean distance directly.

4.2 Datasets

The representation techniques are evaluated on synthetic time series datasets with configurable characteristics as well as on two real-world time series datasets (see Table 3 for dataset dimensions).

Season: A Season dataset contains 1,000 random walk time series, each of which is overlaid with a season mask of length 10. In compliance with [11], the time series length varies between 480 and 1,920. All time series of a dataset have the same season strength fixed between 1 and 99%, with a tolerance of 0.5 percentage points (pp) in both directions.

Trend: A Trend dataset contains 1,000 random walk time series overlaid with a trend. Similarly to Season, the time series length varies between 480 and 1,920. All time series of a dataset have the same trend strength between 1 and 99% with a tolerance of 0.5 pp.

Table 3 Dataset Dimensions

Dataset	Dataset Size I	Length T
Season	1,000	[480; 960; 1,440; 1,920]
Trend	1,000	[480; 960; 1,440; 1,920]
Metering	5,958	21840
Economy	6,400	300
Season (Large)	[6,510,417; 13,020,833]	960

Metering: The Metering dataset is the result of the Smart Metering Project initiated from the Irish Commission for Energy Regulation [12] and contains the electricity consumption of households and SMBs in Ireland between 7/2009 and 12/2010 measured in kW/h at a 30 min. granularity. All time series exhibit seasonal components but lack a strong trend. The sSAX is evaluated with respect to the daily season (season length of 48), which has an average season strength of 18.3%.

Economy: The Economy dataset contains about 100,000 time series from different domains. It originates from the M4-Competition to systematically evaluate the accuracy of forecast methods [8]. Each time series has a specified interval (year, quarter, month, other) and exhibits a trend component. In compliance with the time series dataset (Sec. 2), only time series with the same length are selected, i.e., 6,400 time series measured for 25 years with monthly granularity.

Season (Large): For the efficiency evaluation, we include two Season datasets with a size of 50 and 100 GB. The time series length is fixed to 960 values and the season strength of a time series may vary. We select datasets such that their season strength is on average 10.0% (weak), 50.0% (medium), and 90.0% (strong).

For the accuracy evaluation, each time series of a dataset acts as query and is matched against the remaining time series. For the efficiency evaluation on Season (Large), we randomly select up to 50 query time series for each dataset. We limit an experiment to four hours. Since the runtime differs for each query, each technique is evaluated with the same set of queries.

4.3 Output Variables

We employ four output variables to assess the accuracy and efficiency of our symbolic approximations techniques:

Tightness of Lower Bound: The representation accuracy is evaluated with the tightness of lower bound (TLB) in accordance with [5]. The TLB is the ratio between the representation distance and the Euclidean distance as follows: $\mathrm{TLB}(\underline{x}, \underline{x}') = d_{*\mathrm{SAX}}(\widehat{\underline{x}}, \widehat{\underline{x}}')/d_{\mathrm{ED}}(\underline{x}, \underline{x}')$, where $d_{*\mathrm{SAX}}$ is either d_{SAX}, $d_{\mathrm{1d\text{-}SAX}}$, d_{sSAX}, or d_{tSAX}. To evaluate the TLB of a time series dataset, the mean TLB of all time series combinations is calculated.

Pruning Power: Exact matching is improved by pruning observations to terminate linear search earlier. The pruning power (PP) expresses the fraction of observations that can be pruned without evaluating their Euclidean distance [3]. A PP of 0 means that no observations are pruned, a PP close to 1 means that the linear search terminates after very few observation.

Approximate Accuracy: Approximate matching is improved if the approximate match is closer to the Euclidean distance of the exact match. We introduce the output variable approximate accuracy (AA) which is the quotient of the Euclidean distance between the query and the exact match and the Euclidean distance between the query and the approximate match. An AA of 0 means that the approximate match is very inaccurate, an AA of 1 means that the approximate match is as accurate as the exact match.

Runtime: The efficiency of time series matching is evaluated with the runtime. We measure the wall-clock time in seconds for the calculation of the representation distances and the Euclidean distances.

4.4 Configurations

Table 4 summarizes all possible configurations for the number of segments W and the alphabet size A for each dataset and each representation technique. The representation size is fixed, which sets the alphabet A_{res} of sSAX and tSAX in accordance to A_{seas} and A_{tr}. 1d-SAX uses the alphabet A_a for the base value of a segment and the alphabet A_s for the slope [9]. Alphabet sizes less or equal to 4 are ignored since they evidently cause a high accuracy loss. We limit the size of a lookup table to 4 Mb, which corresponds to an alphabet size of 1,024. The standard deviation of sSAX and tSAX to discretize the residuals is derived from the component strength (Eqs. 13 and 22).

