Modeling trend in temperature volatility using generalized LASSO

Anonymous Author(s)
Affiliation
Address
email

Abstract

words, words,

1 Introduction

There is a considerable interest in determining if there is an increasing trend in the climate variability [6, 8]. An increase in the temperature variability will increase the probability of extreme hot outliers. It might be harder for the society to adapt to these extremes than to the gradual increase in the mean temperature [8].

In this project, we consider the problem of detecting the trend in the temperature volatility. All analyses are performed on a sub-set of the European Centre for Medium-Range Weather Forecasts (ECMWF) ERA-40 dataset [15]. This dataset include the temperature measurements over a grid over the earth from 1957 to 2002.

Research on analyzing the trend in the volatility of spatio-temporal data is scarce. [6] studied the change in the standard deviation (SD) of the surface temperature in the NASA Goddard Institute for Space Studies gridded temperature data set. In their analysis, for each geographical position, the mean of the temperature computed for the period 1951-1980 (called the base-period) at that position, is subtracted from the corresponding time series. Each time series is then divided by the standard deviation computed at each position and during the same time period. The distribution of the resulting data is then plotted for different periods. These distributions represent the deviation of the temperature for a specific period, from the mean in the base period, in units of the standard deviation in that period. The results showed that these distributions are widen for the resent time periods compared to 1951-1980. [8] took a similar approach in analysing the ERA-40 data set. However, in addition to the aforementioned method, they computed the distribution of the SDs in an alternative way: for each position and each time period, the deviation of the time-series at that position from the mean in that time period at that position was computed, and then divided by the SD of that position in the period before 1981. The results showed that there still is an increase in the SDs from 1958-1970 to 1991-2001, but this is much less than what is obtained from the method used in [6]. The authors also computed the time-evolving global SD from the de-trended time-series at each position. The resulting curve suggested that the global SD has been stable.

These previous work (and other related research, e.g., [11]) have several shortcomings. First, no statistical analysis has been performed to examine if the change in the SD is statistically significant. Second, the methodologies for computing the SDs are rather arbitrary. The deviation of each time-series in a given period, is computed from either the mean of a base-period (as in [6]), or from the given period (as in [8, 11]). These deviations are then normalized using the SD of the base-period or the given period. No justification is provided for these choices. Third, the correlation between the observations is ignored. The observations in subsequent days and close geographical positions could be highly correlated. Without considering these correlations, any conclusion based on the averaged data could be flawed.

Submitted to 32nd Conference on Neural Information Processing Systems (NIPS 2018). Do not distribute.

The main contribution of this work is to develop a new methodology for detecting the trend in the volatility of spatio-temporal data. In this methodology, the variance at each position and time, is considered as a hidden (un-observed) variable. The value of these hidden variables are then estimated by maximizing the likelihood of the observed data. We show that this formulation per se, is not appropriate for detecting the trend. To overcome this issue, we penalize the differences between the estimated variances of the observations which are temporally and/or spatially close to each other. This will result in an optimization problem called the *generalized LASSO problem* [13]. As we will see, the dimension of this optimization problem is very high and so the standard methods for solving the generalized LASSO cannot be applied directly. We investigate two methods for solving this optimization problem. In the first method, we adopt an optimization technique called alternative direction method of multipliers (ADMM) [4], to divide the total problem into several sub-problems of much lower dimension and show how the total problem can be solved by iteratively solving these sub-problems. The second method, called the *linearized ADMM algorithm* [10] solves the main problem by iteratively solving a linearized version of it. We will compare the benefits of each method.

2 Data Exploration

This section is devoted to exploring some of the properties of the ERA-40 surface temperature data set. The goal here is to demonstrate some of the difficulties in modeling the trend in the temperature volatility and motivate our methodology for doing so.

