Towards Online Multi-Model Approximation of Time Series

Thanasis G. Papaioannou, Mehdi Riahi, and Karl Aberer
School of Computer and Communication Sciences
Ecole Polytechnique Fédérale de Lausanne (EPFL)
Lausanne, Switzerland 1015
Email: firstname.lastname@epfl.ch

***Abstract*—The increasing use of sensor technology for various monitoring applications (e.g. air-pollution, traffic, climate-change, etc.) has led to an unprecedented volume of streaming data that has to be efficiently aggregated, stored and retrieved. Real-time model-based data approximation and filtering is a common solution for reducing the storage (and communication) overhead. However, the selection of the most efficient model depends on the characteristics of the data stream, namely rate, burstiness, data range, etc., which cannot be always known a priori for (mobile) sensors and they can even dynamically change. In this paper, we investigate the innovative concept of efficiently combining multiple approximation models in real-time. Our approach dynamically adapts to the properties of the data stream and approximates each data segment with the most suitable model. As experimentally proved, our multi-model approximation approach always produces fewer or equal data segments than those of the best individual model, and thus provably achieves higher data compression ratio than individual linear models.**

***Keywords*-lossy compression; regression; storage scheme; efficient data management; error norm**

I. Introduction

Recent advances in sensor technology have enabled the availability of a multitude of (often privately-held) sensors. Embedded sensing functionality (e.g. noise, accelerometer, temperature, GPS, RFID etc.) is now included in mobile devices, such as phones, cars, buses, etc. Environmental and health-care applications based on community sensing in urban areas have been already envisioned, e.g. personalized carbon exposure and impact calculators, healthy lifestyle estimators, traffic monitoring, etc. The large amount of these (mobile) sensing devices and the huge volume of raw monitored data pose new challenges for the sustainable storage and efficient retrieval of sensor data streams.

To this end, multiple regression and filtering techniques (referred to as *models*) have been proposed [1], [2], [3] for the online approximation of time series within a certain error norm (i.e. *lossy* compression). L_∞ is a common error norm that allows modeled data values to fall within a maximum error bound from the raw ones. These models exploit the inherent correlations (e.g. with time or among data streams) in time series to split data in pieces and approximate each segment with a certain mathematical function derived by the model. However, the potential varying burstiness (and possibly rate) of the data streams along time and the variable standard error introduced by the sensor mobility often result in limited effectiveness of a single model for approximating data within the prescribed error bound during a certain period. The same argument is also valid for other time series that may exploit variable burstiness in different time periods, e.g. stock prices during volatile or non-volatile market periods.

In this paper, we propose the innovative concept of combining multiple statistical models for approximating time series. The intuition behind this approach is that different data periods of a sensor stream can be better approximated by different models, thus resulting, overall, in fewer and longer segments. This is because of the greater flexibility offered by the alternative models. We propose our multi-model longest-fit algorithm and prove its correctness for approximating the data stream within the specified error bound. By an extensive series of experiments with both real and artificial data traces, we prove that our approach always produces fewer or equal segments than any of its constituent models employed individually. Moreover, when linear models are employed, we experimentally show that our approach can achieve significant compression improvement over any constituent linear approximation scheme. We also define an appropriate storage scheme for storing approximated data segments. Our approach is fully implemented and can serve as a framework for combining arbitrary online approximation schemes for time series.

The remainder of this paper is organized as follows: In Section II, we compare our approach to the related work and emphasize on the innovative characteristics of our approach. In Section III, we discuss further motivation for our work. In Section IV, we introduce our multi-model approximation algorithm. In Section V, we define the storage schema for our approach. In Section VI, we present our experimental results that prove the effectiveness of our approach. Finally, in Section VII, we conclude our work.

II. Related Work

There is significant work in the literature related to model-based data compression. Among different time series approximation techniques, piecewise linear approximation has been widely used [4]. In [1], two piecewise linear approximation models are proposed, namely *Swing* and *Slide*, and are compared to previous linear models, such as cache and linear filters. Cache filter approximates data within a segment with a

constant value, which can be the first value of the segment, the mean or the median (referred to as Poor Man's Compression-MidRange (PMC-MR) [5]). In linear filter [1], data points are approximated within the error bound by a line connecting the first and second points of the segment.