4.5 Soft- and Hardware

All representation techniques are implemented using R [10]. For the runtime evaluation, matching methods are implemented in C compiled with GCC 6.3.0. Experiments run on a machine with Intel(R) i7 Processor 6660U@2.60GHz, 20 GB of RAM, 1 TB HDD, and 500 GB SSD. Each time series is stored as a binary file on disk. Time series representations and lookup tables are kept in-memory, while time series are read from disk without system cache buffering.

4.6 Representation Accuracy

We evaluate the representation accuracy utilizing the TLB (Fig. 4) keeping the representation size constant for each dataset; the possible configurations are given in Table 4.

Table 4 Configurations of Representation Techniques

Synthetic	**W**	**A or A_{res}**	**A_{seas} or A_{tr}**	**Size/bit**
SAX	[32; 40; 48; 96]	[1,024; 256; 101; 10]	–	320
sSAX	[24; 48; 48]	[1,024; 32; 64]	[256; 256; 9]	320
tSAX	[32; 40; 48; 96]	$\lfloor 2\hat{}((320 - \mathrm{ld}(A_{tr}))/W)\rfloor$	[32; 128; 1,024]	320
Metering	**W**	**A or A_{res}**	**A_{seas} or A_{tr}**	**Size/bit**
SAX	[455; 520; 728; 910]	[256; 128; 32; 16]	–	3,640
sSAX	455	[191; 165; 142; 123]	[16; 64; 256; 1024]	3,640
Economy	**W**	**A, A_a, or A_{res}**	**A_{tr} or A_s**	**Size/bit**
SAX	[10; 12; 15; 20; 30]	[256; 101; 40; 16; 6]	–	80
1d-SAX	[10; 12; 15; 20]	$\lfloor 2\hat{}((80 - \mathrm{ld}(A_s)\cdot W)/W)\rfloor$	[8; 16; 32]	80
tSAX	[10; 12; 15; 20; 30]	$\lfloor 2\hat{}((80 - \mathrm{ld}(A_{tr}))/W)\rfloor$	[16; 64; 256; 1,024]	80

On the synthetic datasets, we compare the TLB of SAX to sSAX and tSAX (Figs. 4a and 4b), grouped by time series length and component strength. Each cell presents the difference in percentage points (pp) between the mean TLB of the most accurate sSAX/tSAX and SAX configuration. Figure 4a shows that sSAX gains accuracy compared to SAX with up to 86 pp. The longer the time series and the stronger the season, the higher is the accuracy gain. Without a season ($R^2_{seas} = 1\%$), a small amount of the representation size is assigned to represent the season, nonetheless. Therefore, sSAX is slightly less accurate than SAX. tSAX gains accuracy by only 1.2 pp and has very slight losses in the absence of a trend (Fig. 4b). The gains of tSAX are lower than expected for two possible reasons: (1) The distance of discretized mean values from SAX already captures the global trend of a time series and, (2) the normalization transforms the time series such that the change in the mean level becomes smaller. Therefore, tSAX has not much room for improvement.

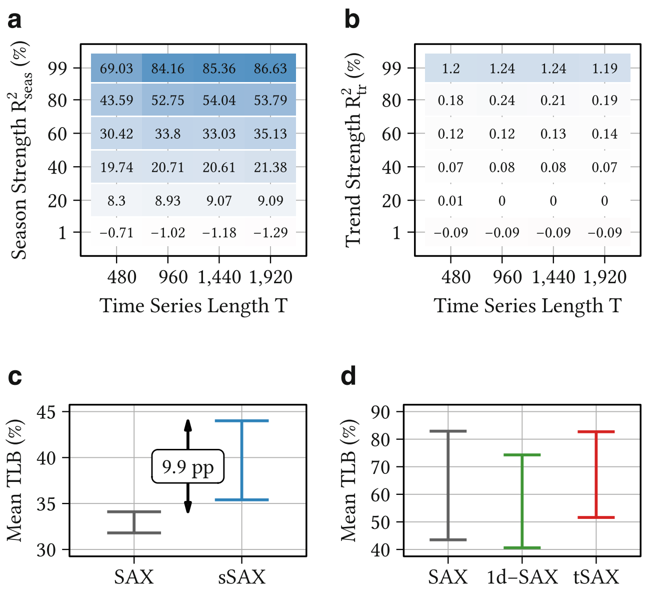

Fig. 4 Increase of TLB compared to SAX. **a** Season. **b** Trend. **c** Metering. **d** Economy

