Spatio-temporal Time-Series Forecasting using an Iterative Kernel-Based Regression

Ben Hen[a], Neta Rabin[a]

[a] *Department of Industrial Engineering, Tel Aviv University, Israel*

Abstract

Spatio-temporal time-series analysis is a growing area of research that includes different types of tasks, such as forecasting, prediction, clustering, and visualization. In many domains, like epidemiology or economics, time series data is collected in order to describe the observed phenomenon in particular locations over a predefined time slot and predict future behavior. Regression methods provide a simple mechanism for evaluating empirical functions over scattered data points. In particular, kernel-based regressions are suitable for cases in which the relationship between the data points and the function is not linear. In this work, we propose a kernel-based iterative regression model, which fuses data from several spatial locations for improving the forecasting accuracy of a given time series. In more detail, the proposed method approximates and extends a function based on two or more spatial input modalities coded by a series of multiscale kernels, which are averaged as a convex combination. The proposed spatio-temporal regression resembles ideas that are present in deep learning architectures, such as passing information between scales. Nevertheless, the construction is easy to implement and it is also suitable for modeling data sets of limited size. Experimental results demonstrate the proposed model for solar energy prediction, forecasting epidemiology infections and future number of fire events. The method is compared with well-known regression techniques and highlights the benefits of the proposed kernel-based regression in terms of accuracy and flexibility.

Keywords: kernel regression, spatio-temporal, multi-scale

1. Introduction

Spatio-temporal time-series analysis is a growing area of research. It is applied when data is collected across space and time and describes a phenomenon in a set of particular locations over a predefined time slot [1]. Spatio-temporal data mining research includes different types of tasks, such as forecasting and prediction, clustering, visualization, and its applications span across various domains [2, 3]. Epidemiologists used spatio-temporal data mining techniques to forecast when a disease will outbreak [4, 5], transportation researchers analyzed historical taxi GPS trajectories to recommend fast routes [6], and crime analysts used spatio-temporal patterns from crime event maps to effectively allocate police resources [7]. Nevertheless, these types of problems pose several challenges, including how to represent the data while capturing the relationships in the temporal and spatial domains and how to efficiently process and analyze the represented data.

In this work, we propose a unique and simple forecasting model using spatio-temporal Auto Adaptive Laplacian Pyramids (SALP), a kernel-based data modeling method. The pyramid model is an iterative regression, which models the data in a multi-scale manner while incorporating the spatial connection between nodes (locations). Thus, the SALP is suited to address these types of challenges in spatio-temporal time series forecasting tasks. The proposed model is an extension of a multi-scale regression method, denoted by Auto Adaptive Laplacian Pyramids (ALP), which is described in the following Sections. While ALP is aimed to process single-location data, our suggested spatio-temporal Laplacian Pyramids model incorporates a combination of kernels into the model, each capturing the temporal data from a different location. The model is easy to implement, has a small number of hyper-parameters, and combines the strength of both multi-scale and multi-location connections in the data.

The rest of the paper is organized as follows. Related work on statio-temporal data analysis is described in Section 2. Section 3 describes the Laplacian Pyramids Regression for a single time-series. Section 4 describes our proposed extension of the model for a spatio-temporal setting. Experimental results are provided in Section 5. Finally, a short discussion is given in Section 6.

2. Related Work

The COVID-19 breakout is one recent motivating case in which spatio-temporal models are needed to predict the desease outbreak. Indeed, machine and deep learning models were proposed for this task. In [8], a Random Forest based model was applied for discovering the spread estimation of the daily cases of the COVID-19 outbreak. The model used the number of confirmed cases from 190 countries worldwide from a period of 147 days to predict the number of infected cases one day ahead. The number of confirmed cases was divided into three main sub-datasets, training sub-data, testing sub-data (interpolation data) and estimating sub-data (extrapolation data) for the random forest model. The study resulted with R^2 between 0.843 and 0.995. These results show that Random Forest performs well in estimating the number of cases for the near future in case of an epidemic. Another relevant study applied a multilayer model for early detection of the COVID-19, using spatio-temporal patterns of the disease, resulting with considerably high Area under the Curve of 79.5% [9].

In other domains, the K-Nearest Neighbor (KNN) model was used for short-term traffic multi-step forecasting in [10]. This study proposed an improved KNN model to enhance the forecasting accuracy based on spatio-temporal correlations and to achieve multi-step forecasting. The traffic state of a road segment was described by a spatio-temporal state matrix instead of only a time series as in the original KNN model. The nearest neighbors were selected according to the Gaussian weighted Euclidean distance, which adjusts the influences of time and space factors on spatio-temporal state matrices. In [11], a spatio-temporal dataset was used to improve concentration estimates of air pollution. The data was collected from 112 monitoring stations across 42 days. A 10-fold cross-validation (CV) was used to determine which of the tested machine learning algorithms resulted with the best predictor. The Generalized Boosting Model had the best $CV - R^2$ score.

Deep learning algorithms have also been suggested for spatio-temporal time-series forecasting. A pioneering study [12] offered a newly proposed graph neural network architecture for spatio-temporal signal processing, called Recurrent Graph Convolutional Neural Network (RGCNN). This network can elegantly solve such spatio-temporal tasks with high predictive performance by training a graph convolutional neural network, which is integrated or stacked with a recurrent neural network layer. This technique was applied for time-series analysis and forecasting on an epidemiology dataset, resulting in an average mean squared error of 0.706. The performance of deep Convolutional Neural Networks (CNN) in multi-variate time series forecasting is examined in [13, 14], where a spatial-temporal relation of traffic flow data is represented as images. A CNN model was trained from images in order to forecast the speed in large transportation networks. In [15], the authors studied an image-like representation of spatial time series data using convolution layers and ensemble learning. Moreover, in the presence of temporal data, recurrent neural networks have shown great performance in time series forecasting [16, 17]. The vanishing gradient in deep Multilayer perceptron and recurrent neural network problem was solved by employing a Long-Short Term Model (LSTM) [18], which significantly improves time series forecasting [19, 20]. However, these methods, although leading to state-of-the-art results, are "black-box" methods, thus hard to interpret, and may need careful setting of the training parameters and configuration.