In [6], a framework for partitioning and approximating raw data segments based on a user-specified mathematical function or model is introduced. In [7], a general technique is introduced to reduce the space complexity of the offline and online optimal and approximate synopsis construction algorithms. The V-optimal histogram [8] for synopsis construction finds the best piecewise constant approximation of n values with at most B pieces, while the sum of the square errors between the actual and approximated values is minimized. Instead, we aim to maximize the compression ratio by approximating the data in real-time with multiple models within a predefined error bound. In [9], piecewise linear approximation algorithms are categorized in three groups, sliding windows, top-down, and bottom-up. Among these, only the sliding window approaches can be online, while top-down and bottom-up would perform better. To this end, the authors propose a new algorithm that combines the online property of sliding windows and performance of the bottom-up algorithm. This approach can be employed by our multi-modeling technique; yet, it needs a predefined buffer length. If the buffer is small then it may produce many small segments, while if large, it will introduce a high latency for the stream approximation.

In [10], it is recognized that different approximation models are more appropriate for data streams of different statistical properties, similarly to our work. However, the approach in [10] aims to find the model best approximating the data stream, based on the hit ratio. Our work aims to effectively find the best combination of models for approximating the various segments of the stream, which has been experimentally proved in Section VI to achieve higher compression than any of the constituent models, when individually employed.

The optimal approximation of one-dimensional signals under L_∞ in linear space is formulated as a LP problem in [11]. However, their approach is limited to L_∞, as opposed to ours that can be employed for arbitrary error norms. Also, in the general case of linearly combining arbitrary base approximating functions in [11], all parameters associated to each function have to be stored per segment, as opposed to our approach that stores only the parameters of a single model.

Finally, [3], [2] dynamically identify and exploit spatial correlations among different data streams. [2] jointly compresses data streams within an error bound with respect to a base signal employing a polynomial-time approximation scheme. Our approach can also exploit spatial correlations among different data streams or different attributes of a data stream , e.g. humidity and temperature from the same sensor.

III. Motivation

Based on their generation algorithm, different online approximation models (as those described in Section II) exploit the correlations of the data in a different way. For example, PMC-MidRange (or simply *MidRange*) piecewise constant approximation is expected to approximate a *longer* data segment of a data stream oscillating across a certain constant value within the error bound, than a linear piecewise approximation technique, such as *Linear* or *Swing* filters; the latter models require that the data stream values consistently follow a certain direction. This case is illustrated in Fig. 1, where a small period of a raw data stream produced by a temperature sensor is approximated by MidRange and Swing models. Only the edge points of the linear segments produced by each model are plotted in Fig. 1. As depicted therein, in the beginning of the stream period, Swing filter approximates with 1 linear segment a data stream subset of 17 raw values, while, for the same raw data, MidRange approximation model needs 3 linear segments. The situation is reversed at the end of the stream period, where 24 raw data values are approximated with 1 linear segment by MidRange, as compared to 3 linear data segments constructed by Swing for the same data. Another example is that, other stream trends in certain periods that involve multiple direction changes or periodicity, such as parabolic or sinusoid, could be better approximated by 2nd-degree or 5th-degree polynomials respectively. Other properties of the data stream that can be differently exploited in different periods by the various models, described in Section II, involve rate variability, oscillation length (i.e. burstiness), direction of trend, rate of trend change, the approximation error bound, etc.

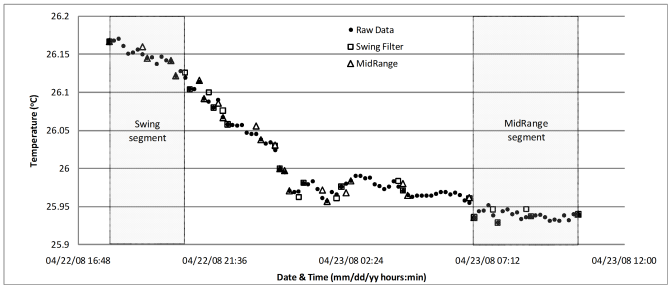


Fig. 1. Different time segments of the data stream are better approximated by different models.