Figures 4c and 4d illustrate the results for the real-world datasets showing the min and max mean TLB reached with the chosen configurations. On the Metering dataset, the best sSAX configuration gains up to 9.9 pp compared to the best SAX configuration (Fig. 4c). Thus, taking the season into account leads to a much higher representation accuracy. On the Economy dataset, we include 1d-SAX in our comparison, which is the only trend-aware SAX extension that has the same representation size as SAX. Overall, tSAX has a better representation accuracy than 1d-SAX, as it better takes advantage of the available representation size. tSAX performs on par compared to the best SAX (Fig. 4d). Since 1d-SAX already performs worse than SAX on the Economy data, we omit the comparison on Trend and use the original SAX as our baseline.

4.7 Exact Matching

For exact matching, we evaluate pruning power (Fig. 5) and efficiency/runtime (Table 5).

We begin with the pruning power. On the synthetic datasets, sSAX and tSAX exhibit a gain in pruning power compared to SAX. Remarkably, sSAX improves the pruning power up to 99 pp in the presence of a strong season (Fig. 5a). With no season, sSAX has a negligibly worse pruning power. The gains of tSAX are very limited, again (Fig. 5b).

This behavior is confirmed on the real-world datasets. On Metering, sSAX gains 3.8 pp in pruning power (reaching 6.6% in Fig. 5c). sSAX can prune 393 time series while SAX only prunes 274 of the 5,958 time series. On Economy, the best SAX configuration already has a very high pruning power (Fig. 5d) with 97.4%. tSAX outperforms 1d-SAX and performs on par with SAX.

With respect to runtime efficiency, Table 5 details the runtimes for both disks (HDD, SSD) on the Season (large) datasets with 50GB and 100GB. The runtime is broken down into one part for calculating the representation distances including result ordering (Repr.) and in a second part for accessing the time series and calculating the true dis-

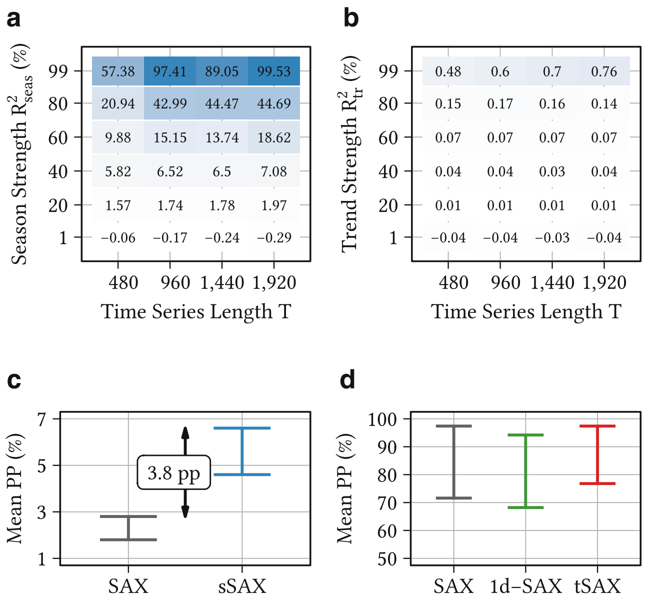

Fig. 5 Increase of pruning power compared to SAX. **a** Season. **b** Trend. **c** Metering. **d** Economy

tances (Raw). For each season strength, the sum of both parts indicates the mean runtime per query. Calculating the representation distances is independent of the season strengths.

The table reveals that sSAX is (1) faster for all datasets from HDD even when there is only a weak season strength and (2) faster for all datasets from SSD for a significant season strength. The most striking result to emerge from the data is that sSAX is up to three orders of magnitude faster for time series with a strong season, since much more time series can be pruned from the dataset. On HDD, sSAX requires approximately 17sec for querying the 100 GB dataset, due to the increased pruning power. A naïve matching of a query without representation technique would require 1.7 h (50 GB on SSD) and 3.85 h (100 GB on SSD). Due to space limitations, we only present the efficiency results for sSAX. Since tSAX does not achieve such a high pruning power compared to SAX, the number of compared time series is very close which also leads to very similar execution times of tSAX and SAX.

4.8 Approximate Matching

For approximate matching, we evaluate the accuracy of an approximate match compared to the exact match utilizing the AA (Fig. 6).