Modeling spatio-temporal time-series as a graph provides the ability to capture the global or local patterns, independent of the data distribution. Functional or correlation networks have been widely applied to map spatio-temporal data into networks. Such a method connects nodes, i.e., time-series, according to their correlation using the correlation coefficient and the maximum cross-correlation [21]. In [22], the authors constructed a directed and weighted network to study the global impacts of climatic phenomena using the heat map of cross-correlations between pairs of events. One drawback of this kind of networks appears when short-length time series are considered, making the statistical significance of correlations questionable and may result in spurious links in the network.

Alternative methods have been proposed to construct networks from time series data. A network-based model, called Chronnet, for spatio-temporal data analysis, was proposed in [23]. The network construction process consists of dividing a geometric space into grid cells represented by nodes connected chronologically. Strong links in the network represent consecutive recurrent events between cells. The Chronnet construction process is fast, making the model suitable to process large data sets. The Visibility algorithm, that converts a spatio-temporal time series into a graph, was proposed in [24]. In this graph, every node corresponds, in the same order, to a series data, and two nodes are connected if there exists visibility between the corresponding data, that is to say, if there is a straight line that connects the series data, provided that this "visibility line" does not intersect any intermediate data height. This network inherits several properties of the time series, and its study reveals non trivial information about the series itself. Although the interest in a network representation is justified by the benefits it provides, such methods demand pre-processing and supervised data in order to learn the connection between nodes.

Kernel-based techniques play a central role in many unsupervised algorithms and have become a common way for describing the local and global relationships of data samples [25, 26]. In addition, kernels-based methods may be incorporated into regression methods, thus, enabling to perform forecasting tasks. The Laplacian Pyramid (LP) regression [27] is a multi-scale model that iteratively generates a smoothed version of a function by using Gaussian kernels of decreasing widths. In order to avoid the risk of overfitting, the Auto-adaptive Laplacian Pyramids (ALP), which is a modified and adaptive version of LP, was illustrated and examined in a radiation forecasting example in [28]. The data was collected between 1994-2009 from 98 sites. Due to the large dimension of the dataset, a non-linear dimensionality reduction method Diffusion maps (DM) [29] was applied on the training set, and then ALP was applied over the constructed DM coordinates. In [30], ALP was utilizes for forecasting of tropical intraseasonal oscillations. ALP was also used as a method for out-of-sample extension ,in particular for extension of embedding coordinates [31, 32, 33]. The LP technique was also extended for imputation tasks [26, 34] and recently for passing information between scales in numeric simulations [35]. In this paper, this line of work is extended to spatio-temporal forecasting tasks.

3. Mathematical Background

3.1. Auto-adaptive Laplacian Pyramids

Auto-adaptive Laplacian Pyramids (ALP) [25] is an iterative kernel-based regression model, suited for capturing the relationship between a set of scattered data points and a target function. Let $X = \{x_i\}_{i=1}^N, \quad x_i \in \mathbb{R}^M$ be the sample dataset. The algorithm approximates a function f defined over X by constructing a series of functions $\{f_0, f_1, f_2, \ldots\}$ obtained by several refinements $\{d_1, d_2, \ldots\}$ over the approximation errors.

In more detail, a first Gaussian kernel with Euclidean distances and a wide initial scale σ is defined by

$$K_0 = k_0(x_i, x_j) = e^{\frac{-\|x_i - x_j\|^2}{\sigma^2}}, \text{ where } x_i, x_j \in X. \tag{1}$$

The smoothing operator P_0 is constructed as the row-stochastic normalized kernel matrix

$$P_0 = p_0(x_i, x_j) = \frac{k_0(x_i, x_j)}{\sum_{x_j \in X} k_0(x_i, x_j)}. \tag{2}$$

A first coarse of f is then generated by

$$s_0(x_i) = \sum_{x_j \in X} P_0(x_i, x_j) f(x_j). \tag{3}$$

This approximation captures the low-frequencies of the function. We denote the first coarse of f as $f_0(x_i) = s_0(x_i)$. The difference $d_1(x_i) = f(x_i) - f_0(x_i) = f(x_i) - s_0(x_i)$ is averaged by a finer kernel P_1, that is constructed with $\sigma = \frac{\sigma}{2}$. This yields with a finer representation of f, $f_1(x_i) = f_0(x_i) + s_1(x_i)$ where $s_1(x_i) = \sum_{x_j \in X} P_1(x_i, x_j) d_1(x_j)$. In general, for $\ell = 1, 2, 3 \ldots$, we have $d_\ell = f - f_{\ell-1}$, and

$$f_\ell(x_i) = f_{\ell-1}(x_i) + s_\ell(x_i) = f_{\ell-1}(x_i) + \sum_{x_j \in X} P_\ell(x_i, x_j) d_\ell(x_j), \tag{4}$$

where

$$P_\ell = p_\ell(x_i, x_j) = \frac{k_\ell(x_i, x_j)}{\sum_{x_j \in X} k_\ell(x_i, x_j)}.$$

K_ℓ is constructed similarly to K_0 in Eq. (1) but with $\sigma = \frac{\sigma}{2^\ell}$.