IV. The Approach

Online piecewise approximation algorithms seek to find the parameters of a certain mathematical function, so as to fit the raw values of a segment of the data stream within a maximum error bound. When a raw data value cannot be approximated within the error bound by a specific instantiation of the "fitting" function of the model, then a new data segment is initiated; within the new data segment, a new instantiation of the fitting function has to be found and employed for data approximation. Each approximation model has to employ a fixed number of initial raw values in a data segment, in order to find an instantiation of its fitting function for this segment; e.g., 1 value for the cache filter, 2 values for the linear filter, 3 values for 2nd-degree polynomial regression, etc. We refer to this model requirement as *initialization*. The raw values of the segment necessarily fit into the model function

during its initialization; the minimum initialization length that may be required by a model is 1 value. Note that, after the initialization, the model parameters may still change as new data arrives to update the instantiation of the model.

In our approach, a collection of models are jointly employed for approximating the data stream. Each data segment is approximated by the most effective model instantiation for that segment. The model effectiveness for a segment is determined based on the *segment length* (i.e. fitting period) in terms of raw data values that are approximated by the same instance of the model and its achievable *compression ratio* for this segment.

More formally, we want to construct a multi-segment function. For simplicity, we consider the approximation $H(.)$ of a single attribute of the stream that exploits a temporal correlation, as follows:

$$H(t) = \begin{cases} h_1(t)\,, & t \in [t_0,\ t_1] \\ h_2(t)\,, & t \in (t_1,\ t_2] \\ & \dots \\ h_N(t)\,, & t \in (t_{N-1},\ t_N]\,, \end{cases} \quad (1)$$

where h_i is a certain instantiation of the formula $f_m(t)$ of the model m that achieves the highest compression ratio for the data in the time period $(t_{i-1},\ t_i]$.

We want to select the model m that approximates each segment i, so that the total number N of segments is minimized. Minimizing the number of segments necessarily achieves the best compression ratio, since the model selected at each segment is the cheapest in terms of storage requirements.

There is no known optimal online algorithm for finding the most efficient combination of models. To this end, we propose a greedy algorithm (described in Algorithm 1) for selecting for each segment the model that i) maximizes its length (i.e. approximates the largest number of raw values), and ii) it is the cheapest to be stored. The algorithm achieves this as follows: Consider a set M of models that jointly approximate a certain data segment of the data stream S. Each raw data item $< t, v >$, i.e. with value v at time t, is examined by each of the initialized models for this data segment whether it falls ("hit") or not ("miss") within the error bound ϵ from the estimated data value by the model at time t, i.e. whether $|f_m(t) - v| < \epsilon$ or not for a model m with an instantiated function f_m. All uninitialized models succeed into approximating the raw data item $< t, v >$ in this segment by default. We calculate the compression ratio (i.e. storage cost of uncompressed data over storage cost of model) r_m for each missing model m, and we then exclude m from the models that are further examined against the aforementioned *hitting condition* for this segment. We repeat examining the hitting condition for all subsequent raw data items of the stream, until all models "miss". At this point, the model m^* with the highest compression ratio r_{m*} is dumped into the database to approximate the data segment until the time t_{m*} that it missed. Afterwards, the data stream is retracted to time t_{m*} (i.e. the data stream is considered from time t_{m*} onwards), the approximation formulas of all models are cleared and all models are considered for the approximation of the next data segment according to the aforementioned procedure. If at the time that all models miss, there are several missed models with the same highest compression ratio, then the one with minimum root mean squared error is selected to be dumped in the database. Note that the segment length introduces a certain "latency" between the data generation time and the time that this data becomes available to the application. If a certain data "freshness" is required by the application, then the segment length should be upper-bounded at the expense of the highest achievable compression.

The formulation (1) applies to the approximation of the data stream with *connected* segments. In this case, the approximation of a new segment starts from the ending time of the last segment. When the data stream is approximated by *disconnected* segments, the approximation of a new segment starts at the time of the "miss" for fitting a raw data value.

The optimal offline combination of arbitrary models could be found by Dijkstra algorithm for finding the shortest path in communication networks as follows. The nodes are the data items of the stream, while the models are potential outgoing links at each node. Each model leads to a different ending time for a segment and it has a different storage cost that is used as the weight of the link. Each node is connected to its subsequent node in the stream by a link annotated with the cost of storing a linear segment. We want to find the path across the data items from the starting time of the stream until now, which is expected to achieve the highest compression ratio, i.e. the minimum storage cost. Overall, this algorithm would have complexity $O(|S||M| + |S| \log |S|)$ for a data stream S and a set of models M.