Figure 1 shows the time-series of the temperature of three cities: Bloomington (USA), San Diego (USA) and Manaus (Brazil). The time-series of Bloomington and San Diego show clear cyclic behavior. However, while it seems (qualitatively) that these cycles can be modeled by a sinusoidal function for Bloomington, the same is not true for San Diego. Also, the amplitude of the cycles changes from some years to others. The time-series of Manaus does not show any regular cyclic behavior. This demonstrates the first difficulty in analyzing the variance of this data: to analyze the variance, we first need to remove the cyclic terms from all time-series. However, there is a lot of variations in the cyclic behavior of the time-series of different locations. In addition, some of these cycles cannot be easily modeled by a parametric function [1]. To overcome these issues, we will use a non-parametric approach to remove the cyclic terms from the time-series and de-trend them. This approach, called *ℓ_1-trend filtering* is explained in the next section.

The right panel of Figure 2 shows the time-series of the temperature of Bloomington, after removing the cyclic terms and de-trending using the method explained in the next section. The goal is to investigate the trend in the variance of this signal. This figure, reveals another issue toward this goal: the variance of this signal, shows cyclic behavior. Also, the cycles are not regular and their amplitude and frequency change. Even if one can describe the behavior of the variance of all the time-series using a single parametric model (for example a variant of the GARCH models [2]), it is not clear how the trend in the variance should be investigated in this framework. These observations motivate the need to develop a non-parametric framework for the problem at hand.

3 ℓ_1-trend filtering for de-trending

ℓ_1-trend filtering was proposed by [9] as a solution to the following problem: a time-series y_t, $t = 1, ..., T$ is observed and we believe that it consists of a smooth trend and a stochastic component and the goal is to estimate the trend. The estimated trend is desired to be smooth, but at the same time, we seek small estimated residuals (stochastic component). These two objectives cannot be achieved simultaneously and so a trade-off between them should be made. In ℓ_1-trend filtering, this trade-off is formulated as the following convex optimization problem:

$$\min_{\beta} \frac{1}{2} \sum_{t=1}^{T} (y_t - \beta_t)^2 + \lambda \sum_{t=1}^{T-2} |\beta_t - 2\beta_{t+1} + \beta_{t+2}| \tag{1}$$

[1] One might try to model the cycles by the summation of sinusoidal terms with different frequencies. However, for some time-series this may need many terms to be included in the summation to achieve a reasonable level of accuracy. In addition, this model cannot capture the non-stationarity in the cycles.

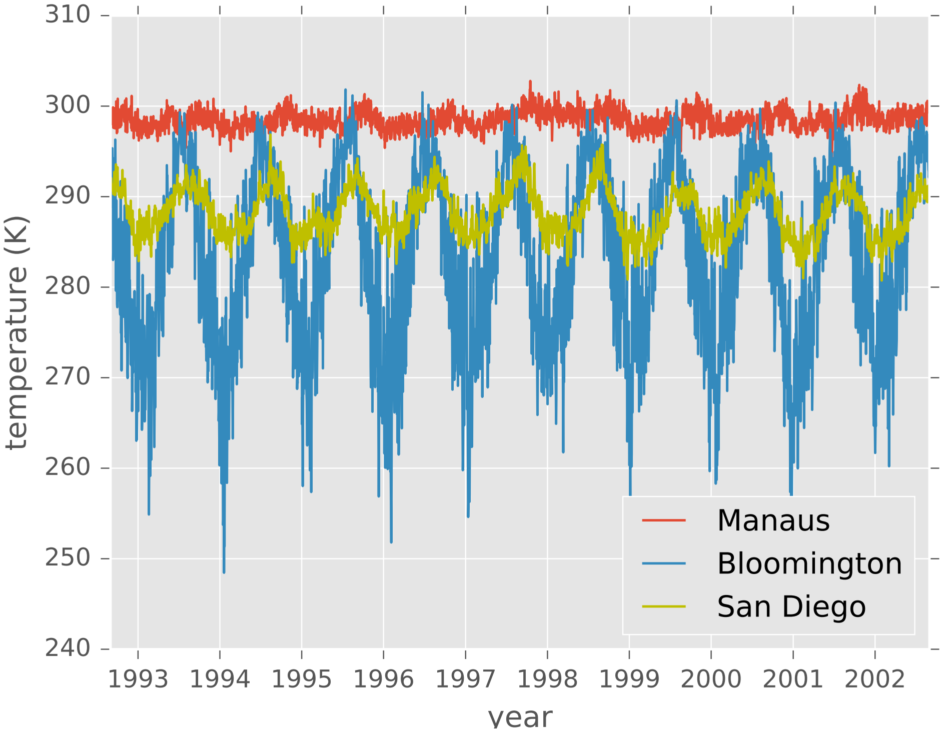

Figure 1: Time-series of the temperature (in Kelvin) of three cities.