Extension of the model to new points is straightforward. Given a new point $\tilde{x}$, the multi-scale representations $f_0, f_1, \ldots, f_\ell$ are extended to evaluate $f_\ell(\tilde{x})$. First, s_0 is extended by

$$s_0(\tilde{x}) = \sum_{x_j \in X} P_0(\tilde{x}, x_j) f(x_j), \tag{5}$$

where $P_0(\tilde{x}, x_j)$ is the row-normalized output of $K_0(\tilde{x}, x_j) = e^{\frac{-\|\tilde{x} - x_j\|^2}{\sigma^2}}$. Similarly, the kernels that form the finer resolutions $s_1, \ldots, s_l$ are extended, resulting with

$$f_\ell(\tilde{x}) = f_{\ell-1}(\tilde{x}) + \sum_{x_j \in X} P_\ell(\tilde{x}, x_j) d_\ell(x_j). \tag{6}$$

The iterative train (approximation) algorithm stops once $\text{err}_\ell = \|f - f_\ell\|$ is smaller than a predefined threshold. Since the error of the LP method decays fast, setting a small threshold may easily result in f_ℓ that almost interpolates f, hence overfitting the data [36]. K-fold cross validation is a standard way to prevent overfitting. Samples are randomly distributed in k subsets, and $k - 1$ subsets are used for training while the remaining samples are used for validation. In the extreme case when just one sample is used for validation, cross validation becomes Leave-One-Out Cross Validation (LOOCV). ALP incorporates a modified LOOCV procedure, which makes the method stable and automatic in terms of parameters selection without extra cost. LOOCV can be applied by modifying the previous kernels to have a zero diagonal, for each scale ℓ by setting $K_\ell(x_i, x_i) = 0$. The stopping scale L is set by running the iterations for some predefined number, denoted here by maxIter, computing the mean square error at each iteration and choosing the scale L for which the minimum error value occurs. This results in a series of functions $f_0, f_1, \ldots, f_L$ that approximate f in a multi-scale manner. In [25], it was suggested to set the initial value for σ as $\sigma = 10\max(\mathcal{W}_{ij})$, where $\mathcal{W}_{ij} = \|x_i - x_j\|^2$, holds the Euclidean distances between pairs of data points.

Algorithms 1 and 2 describe the train and test the ALP procedures.

```
Algorithm 1: ALP Train Model
Input: $\{x_i, f(x_i)\}_{i=1}^{n}, \sigma, max_{its}$
Output: Train Model: $f_0(x_i), d_1(x_i), \ldots, d_L(x_i), L -$ stopping scale
Compute $K_0 = \exp(-\|x_i - x_j\|^2/\sigma^2)$, $K_0(x_i, x_i) = 0$.
Normalize $K_0$ and yield $P_0$ (see Eq. (2)).
Compute $s_0(x_i) = \sum_{x_j \in X} P_0(x_i, x_j) f(x_j)$, set $f_0 = s_0$.
$\sigma = \sigma/2, \ell = 1$.
while ($\ell < max_{its}$) do
    Compute $K_\ell = \exp(-\|x_i - x_j\|^2/\sigma^2)$, $K_\ell(x_i, x_i) = 0$.
    Normalize $K_\ell$ and yield $P_\ell$.
    $f_\ell(x_i) = f_{\ell-1}(x_i) + s_\ell(x_i)$, as described in Eq. (4).
    $d_\ell(x_i) = f(x_i) - f_\ell(x_i)$.
    $err_\ell = \|d_\ell\|^2$.
    $\sigma = \sigma/2, \quad \ell = \ell + 1$.
$L \leftarrow argmin_\ell(err_\ell)$, stopping scale.
```

Algorithm 2: ALP Prediction

```
Input: ${x_i, f(x_i)}_{i=1}^n$, ${d_1, d_2, \ldots, d_L}$, $\sigma$, L, test point - $\tilde{x}$
Output: $f_L(\tilde{x})$
$K_0(\tilde{x}, x_j) = e^{\frac{-\|\tilde{x}-x_j||^2}{\sigma^2}}$.
$P_0(\tilde{x}, x_j) = \frac{k_0(\tilde{x},x_j)}{\sum_{x_j \in X} k_0(\tilde{x},x_j)}$.
$f_0(\tilde{x}) = s_0(\tilde{x}) = \sum_{x_j \in X} P_0(\tilde{x}, x_j) f(x_j)$.
$\sigma = \sigma/2$
for $\ell = 1$ to $L$ do
    $K_\ell(\tilde{x}, x_j) = e^{\frac{-\|\tilde{x}-x_j||^2}{\sigma^2}}$.
    $P_\ell(\tilde{x}, x_j) = \frac{k_\ell(\tilde{x},x_j)}{\sum_{x_j \in X} k_\ell(\tilde{x},x_j)}$.
    $f_\ell(\tilde{x}) = f_{\ell-1}(\tilde{x}) + \sum_{x_j \in X} P_\ell(\tilde{x}, x_j) d_\ell(x_j)$.
    $\sigma = \sigma/2, \quad \ell = \ell + 1$.
```

3.2. Auto-adaptive Laplacian Pyramids for Time Series Forecasting

Since this work focuses on time series forecasting applications, we describe the setting that was used for ALP in this work. Given a time series data $y(t)$, where $1 \leq t \leq n$, we wish to make a future prediction, $y(n+1)$ based on short-term trajectories from $y(t)$. Denote the set of overlapping short-term trajectories of length k by $X = \{x(t,:)\}_{t=1}^{n-k+1}$. These are constructed using an overlapping sliding window over $y(t)$. The training set is composed of pairs $\{x(t,:), f(t)\}$, where $f(t) = y(t+k+1)$ is the target. Algorithm 1 is then applied to $\{x(t,:), f(t)\}$ to yield a multi-scale model of $f(t)$. Given a new time-trajectory sample $x(\tilde{t},:)$, the task is to predict $f(\tilde{t})$. The prediction is done by evoking Algorithm 2 on the constructed train model and the new test point $x(\tilde{t},:)$. The predicted value is the output $f_L(\tilde{t})$.