The uncertainty of our greedy algorithm lies on the fact that the selection of a certain model that maximizes the compression ratio for approximating a segment may lead to subsequent increase of the number of segments that approximate the data stream. However, as experimentally proved in Section VI, our greedy algorithm for multi-model approximation always produces fewer or equal segments to the ones produced by the most efficient of the models when individually employed for approximating the data stream.

A. Correctness

Theorem 4.1: The approximate data stream produced by our multi-model fitting algorithm always satisfies the prespecified error bound.

Proof: Assume that a certain raw data item is not approximated within the error bound. Then, there should be a data segment that includes this data item approximated by a certain model that violates the error bound within the segment. However, according to Algorithm 1, a violation of the error bound would lead to the automatic exclusion of the missing model from the considered ones for this data segment. If all approximating models concurrently violated the error bound for this particular raw data item, then the previous data segment would be dumped to the database, while the raw data item would belong to a subsequent segment. Then, if again

Algorithm 1 Multi-model data approximation

Require: Set M of models, stream $S = \{< t_i, v_i >\}$
  $F \leftarrow M$, $F^* \leftarrow \emptyset$
  while $|S| > 0$ **do**
    $< t, v > \leftarrow fetch(S)$
    if $\forall m \in M$, $is\_initialized(m) =$ **false then**
      $t_0 \leftarrow t$
    end if
    for all $m \in F$ **do**
      if $is\_initialized(m) =$ **false then**
        $initialize(m, < t, v >)$
        continue
      end if
      if $|v - f_m(t)| > \epsilon$ **then**
        $F \leftarrow F \setminus \{m\}$, $F^* \leftarrow F^* \cup \{m\}$, calculate r_m
      else
        $t_m \leftarrow t$
      end if
      {check if all models are dropped}
      if $F = \emptyset$ **then**
        $m^* \leftarrow \arg\max_m \{r_m\}$, s.t. $m \in F^*$
        $dump(t_0, t_{m*}, f_{m*})$
        $F \leftarrow M$, $F^* \leftarrow \emptyset$
        for all $m \in F$ **do**
          $clear(m)$
        end for
        $retract(< t^*_m, v^* >, S)$
      end if
    end for
  end while

no model could approximate this raw data item in the new segment, then the raw data item itself would be dumped to the database, i.e. as in lossless approximation. Therefore, there cannot be a raw data item that is not approximated by our algorithm within the pre-specified error bound. ■

V. Storage

Since we are employing the approximation models for compression purposes, only the approximated data segments are stored in the database, instead of the raw values of the data stream. A generic database schema for multi-model approximation consists of one table (*SegmentTable*) for storing the data segments, and a second table (*ModelTable*) for storing the model functions. A tuple of the *SegmentTable* contains the approximation data for a segment in the period [$start\_time$, end_time]. The attribute id stands for identification of the model that is used in the segment. When both linear and non-linear models are employed for the approximation, $left\_value$ is the lowest raw value encountered in the segment and $right\_value$ is the highest raw value encountered in the segment. Also, the attribute $model\_params$ stores the parameters of the instance of the model id that approximates this segment, e.g. the regression coefficients for a regression model. Each tuple in the $ModelTable$ corresponds to a model with a particular id and a certain $function$. The attribute $function$ represents the name of the model and it corresponds to the names of corresponding user defined functions (UDFs) stored in the database. Finally, if the multi-model approximation algorithm is configured to produce connected segments, then the end_time attribute could be omitted from the segment tuple. However, this option would deteriorate the efficiency of the storage scheme for answering time range and point queries, as proper indexing of data segments would be no longer possible. A detailed description of our storage and indexing scheme and its evaluation in terms of response time can be found in [12].