or equivalently:

$$\min_{\beta} \frac{1}{2}\|y-\beta\|_2^2+\lambda\|Dh\|_1 \tag{2}$$

where $||u||_1$ is the ℓ_1 norm of the vector u. Here, β is the vector of the free parameters of the model and there is one parameter per observation. The *penalty matrix* is:

$$D = \begin{bmatrix} 1 & -2 & 1 & & & & \\ & 1 & -2 & 1 & & & \\ & & \ddots & \ddots & \ddots & & \\ & & & 1 & -2 & 1 & \\ & & & & 1 & -2 & 1 \end{bmatrix} \tag{3}$$

The first term in (2) penalizes large residuals while the second term encourages smooth estimated β. Specifically, with the penalty matrix (3), the estimated β will be piecewise linear [9]. The trade-off between the two aforementioned objectives is determined by the constant λ. [9] proposed a specialized primal-dual interior point (PDIP) method for solving the optimization problem (2) with the penalty matrix (3).

The results of applying ℓ_1-trend filtering on the Bloomington time-series are shown in Figure 2. The left panel shows the original time-series (blue) together with the estimated β (red) and the right panel shows the residuals $y_t - \beta_t$ (blue). As it can be seen, the estimated βs are piecewise linear. The number of linear segments is determined by the value of λ. Here, we chose $\lambda = 500$. The choice of λ is a model selection problem and we will return to it in the next section ???.

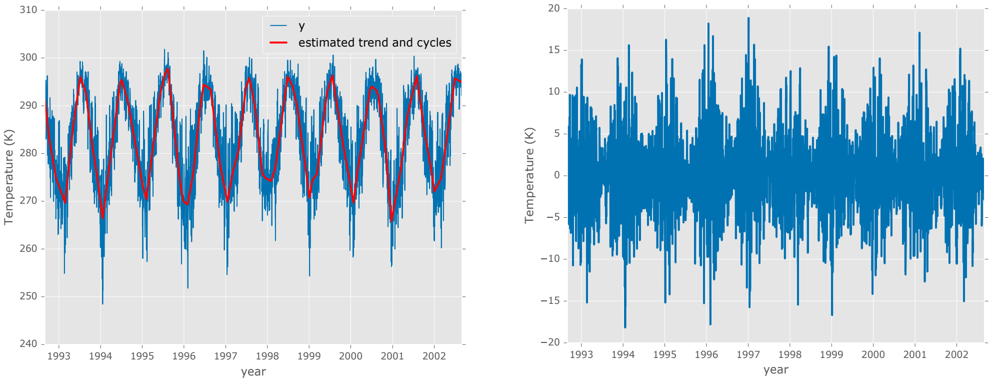

Figure 2: Left: Time-series of the temperature of Bloomington (blue) and the estimated trend and cycles obtained from the ℓ_1-trend filtering (red). Right: the same time-series after removing the cyclic terms and de-trending using ℓ_1-trend filtering.

4 ℓ_1-trend filtering for estimating variance of a time-series

In this section, inspired by the ℓ_1-trend filtering algorithm, we propose a non-parametric model for estimating the variance of a time-series. To this end, we assume that at each time step t, there is a hidden variable h_t such that conditioned on h_t the observations y_t are independent normal variables with zero mean and variance $\exp(h_t)$. The conditional log-likelihood of the observed data in this model is $l(y|h) = c - \sum_{t=1}^{T} h_t - y_t^2 e^{-h_t}$, where c is a constant that does not depend on y and h. Crucially, we assume that the hidden variables h_t vary smoothly. To impose this assumption, we estimate h_t by solving the penalized conditional log-likelihood:

$$\min_h -l(y|h) + \lambda \|Dh\|_1 \tag{4}$$

where D is the same as in (3).