Figures 6a and 6b show the increase of AA on the synthetic datasets by time series length and component strength. The longer the time series and the stronger the deterministic component, the higher the gains of sSAX and tSAX over SAX. sSAX reaches up to 47 pp improvement on Season (Fig. 6a). tSAX only reaches minor improvements on Trend with up to 0.14 pp (Fig. 6b). On datasets with a high component strength, both representations reach an AA of ≈ 99%. sSAX and tSAX reach this accuracy thanks to their accurate representation. SAX reaches this accuracy since most observations have the same representation. So, SAX re-evaluates their Euclidean distance to the query in order to retrieve the most accurate approximate observation. Thus, it reaches a slightly higher AA, mainly due to the evaluation of the Euclidean distance.

On the real-world datasets, sSAX and tSAX show a similar behavior. All sSAX configurations outperform SAX on Metering, and the best sSAX configuration is up to 1.3 pp more accurate than the best SAX configuration (Fig. 6c). Exact search shows that the time series are very similar regarding the Euclidean distance, as also shown by the pruning power of 6.6% in Fig. 5c. However, the approximate match already reaches 91.5% of the exact match accuracy. Although tSAX provides a higher approximate accuracy than 1d-SAX (95.4% vs. 84.2%), it is slightly behind the best approximate accuracy of SAX (96.9%) (Fig. 6d).

Table 5 shows the details of the efficiency evaluation and reveals that approximate matching with sSAX is slower

Table 5 Matching Efficiency on Season (Large)

HDD			R^2_{seas} = 10.0%		R^2_{seas} = 50.0%		R^2_{seas} = 90.0%	
Size	**Technique**	**Repr.**	**Raw**	**Sum**	**Raw**	**Sum**	**Raw**	**Sum**
50 Gb	SAX	1.80	135.82	137.61	1,801.12	1,802.92	6046.75	6048.55
	sSAX	8.67	41.76	**50.43**	0.54	**9.21**	0.08	**8.75**
100 Gb	SAX	3.69	73.61	77.30	4181.02	4184.72	13,423.47	13,427.16
	sSAX	16.86	4.77	**21.63**	1.09	**17.95**	0.11	**16.97**
SSD			R^2_{seas} = 10.0%		R^2_{seas} = 50.0%		R^2_{seas} = 90.0%	
Size	**Technique**	**Repr.**	**Raw**	**Sum**	**Raw**	**Sum**	**Raw**	**Sum**
50 Gb	SAX	1.84	4.05	**5.89**	101.61	103.45	850.81	852.65
	sSAX	9.12	0.71	9.83	0.04	**9.16**	0.02	**9.14**
100 Gb	SAX	3.80	8.29	**12.09**	115.14	118.95	1,088.80	1,092.60
	sSAX	17.99	1.05	19.04	0.07	**18.06**	0.02	**18.02**

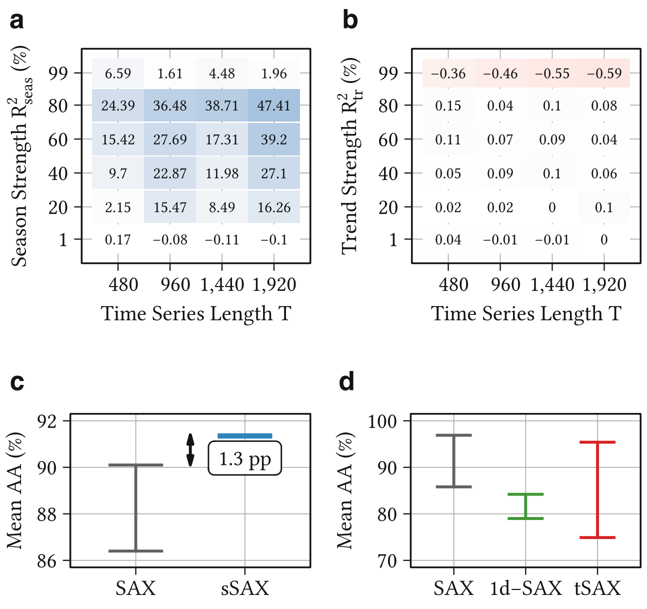

Fig. 6 Increase of approximate accuracy compared to SAX. **a** Season. **b** Trend. **c** Metering. **d** Economy

compared to SAX due to an increased number of lookups. However, these approximate matches are much more accurate, as Fig. 6 suggests.