VI. Evaluation

A. Experimental setup

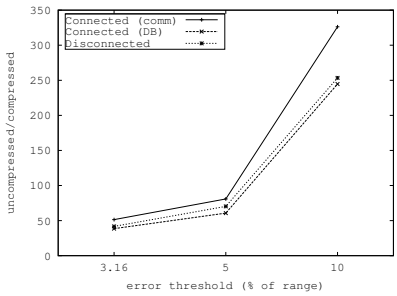
We fully implemented our multi-model approximation algorithm in Java and the proposed storage schema in Oracle 11g (default parameters, 580MB memory). Our experiments were run on a (Core2Duo 2.5GHz CPU, 4GB) machine. In our experiments, both real and synthetic data sets were employed. As real data sets, we used measurements for various environmental parameters (air temperature, humidity, wind direction) collected from existing sensor deployments in the Swiss Alps by the Swiss Experiment project (www.swiss-experiment.ch) and ocean surface temperature sensor data from the TAO project (www.pmel.noaa.gov/tao/). Air temperature, ocean surface and humidity time series have smooth statistical behavior (due to their inherent physical laws), while the wind direction data set is highly *bursty* and quite unpredictable in nature. We also used one synthetic data set: we generated *Lorenz* time series. The Lorenz attractor is a three-dimensional structure corresponding to the long-term behavior of a chaotic flow [13]. We employed as Prantl number $\sigma = 10$, as Rayleigh number $\rho = 28$, and as physical proportion $\beta = \frac{8}{3}$, in the ordinary differential equations of the Lorenz attractor. For the Lorenz synthetic data set, we calculated 10000 Lorenz samples with a step width of 10^{-2} and employed the x-coordinate of the attractor.

All data sets were approximated within 3 different maximum error bounds: 3.16%, 5%, and 10% of the data range. We implemented multiple models including Swing (SW), MidRange (MR), Linear Filter (LF), Linear Regression (LR), Least Squares Line (LS), Constant Filter (CF), and Chebyshev Polynomial with different degrees (referred to as *Cheb* in figures with the degree specified in parenthesis).

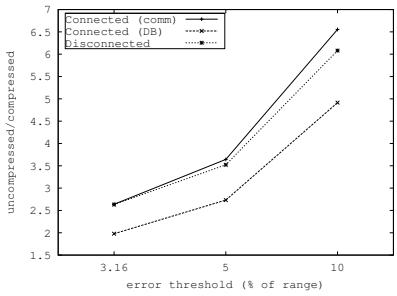
B. Results

a) Connected vs. Disconnected segments: We first assess the effectiveness of our multi-model approximation algorithm for compressing a data stream with *connected* or *disconnected* segments. Recall from Section V that, when connected segments are employed, each data segment could be represented by one attribute less (i.e. end_time is redundant), as compared to the disconnected segments. However, these space savings would come at the cost of inefficient data retrieval in the case

of time-based range queries. Thus, the omission of end_time can be exploited only for data communication compression (referred to as *connected(comm)* in Fig. 2). For fast data retrieval through indexing, the end_time attribute has still to be stored per segment, when the connected segments are employed (referred to as *connected(DB)* in Fig. 2). On the other hand, disconnected segments offer more flexibility to the models for approximating data, as they do not have to start the approximation of a new segment from the ending time of the previous one. In this experiment, we approximate the data combining six different linear models, namely Swing (SW), MidRange (MR), Linear Filter (LF), Linear Regression (LR), Least Squares Line (LS) and Constant Filter (CF). As depicted in Fig. 2, our algorithm with disconnected segments achieves better storage compression than with connected ones both for ocean temperature (Fig. 2(a)) and for wind direction data sets (Fig. 2(b)), and for all the different maximum error bounds considered. The achieved compression improvement increases for more bursty data sets, such as wind direction, as shown in Fig. 2(b). Similar results were obtained for all different data sets considered and for various combinations of both linear and non-linear models. Therefore, disconnected segments are employed by the models for data approximation in the rest of this paper.



(a)



(b)

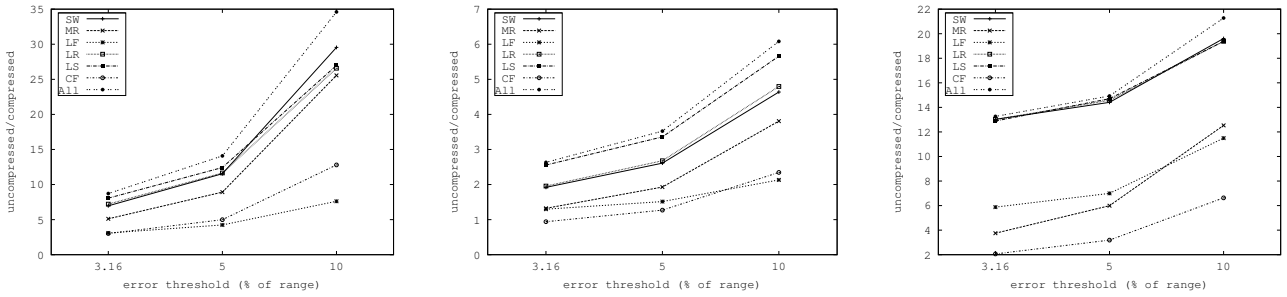
Fig. 2. Compression ratios with connected vs. disconnected segments for a) Ocean Temperature and b) Wind Direction datasets.