We solve this optimization problem using the PDIP method [2]. First, we note that this is a generalized LASSO problem [13]. The dual of a generalized LASSO with the objective $f(x) + \lambda\|Dx\|_1$ is:

$$\begin{aligned} &\min_\nu f^*(-D^T\nu) \\ &s.t. \ \ \|\nu\|_\infty \leq \lambda \end{aligned} \tag{5}$$

where D^t is the transpose of D, and $f^*(\cdot)$ is the *conjugate* of f defined by: $f^*(u) = \max_x u^t x - f(x)$. It is simple to show that the conjugate of $f(h) = -l(y|h)$ is:

$$f^*(\mathsf{u}) = \sum_t (u_t - 1)\log\frac{y_t^2}{1-u_t} + u_t - 1 \tag{6}$$

The red curve in the left panel of Figure 3 shows the estimated SD (which is $\exp(h_t/2)$) of the residuals of the time-series of Bloomington. To reduce the number of time-steps in this figure and in the remainder of the paper we work on the weekly averaged of the data.

The curve of the estimated SD captures the periodic variations in the SD of the signal. Just by looking at this curve, it is hard to say if the SD is decreasing or increasing. Therefore, we compute the average of the estimated SD for each year. The estimated SD together with this annual average is shown in the middle panel of Figure 3. As it can be seen, the annual trend is not smooth. This is because in the

[2] We use PDIP implemented in the cvxopt Python package [1].

optimization problem (4), the smoothness of the annual trend is not encouraged. To remedy this, we add the following long horizon penalty to (4):

$$\sum_{i=1}^{N_{year}-2} \left| \sum_{t=1}^{52} h_{t_1} - 2h_{t_2} + h_{t_3} \right| \tag{7}$$

where $t_1 = 52(i-1)+t$, $t_2 = 52i + t$ and $t_3 = 52(i+1)+t$. Also, N_{year} is the number of years over which we are performing our analysis (here $N_{year} = 10$). Since we are working on the weekly averaged data, each year corresponds to 52 observations. In the matrix form, the penalty (7) adds N_{year} rows to the matrix D. The estimated SDs using this penalty matrix is shown in the right panel of Figure 3. The annual average of the estimated SDs shows a linear trend with a positive slope.

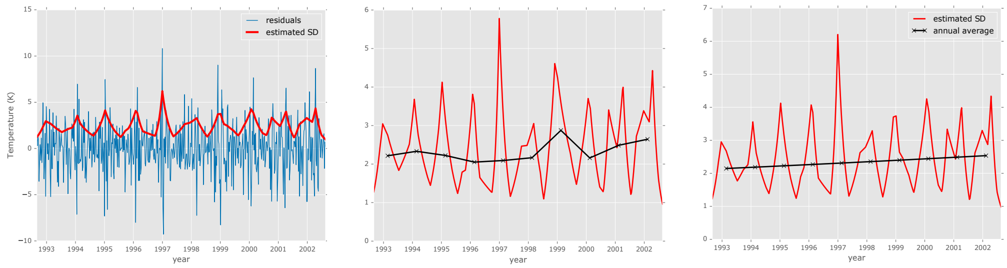

Figure 3: Left: The residuals of the time-series of Bloomington (averaged weekly) and the estimated SD obtained from the method of Section 4 (red). Middle: the estimated SDs (red) and their annual average (black) without imposing the long horizon penalty. Right: the same as middle panel but here the long horizon penalty is imposed. See the text for more details.

5 Extension to spatio-temporal data

The method in the previous section can be used to estimate the variance of a single time-series. In this section, we extend this method to the estimation of the variance of spatio-temporal data.