5 Conclusion and Future Work

We have proposed two novel symbolic approximations sSAX and tSAX to improve representation accuracy and time series matching compared to state-of-the-art techniques. Our evaluation shows that considering the deterministic features during time series matching is worth the effort. It improves the representation accuracy, especially if the deterministic component is strong, and the accuracy of exact and approximate matching. Moreover, exact matching with sSAX is more efficient by up to three orders of magnitude. Even if exact matching was carried out with indexes based on SAX such as iSAX and its successors [11, 17], it could not avoid the disk access for Euclidean distance calculation. While sSAX provides significant improvements, the improvements of tSAX are interestingly far less significant.

In the future, we will concentrate on representing combinations of deterministic components since time series usually exhibit (several) seasons simultaneously in combination with an (potentially non-linear) trend. Additionally, trend and season patterns are not always stable. So, maintenance strategies for the representations are also an interesting future research direction.

Furthermore, recent work has focused on indexes based on SAX for matching billions of time series [17]. However, the authors analyzed rather short time series ($T \leq 640$) and our approximations have the potential to efficiently index and match much longer time series thanks to their higher representation accuracy.

Funding Open Access funding enabled and organized by Projekt DEAL.

References

1. Agrawal R, Faloutsos C, Swami A (1993) Efficient similarity search in sequence databases. FODO 730:69–84
2. Butler M, Kazakov D (2015) SAX discretization does not guarantee equiprobable symbols. IKDE 27(4):1162–1166. https://doi.org/10.1109/TKDE.2014.2382882
3. Chen Q, Chen L, Lian X, Liu Y, Yu JX (2007) Indexable PLA for efficient similarity search. In: Proc. of VLDB, pp 435–446
4. Kendall MG, Stuart A (1983) The advanced theory of statistics vol 3. Griffin, , pp 410–414
5. Lin J, Keogh EJ, Lonardi S, Chiu B (2003) A symbolic representation of time series, with implications for streaming algorithms. In: Workshop Proc. of SIGMOD, pp 2–11 https://doi.org/10.1145/882082.882086
6. Lin J, Keogh E, Wei L, Lonardi S (2007) Experiencing SAX: A novel symbolic representation of time series. Data Min Knowl Discov 15(2):107–144. https://doi.org/10.1007/s10618-007-0064-z
7. Lkhagva B, Suzuki Y, Kawagoe K (2006) New time series data representation ESAX for financial applications. In: ICDEW https://doi.org/10.1109/ICDEW.2006.99
8. Makridakis S, Spiliotis E, Assimakopoulos V (2018) The M4 competition: results, findings, conclusion and way forward. Int J Forecast 34(4):802–808. https://doi.org/10.1016/j.ijforecast.2018.06.001
9. Malinowski S, Guyet T, Quiniou R, Tavenard R (2013) 1d-SAX: a novel symbolic representation for time series. In: Proc. IDA https://doi.org/10.1007/978-3-642-41398-8_24
10. R Core Team (2018) R: a language and environment for stat. Computing. R Foundation for Stat. Comp, Vienna (https://www.R-project.org/)
11. Shieh J, Keogh EJ (2008) iSAX: indexing and mining terabyte sized time series. In: Proc. of SIGKDD, pp 623–631 https://doi.org/10.1145/1401890.1401966
12. The Commission for Energy Regulation (2015) CER smart metering project (www.ucd.ie/issda)
13. Wang X, Smith K, Hyndman R (2006) Characteristic-based clustering for time series data. Data Min Knowl Discov. https://doi.org/10.1007/s10618-005-0039-x
14. Yi BK, Faloutsos C (2000) Fast time sequence indexing for arbitrary Lp norms. In: Proc. of VLDB

15. Yin H, Yang S, Zhu X, Ma S, Zhang L (2015) Symbolic representation based on trend features for biomedical data classification. Front Inform Technol Electron Eng 16(9):744–758. https://doi.org/10.3233/THC-151002
16. Zan CT, Yamana H (2016) An improved symbolic aggregate approximation distance measure based on its statistical features. In: Proc. of iiWAS, pp 72–80 https://doi.org/10.1145/3011141.3011146
17. Zhang L, Alghamdi N, Eltabakh MY, Rundensteiner EA (2019) TARDIS : distributed indexing framework for big time series data. In: Proc. of ICDE https://doi.org/10.1109/ICDE.2019.00110
18. Zoumpatianos K, Palpanas T (2018) Data series management : fulfilling the need for big sequence analytics. In: Proc. of ICDE, pp 16–17 https://doi.org/10.1109/ICDE.2018.00211