b) Multiple linear models: We now assess the compression effectiveness of our multi-model approximation as compared to the compression achieved by its constituent linear models when individually applied to the data stream. Combining linear models is interesting, because they are computationally very efficient, very cheap to store as discussed in Section V, and they achieve comparable effectiveness to their more complex counterparts. As depicted in Figs. 3(a) and 3(b), our multi-model algorithm achieves better compression ratio, than any individual linear model, both for smooth (humidity) and bursty (wind direction) real time series respectively. However, the more bursty the data set, the lower the achievable compression ratio. Also, as shown in Figs. 4(a) and 4(b), our approach indeed produces up to 80% fewer segments against individual models, according to its design objective. The better compression effectiveness of the multi-model approximation algorithm over the individual linear models remains stable for the synthetic data sets, e.g. as depicted in Figs. 3(c) and 4(c) for the Lorenz data set. Moreover, as experimentally proved, different linear models are selected by the multi-model approximation algorithm for approximating the different data segments of the stream. For example, the segments of the humidity stream are approximated 43% by SW, 30% by LS, 19% by MR, 6% by LR and 2% by CF.

c) Combination of Linear and Non-Lineal Models: We finally assess the compression effectiveness of our multi-model approximation algorithm when both linear and non-linear models are combined together. Non-linear models may be more effective to approximate complex data trends of time series, but their segments are also more costly to be stored. As shown in Fig. 5(b), the combination of multiple linear and non-linear models achieves the highest compression ratio for approximating the Lorenz synthetic data set. More interestingly, combining linear and non-linear models for approximating this data set has better compression effectiveness than the combination of only the subset of linear models. In this case, our greedy algorithm achieved to exploit the most of the benefits and avoid most of the weakness of its constituent models. Almost the same result is achieved for the ocean surface time series in Fig. 5(a). The contradiction here is that for maximum error bound 10%, the combination of the subset of linear models achieves slightly better compression ratio than the combination of all models. Still, this result only means that our algorithm is very efficient for finding a combination of models that achieves higher compression than the individual ones; however, it is not guaranteed to find the *optimum* model combination, as expected.

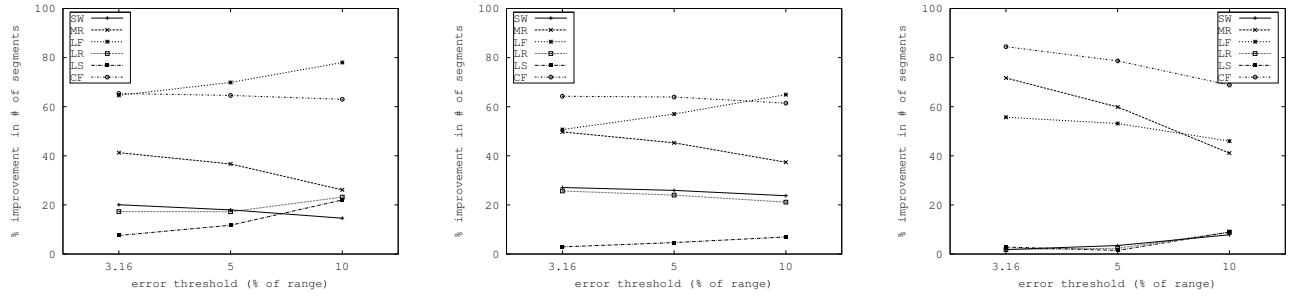
VII. Conclusion

In this paper, we investigated the innovative concept of combining multiple models that approximate time series within a maximum error bound for achieving higher compression effectiveness than the individual models themselves. We proposed a greedy algorithm for finding an efficient model combination for high data compression and experimentally verified its effectiveness for all real and synthetic time series considered. Specifically, as found by the experiments up to 80% compression improvement can be achieved by our algorithm against individual approximation models. As a future work, we intend to theoretically evaluate the effectiveness of our algorithm as compared to the optimal combination of models for approximating the data segments of time series.