4. Spatio-temporal Laplacian Pyramids

In the previous section, we introduced the ALP algorithm as a forecasting tool for a single data modality, a single time-series. Here, we propose a natural extension of the ALP framework to a spatio-temporal setting. Let $y_1(t), \ldots y_\nu(t)$, $1 \leq t \leq n$, be ν time-series that are captured at ν different spatial locations. The task is to forecast the next value of each time series $y_1(n+1), \ldots, y_\nu(n+1)$. We formulate the data samples from each spatial location to be short overlapping time series, as described in Section 3.2. Denote these ν data sets by $X^{(1)}, X^{(2)}, \ldots, X^{(\nu)}$. The target function for each spatial location is denoted by $f^{(1)}, f^{(2)}, \ldots, f^{(\nu)}$ respectively. In what follows, we will focus on the task of forecasting a single station, for example, $\{X^{(1)}, f^{(1)}\}$ based on the spatial time-series $\{X^{(j)}, f^{(j)}\}_{j=1}^{\nu}$. In other words, we aim to forecast the values of $f^{(1)}$, belonging to the first spatial location, by the historic time-trajectories that are stored in the same location, $X^{(1)}$, and in nearby locations $X^{(2)}, X^{(3)} \ldots X^{(\nu)}$.

The model construction begins by forming ν coarse kernels denoted by $K_0^{(1)}, \ldots, K_0^{(\nu)}$, based on the data sets $X^{(1)}, \ldots, X^{(\nu)}$. The initial corresponding kernel scales are $\sigma^{(1)}, \ldots \sigma^{(\nu)}$. Denote the associated row-normalised kernels by $P_0^{(1)}, \ldots P_0^{(\nu)}$. We consider a new series of kernels, which are formed as a convex combination of ν kernels at a given scale. These are defined by

$$\mathcal{P}_\ell = \alpha_1 P_\ell^{(1)} + \ldots + \alpha_\nu P_\ell^{(\nu)}, \quad \text{where} \quad \sum_{i=1}^{\nu} \alpha_i = 1. \tag{7}$$

For the first level, $\ell = 0$, we have

$$\mathcal{P}_0(x_i^*, x_j^*) = \alpha_1 P_0^{(1)}(x_i^1, x_j^1) + \ldots + \alpha_v P_0^{(\nu)}(x_i^\nu, x_j^\nu). \tag{8}$$

A first coarse approximation of $f^{(1)}$, $f_0^{(1)} = s_0^{(1)}$, is then defined by

$$s_0^{(1)}(x_i^1) = \sum_{x_j^1 \in X^{(1)}} \mathcal{P}_0(x_i^*, x_j^*) f^{(1)}(x_j^1), \text{ where } * \in \{1, 2, \ldots, \nu\}. \tag{9}$$

Then, the residual $d_1^{(1)} = f^{(1)} - f_0^{(1)}$ is smoothed by the linear combinations of the kernels at level $\ell = 1$, as defined in Eq. (7), denoted by $\mathcal{P}_1$.

The iterations are defined by

$$f_\ell^{(1)}(x_i^1) = f_{\ell-1}^{(1)}(x_i^1) + s_\ell^{(1)}(x_i^1), \tag{10}$$

where,

$$s_\ell^{(1)}(x_i^1) = \sum_{x_j^1 \in X^{(1)}} \mathcal{P}_\ell(x_i^*, x_j^*) d_\ell^{(1)}(x_j^1), \text{ where } * \in \{1, 2, \ldots, \nu\} \tag{11}$$

Here, $d_\ell^{(1)} = f^{(1)} - f_{\ell-1}^{(1)}$.

To avoid overfitting, the kernels in the convex combination of Eq. (7), are formed with a 0-diaginal, like described in Section 3.1. Extension to a new point $\tilde{x}^1$ (which is a time-trajectoey) is similar to what is described in Eqs. (5) and (6), when replacing P_0 and P_ℓ with $\mathcal{P}_0$ and $\mathcal{P}_\ell$ (see Eq. (7)).

Figure 1 demonstrates the training stage of the SALP model on the solar energy dataset that will be further detailed in Section 5. The time series in black is the values in one spatial location. The multi-scale approximations in orange use the historic time trajectories of the station to be predicted as well as two other spatial locations.

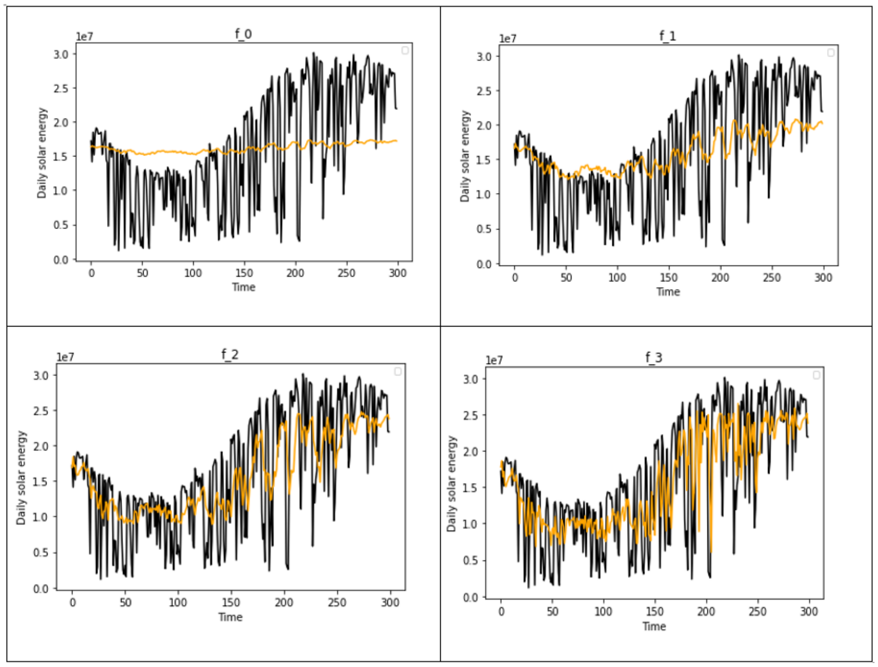

Figure 1: SALP train model. The black time series hold solar energy values in a single location. The multi-scale construction is orange and goes from coarse to fine (f_0 to f_3). The kernels $\mathcal{P}_\ell$ are formed as a linear combination of 3 terms, one is the station to be predicted and two other spatial locations.