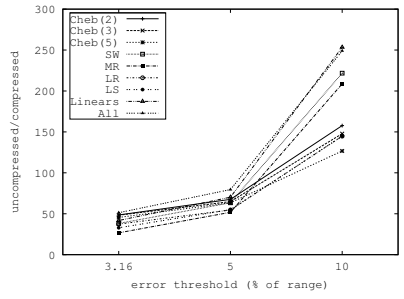
(a) (b) (c)

Fig. 3. Compression ratios for a) Humidity b) Wind Direction and c) Lorenz datasets.

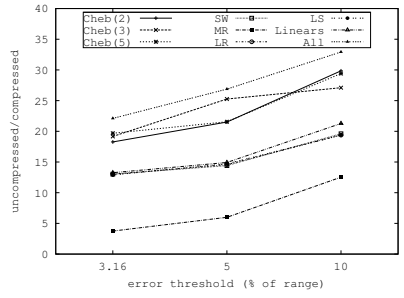


(a) (b) (c)

Fig. 4. Improvement in the number of segments for a) Humidity b) Wind Direction and c) Lorenz datasets.



(a)



(b)

Fig. 5. Compression ratios of combining linear and polynomial models vs. all linear vs. individual models for a) Ocean Temperature and b) Lorenz datasets.

ACKNOWLEDGMENT

This work was funded by the projects OpenSense (NTCH 839_401) and EU HYDROSYS (224416, DG-INFSO).

REFERENCES

[1] H. Elmeleegy, A. K. Elmagarmid, E. Cecchet, W. G. Aref, and W. Zwaenepoel, "Online piece-wise linear approximation of numerical streams with precision guarantees," in *Proc. of VLDB*, Lyon, France, August 2009.

[2] S. Gandhi, S. Nath, S. Suri, and J. Liu, "GAMPS: compressing multi sensor data by grouping and amplitude scaling," in *Proc. of the ACM SIGMOD*, 2009.

[3] A. Deligiannakis, Y. Kotidis, and N. Roussopoulos, "Compressing historical information in sensor networks," in *Proc. of the ACM SIGMOD*, 2004.

[4] T. Palpanas, M. Vlachos, E. Keogh, D. Gunopulos, and W. Truppel, "Online amnesic approximation of streaming time series," in *Proc. of ICDE*, 2004.

[5] I. Lazaridis and S. Mehrotra, "Capturing sensor-generated time series with quality guarantees," in *Proc. of ICDE*, March 2003, pp. 429 – 440.

[6] A. Thiagarajan and S. Madden, "Querying continuous functions in a database system," in *Proc. of the ACM SIGMOD*, 2008.

[7] S. Guha, "On the space—time of optimal, approximate and streaming algorithms for synopsis construction problems," *The VLDB Journal*, vol. 17, no. 6, pp. 1509–1535, 2008.

[8] V. Poosala, P. J. Haas, Y. E. Ioannidis, and E. J. Shekita, "Improved histograms for selectivity estimation of range predicates," in *Proc. of the ACM SIGMOD*, 1996.

[9] E. J. Keogh, S. Chu, D. Hart, and M. J. Pazzani, "An online algorithm for segmenting time series," in *ICDM '01 Proc. of the IEEE International Conference on Data Mining*, 2001.

[10] Y.-A. Le Borgne, S. Santini, and G. Bontempi, "Adaptive model selection for time series prediction in wireless sensor networks," *Signal Process.*, vol. 87, pp. 3010–3020, December 2007.

[11] M. Dalai and R. Leonardi, "Approximations of one-dimensional digital signals under the l^∞ norm," *IEEE Transactions on Signal Processing*, vol. 54, no. 8, pp. 3111–3124, 2006.

[12] T. G. Papaioannou, M. Riahi, and K. Aberer, "Towards online multi-model approximation of time series," Tech. Rep. EPFL-REPORT-164651, http://infoscience.epfl.ch/record/164651, 2011.

[13] E. N. Lorenz, "Deterministic nonperiodic flow," *Journal of the Atmospheric Sciences*, vol. 20, no. 2, pp. 130–141, 1963.