4.1. Error Analysis of the Laplacian Pyramids Model

In this section, we review the analysis of the LP model (see [25] for a detailed version), and extend the results to the spatio-temporal setting. To simplify the analysis, we consider the kernels $P_0, P_1, \ldots, P_\ell$ that were defined in Section 3.1, without the 0-diagonal, assuming that the iterations stop at some fine level ℓ. When working in the continuous kernel setting, the summation becomes an integral. Therefore, we have $k_l(x, x')$ for a Gaussian function.

Furthermore, for all ℓ, writing now $p_\ell\,(x) = k_\ell(x, 0)$, is an approximation to a delta function satisfying

$$\begin{aligned} &\int p_l\,(x)\,dx = 1, \qquad \int x p_\ell\,(x)\,dx \quad = 0, \\ &\int \|x\|_2^2\; p_\ell\,(x)\,dx \leq 2C, \end{aligned} \tag{12}$$

where C is a constant. Assume that f is in L_2, then (see [37])

$$\|f_\ell - f\|_{L^2} \leq C\sigma^2 \left(\frac{\sigma^2}{\mu^{(\ell+1)}}\right)^\ell \|f\|_{2\ell+2,2}, \tag{13}$$

where $\|f\|_{m,2}$ denotes the Sobolev norm of a function with up to m derivatives in L_2. Therefore, the L_2 norm of the LP error decays at a very fast rate.

Applying the previous result to the kernel $\sum_i \alpha_i k_\ell^{(i)}$, where $\sum_i \alpha_i = 1$, and defining $f - f_\ell = d_{\ell-1}$ (see Eq. (11)), we have for the convex combinations of the multiple kernels the same bound for the error as in Eq. (13).

4.2. Setting the Values of the Convex Combination

One may consider a graph to model the spatial relationships between the time series. The nodes of the graph are the time-series, and the edge weights α_i from Eq. (7). In this work, we test two options for determining the weights α_i, these are similarities based on correlations and dynamic time wrapping. In both cases, a similarity matrix between the spatial locations is formed based on train samples belonging to the time series $y_1(t), y_2(t), \ldots, y_\nu(t)$. In the first case, the $\nu \times \nu$ matrix entries hold the correlation values. In the second, these hold the result of the dynamic time wrapping (DTW) method. In the following sub-section, DTW is reviewed.

4.2.1. Dynamic Time Warping

Dynamic time warping (DTW) [38] is a technique that finds the optimal alignment between two-time series if one time series may be warped non-linearly by stretching or shrinking it along its time axis. This warping can then be used to determine the similarity between the two time series. This technique is commonly applied in data mining as a distance measure.

Suppose we have two time series, $q_1(t)$ and $q_2(t)$, of length n and m, respectively, where $q_1(t) = q_1(1), \ldots, q_1(n)$ and $q_2(t) = q_2(1), \ldots, q_2(m)$. To align two sequences using DTW, we construct an $n+1$-by-$m+1$ matrix where the (i^{th}, j^{th}) element of the matrix contains the distance $d(q_1(i), q_2(j))$ with the best alignment between the two points $q_1(i)$ and $q_2(j)$. A warping path W is a contiguous set of matrix elements that defines a mapping between q_1 and q_2. The k^{th} element of W is defined as $w_k = (i,j)_k$, so we have $W = w_1, w_2, \ldots, w_k, \ldots, w_K \; max(m,n) \leq K \leq m+n-1$. The warping path is typically subject to several constraints: boundary conditions, continuity, and monotonicity. There are exponentially many warping paths that satisfy the above conditions. However, we are only interested in the path that minimizes the warping cost $DTW(q_1, q_2) = min\sqrt{\sum_{k=1}^{K} w_k}$. This path can be found using dynamic programming. The time and space complexity of DTW $O(nm)$.

5. Experimental Results

Experimental results are demonstrated on three different datasets. The first is the AMS 2013-2014 Solar Energy Prediction Contest[1]. The goal was to predict the total daily incoming solar energy. The data represents the daily aggregated radiation from 98 stations in Oklahoma between 1994-2007. For this experiment, we selected 5 batches of size 98×600, which were converted into an overlapping time series of length 7, for each station, as described at the beginning of Section 4. The model was created based on the first 300 time trajectories, and the remaining 300 trajectories were test points.

The second dataset is the Hungarian Chickenpox[2] dataset, which holds the weekly number of reported chickenpox cases in Hungary. The data was collected between 2005 and 2015 from 20 counties. For this experiment, the model was created based on the first 250 time trajectories, and the remaining 270 trajectories were test points.

The third dataset describes the monthly number of reported fire events given by the Israeli Fire and Rescue Services[3], collected between Jan. 2012 to Sep. 2022 from 7 districts. The model was created based on the $2012-2020$

[1] https://www.kaggle.com/

[2] https://archive.ics.uci.edu/

[3] https://info.data.gov.il/datagov/home/

time trajectories, and the rest were test points. The representation of the data was transformed to be the percentage of change between the current and last month. This type of normalization made it easier to learn from different districts that hold the same temporal pattern, but have different amplitudes. Figure 2 plots the monthly fire events from the 7 districts in Israel.

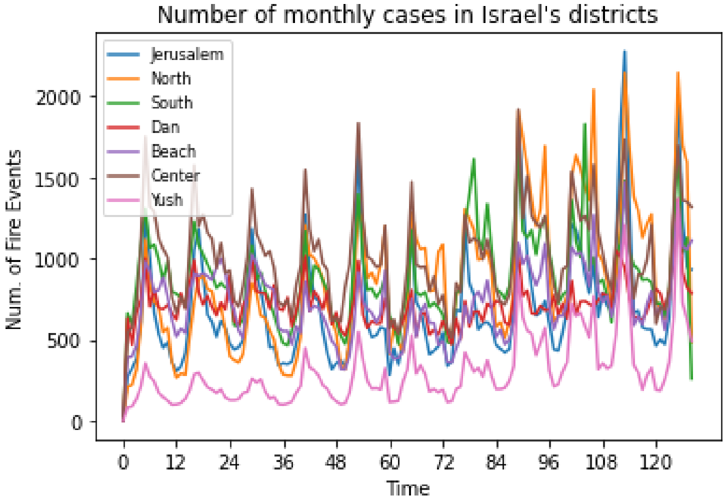

Figure 2: Fire and Rescue dataset from 7 districts

In order to construct the saptio-temporal ALP model (SALP) for forecasting future values of a given time-series $\{X^{(1)}, f^{(1)}\}$, we first identify its most similar spatial locations, as described in Section 4.2, either by correlations or DWT. In this work, we set a constant number of terms for the linear combination of Eq. (7). This number was determined by performing a grid search with $\nu^* \in \{1, 2, 3, 4\}$ for all datasets, where ν^* is the overall optimal number of terms, i.e. number of spatial locations that are considered for the predicting for the single station, $f^1(t)$. Setting $\nu^* = 3$ provided the best results. Denote the two most similar time-series to $\{X^{(1)}\}$, by $\{X^{(2)}\}$ and $\{X^{(3)}\}$. The weights for the linear combination were then set to $\alpha_1 = 0.9, \alpha_2 = \alpha_3 = 0.05$. We note that further analysis may be carried out in order to fine-tune the way ν^* and α_i are chosen, however, even with these fixed weight values, the proposed method yields satisfying results.

The complete train procedure for a given time-series that is represented by short trajectories $\{X^{(1)}(t), f^{(1)}(t)\}$ is evoked by computing the kernels $\mathcal{P}_\ell = 0.9P_\ell^{(1)} + 0.05P_\ell^{(2)} + 0.05P_\ell^{(3)}$, and constructing the ALP model as described in Alg. 1.

In the results tables, we first compare the single-station ALP model to other single station models. These are Kernel Ridge Regression (KRR) with an RBF kernel, Support Vector Regression (SVR), and KNN. These were evoked on train and test data from $\{X^{(1)}(t), f^{(1)}(t)\}$. Then, we show how adding spatial information further improves the results of the ALP model. For the spatio-temporal models, we have three variants of SALP. The first, SALP-C, uses correlations to find the best spatial neighbors. The second, SALP-D, uses DTW to find the spatial neighbors. Another variant we compared to is denoted by SALP-SS, which stands for SALP Single Scale. This model contains several stations as inputs (like the other SALP models). However, we only use one single scale ℓ (a single kernel scale $\frac{\sigma}{2^\ell}$) for the kernels. This yields a saptio-temporal kernel-based regression, but the model doesn't enjoy the benefits of the multi-scale construction. The results were also compared with LSTM and XGBoost. For these two models, we added the same neighbors that were selected by the correlation similarity.

Table 1 presents the average results for the 5 batches of the solar energy dataset in terms of Root Mean Square Error (RMSE) and Mean Absolute Error (MAE) when using a single station. Table 2 presents the errors when using spatial information for 2 additional neighbors. We note that the DWT method (SALP-D) wasn't evoked for this example due to the large number of stations that resulted in a long running time. It can be seen the SALP-C model achieves low errors in both measures.

Table 1: Single-location Prediction Errors for the Solar Dataset

	KNN	**KRR**	**SVR**	**ALP**
RMSE	5,634,483.77	5,363,311.70	8,082,705.95	3,066,830.90
MAE	4,496,800.29	4,054,144.3	6,966,131.33	2,317,464.91

Table 2: Spatio-temporal Prediction Errors for the Solar Dataset

	XGBoost	**LSTM**	**SALP-SS**	**SALP-C**
RMSE	5,194,383.65	5,103,351.54	3,014,911.16	**2,936,759.79**
MAE	4,151,724.21	3,832,125.55	2,315,745.18	**2,221,837.95**

Figure 3 displays the forecasting results of the ALP and SALP models. The left panel displays two single station models, ALP and KRR with an

RBF kernel. The right panel displays the ALP vs. SALP for the same station. It can be seen that the additional spatial information improves the forecasting, in several test points, as the orange line (SALP) is more accurate than the blue (ALP) line.

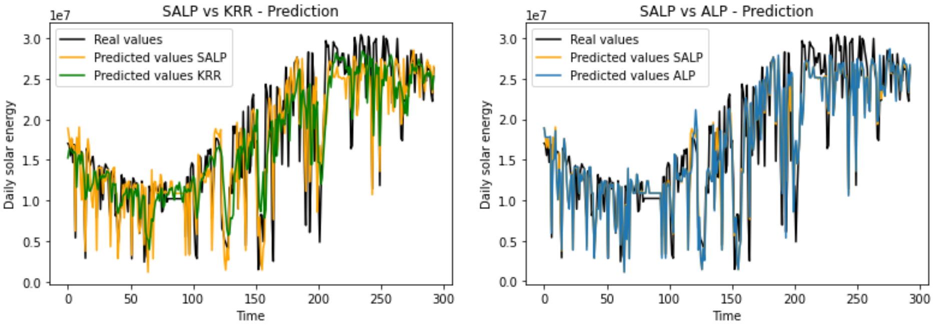

Figure 3: Prediction results for the Solar Energy dataset. Left: Single station models, KRR and SALP. Right: Single station ALP vs. spatio-temporal SALP.

The results for the Chickenpox dataset are displayed in Tables 3 and 4. There are the average RMSE and MAE values across the 20 counties.

Table 3: Single-location Prediction Errors for the Chickenpox Dataset

	KNN	**KRR**	**SVR**	**ALP**
RMSE	24.62	24.77	25.1	15.92
MAE	16.73	16.27	17.55	12.36

Table 4: Spatio-temporal Prediction Errors for the Chickenpox Dataset

	XGBoost	**LSTM**	**SALP-SS**	**SALP-D**	**SALP-C**
RMSE	24.11	22.59	15.98	15.56	**15.33**
MAE	16.29	16.04	21.37	12.07	**11.87**

Last, Tables 5 and 7 display the single-location and spatio-temporal results for the fire and rescue dataset. Recall that the predictions were for the percentage change in fire events from the previous month. It can be seen that ALP achieves low errors for the single station prediction and that these are slightly improved when spatial information is added to the model. In this example, the DTW similarity metric for the selection of spatial neighbors performed better than the correlation-based similarity.

Table 5: Single-location Prediction Errors for the Fire and Rescue Dataset

	KNN	**KRR**	**SVR**	**ALP**
RMSE	0.254	0.291	0.238	0.136
MAE	0.194	0.224	0.181	0.108

Table 6: Spatio-temporal Prediction Errors for the Fire and Rescue Dataset

	XGBoost	**SALP-SS**	**SALP-C**	**SALP-D**
RMSE	0.226	0.216	0.134	**0.133**
MAE	0.174	0.167	0.106	**0.105**

Table 7: Prediction Errors for the solar datadet

Overall, we see that the multi-scale component is important, as small errors are achieved in all of the ALP models. Furthermore, the addition of spatial information further reduces the errors.

6. Discussion

In this paper, we proposed an extension of an iterative, multi-scale regression model to a spatio-temporal setting. The proposed model is appealing since it is easy to implement and yields accurate results due to its multi-scale construction that capture the lower and higher frequencies of the data. Integrating kernels that are formed as convex combinations of data from similar locations, doesn't change the overall train and test algorithms and is shown to improve the prediction results. We emphasize the importance of both the multi-scale and multi-location by comparing the results with the SALP-SS model. This model incorporates spatial information but has only one "layer" of kernels (one convex combination) with a single scale. Our model resembles some characteristics of network models, as information is passed between scales. At the same time, it is explainable and one can understand the relationships between the input and output, as well as analyze the model's convergence rate. The experimental results compare the performance of ALP and SAPL with other well-known regression and machine-learning techniques and highlight the benefits of the proposed kernel-based regression. In future work, we plan to develop data-driven criteria for setting the value of the parameter α in Eq. (7) as well as the number of spatial neighbors ν.

Code Availability

The source code used in this research is available at `https://github.com/benhen96/Spatiotemporal_ALP`

Acknowledgements

This research was supported by the Israel Science Foundation, grant no. [1144/20].

References

[1] K. V. Rao, A. Govardhan, K. C. Rao, Spatiotemporal data mining: Issues, tasks and applications, International Journal of Computer Science and Engineering Survey 3 (1) (2012) 39.

[2] S. Shekhar, Z. Jiang, R. Y. Ali, E. Eftelioglu, X. Tang, V. M. Gunturi, X. Zhou, Spatiotemporal data mining: A computational perspective, ISPRS International Journal of Geo-Information 4 (4) (2015) 2306–2338.

[3] A. Hamdi, K. Shaban, A. Erradi, A. Mohamed, S. K. Rumi, F. D. Salim, Spatiotemporal data mining: a survey on challenges and open problems, Artificial Intelligence Review (2022) 1–48.

[4] P. Elliot, J. C. Wakefield, N. G. Best, D. J. Briggs, et al., Spatial epidemiology: methods and applications., Oxford University Press, 2000.

[5] M. Y. Zhai, R. Lu, W. Jiao, Y. Dan, W. J. Yang, Y. Xu, W. Lin, Epidemiological characteristics and spatiotemporal distribution patterns of human norovirus outbreaks in china, 2012–2018, Biomedical and Environmental Sciences 36 (1) (2023) 76–85.

[6] M. E. Hohn, A. M. Liebhold, L. S. Gribko, Geostatistical model for forecasting spatial dynamics of defoliation caused by the gypsy moth (lepidoptera: Lymantriidae), Environmental Entomology 22 (5) (1993) 1066–1075.

[7] M. R. Leipnik, D. P. Albert, GIS in law enforcement: Implementation issues and case studies, CRC Press, 2002.

[8] C. M. Yeşilkanat, Spatio-temporal estimation of the daily cases of covid-19 in worldwide using random forest machine learning algorithm, Chaos, Solitons & Fractals 140 (2020) 110210.

[9] S. Oved, M. Mofaz, A. Lan, H. Einat, N. Kronfeld-Schor, D. Yamin, E. Shmueli, Differential effects of covid-19 lockdowns on well-being: interaction between age, gender and chronotype, Journal of the Royal Society Interface 18 (179) (2021) 20210078.

[10] P. Cai, Y. Wang, G. Lu, P. Chen, C. Ding, J. Sun, A spatiotemporal correlative k-nearest neighbor model for short-term traffic multistep forecasting, Transportation Research Part C: Emerging Technologies 62 (2016) 21–34.

[11] C. E. Reid, M. Jerrett, M. L. Petersen, G. G. Pfister, P. E. Morefield, I. B. Tager, S. M. Raffuse, J. R. Balmes, Spatiotemporal prediction of fine particulate matter during the 2008 northern california wildfires using machine learning, Environmental science & technology 49 (6) (2015) 3887–3896.

[12] B. Rozemberczki, P. Scherer, O. Kiss, R. Sarkar, T. Ferenci, Chickenpox cases in hungary: a benchmark dataset for spatiotemporal signal processing with graph neural networks, arXiv preprint arXiv:2102.08100 (2021).

[13] X. Ma, Z. Dai, Z. He, J. Ma, Y. Wang, Y. Wang, Learning traffic as images: A deep convolutional neural network for large-scale transportation network speed prediction, Sensors 17 (4) (2017) 818.

[14] K.-H. N. Bui, J. Cho, H. Yi, Spatial-temporal graph neural network for traffic forecasting: An overview and open research issues, Applied Intelligence 52 (3) (2022) 2763–2774.

[15] S. Deng, S. Jia, J. Chen, Exploring spatial–temporal relations via deep convolutional neural networks for traffic flow prediction with incomplete data, Applied Soft Computing 78 (2019) 712–721.

[16] J. T. Connor, R. D. Martin, L. E. Atlas, Recurrent neural networks and robust time series prediction, IEEE transactions on neural networks 5 (2) (1994) 240–254.

[17] B. Huang, K. Ruan, W. Yu, J. Xiao, R. Xie, J. Huang, Odformer: Spatial-temporal transformers for long sequence origin-destination matrix forecasting against cross application scenario, Expert Systems with Applications (2023) 119835.

[18] M. Bukhsh, M. S. Ali, A. Alourani, K. Shinan, M. U. Ashraf, A. Jabbar, W. Chen, Long short-term memory recurrent neural network approach for approximating roots (eigen values) of transcendental equation of cantilever beam, Applied Sciences 13 (5) (2023) 2887.

[19] Z. Zhao, W. Chen, X. Wu, P. C. Chen, J. Liu, Lstm network: a deep learning approach for short-term traffic forecast, IET Intelligent Transport Systems 11 (2) (2017) 68–75.

[20] T. Jia, C. Cai, Forecasting citywide short-term turning traffic flow at intersections using an attention-based spatiotemporal deep learning model, Transportmetrica B: Transport Dynamics 11 (1) (2023) 683–705.

[21] S. Bialonski, M. Wendler, K. Lehnertz, Unraveling spurious properties of interaction networks with tailored random networks, PloS one 6 (8) (2011) e22826.

[22] J. Fan, J. Meng, Y. Ashkenazy, S. Havlin, H. J. Schellnhuber, Network analysis reveals strongly localized impacts of el niño, Proceedings of the National Academy of Sciences 114 (29) (2017) 7543–7548.

[23] L. N. Ferreira, D. A. Vega-Oliveros, M. Cotacallapa, M. F. Cardoso, M. G. Quiles, L. Zhao, E. E. Macau, Spatiotemporal data analysis with chronological networks, Nature communications 11 (1) (2020) 4036.

[24] L. Lacasa, B. Luque, F. Ballesteros, J. Luque, J. C. Nuno, From time series to complex networks: The visibility graph, Proceedings of the National Academy of Sciences 105 (13) (2008) 4972–4975.

[25] Á. Fernández, N. Rabin, D. Fishelov, J. R. Dorronsoro, Auto-adaptive multi-scale laplacian pyramids for modeling non-uniform data, Engineering Applications of Artificial Intelligence 93 (2020) 103682.

[26] N. Rabin, D. Fishelov, Two directional laplacian pyramids with application to data imputation, Advances in Computational Mathematics 45 (4) (2019) 2123–2146.

[27] N. Rabin, R. R. Coifman, Heterogeneous datasets representation and learning using diffusion maps and laplacian pyramids, in: Proceedings of the 2012 SIAM International Conference on Data Mining, SIAM, 2012, pp. 189–199.

[28] Á. Fernández, N. Rabin, D. Fishelov, J. R. Dorronsoro, Auto-adaptive laplacian pyramids., in: ESANN, 2016.

[29] R. R. Coifman, S. Lafon, Diffusion maps, Applied and computational harmonic analysis 21 (1) (2006) 5–30.

[30] R. Alexander, Z. Zhao, E. Székely, D. Giannakis, Kernel analog forecasting of tropical intraseasonal oscillations, Journal of the Atmospheric Sciences 74 (4) (2017) 1321–1342.

[31] D. Lehmberg, F. Dietrich, G. Köster, H.-J. Bungartz, Datafold: Data-driven models for point clouds and time series on manifolds, Journal of Open Source Software 5 (51) (2020) 2283.

[32] G. Mishne, I. Cohen, Multiscale anomaly detection using diffusion maps and saliency score, in: 2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE, 2014, pp. 2823–2827.

[33] M. Li, I. Cohen, S. Mousazadeh, Multisensory speech enhancement in noisy environments using bone-conducted and air-conducted microphones, in: 2014 IEEE China Summit & International Conference on Signal and Information Processing (ChinaSIP), IEEE, 2014, pp. 1–5.

[34] N. Rabin, Multi-directional laplacian pyramids for completion of missing data entries., in: ESANN, 2020, pp. 709–714.

[35] N. Rabin, Á. Fernández, D. Fishelov, Multiscale extensions for enhancing coarse grid computations, Journal of Computational and Applied Mathematics (2023) 115116.

[36] L. Kang, V. R. Joseph, Kernel approximation: From regression to interpolation, SIAM/ASA Journal on Uncertainty Quantification 4 (1) (2016) 112–129.

[37] D. Fishelov, A new vortex scheme for viscous flows, Journal of computational physics 86 (1) (1990) 211–224.

[38] M. Müller, Dynamic time warping, Information retrieval for music and motion (2007) 69–84.