At a specific time t, the data is measured on a grid of points with n_r rows and n_c columns. Let y_{ijt} denote the value of the observation at time t on the i^{th} row and j^{th} column of the grid, and h_{ijt} denote the corresponding hidden variable. We seek to impose both temporal and spatial smoothness constraints on the hidden variables. Specifically, we seek a solution for h which is piecewise linear in time and piecewise constant in space [3]. This can be achieved by solving the following optimization problem:

$$\begin{aligned} \min_h & \sum_{i,j,t} h_{ijt} + y_{ijt}^2 e^{-h_{ijt}} \\ & + \lambda_t \sum_{i,j} \sum_{t=1}^{T-2} \left| h_{ijt} - 2h_{ij(t+1)} + h_{ij(t+2)} \right| \\ & + \lambda_s \sum_{t,j} \sum_{i=1}^{n_r - 1} \left| h_{ijt} - h_{(i+1)jt} \right| \\ & + \lambda_s \sum_{t,i} \sum_{j=1}^{n_c - 1} \left| h_{ijt} - h_{i(j+1)t} \right| \end{aligned} \tag{8}$$

[3] The assumption that the variance is spatially piecewise constant simplifies the computations. The justification for this assumption is that we are interested in examining the trend in the variance over time and not over space.

The first term in the objective is the negative log-likelihood (minus a constant term). The second term is the temporal penalty for the time-series at each location (i, j). The third and fourth terms, penalize the difference between the estimated variance of two vertically and horizontally adjacent points, respectively. This penalty is a special case of the penalty used in the *trend filtering on graphs* [16] (where the difference between the estimated values of the signal at each two nodes connected with an edge is penalized.). The optimization problem (8) can be written in the matrix form. Let h be a vector whose first T entries are h_{11t} for $t = 1, ..., T$, its next T entries are h_{21t} and so on. Then the optimization problem in the matrix form is as follows:

$$\min_{h} -l(y|h) + \Lambda^t \|D_{total} h\|_1 \tag{9}$$

Let, $\mathbf{e}_n$ denote a vector of size n with all entries being equal to 1. We have: $\Lambda^t = (\lambda_t \mathbf{e}_{n_t}^t | \lambda_s \mathbf{e}_{n_s}^t)$, $D_{total}^t = [D_t^t | D_s^t]$, where n_t and n_s are the number of rows of D_t and D_s, respectively (see below). The matrix D_t is the following block-diagonal matrix and corresponds to the temporal penalty:

$$D_t = \begin{bmatrix} D & & \\ & \ddots & \\ & & D \end{bmatrix} \tag{10}$$

where D was defined in (3). The number of the diagonal blocks is equal to the grid size $n_r \times n_c$. Each row of the matrix D_s corresponds to one spatial constraint in (8). For example, the first T rows correspond to $|h_{11t} - h_{21t}|$ for $t = 1, ..., T$, the next T rows correspond to $|h_{11t} - h_{12t}|$, and so on.

The dual of this problem is:

$$\begin{aligned} &\min_{\nu} f^*(-D_{total}^t \nu) \\ &s.t. \;\; |\nu_i| \leq \Lambda_i \end{aligned} \tag{11}$$

where f^* is given in (6).

6 Optimization

For a spatial grid of size $n_r \times n_c$ and for T time steps, we have $n_t = 3n_r n_c - T n_c - 2n_r n_c$ and $n_s = n_r n_c T$. For a grid over the united states and for weekly averaged data of 10 years we have $n_r = 32$, $n_c = 68$, $T = 521$ and so $n_t \approx 3.5 \times 10^6$ and $n_s \approx 1.0 \times 10^6$. Therefore, the size of the optimization problem (11) is about $n_t \approx 4.5 \times 10^6$. In each step of the PDIP algorithm, we need to solve a linear system of equations in the form $z = Ax$ where A is a square matrix of size $2(n_t + n_s)$ (see [3] equation 11.54). Therefore, applying the PDIP directly for solving the optimization problem (11) is infeasible.

In the next section, we develop an ADMM algorithm for solving this problem. The idea is to cast the problem (11) as a so-called *consensus optimization problem* [4] and solve it by breaking the problem into smaller sub-problems using an ADMM-based algorithm. Next, we first give a brief overview of the consensus optimization problem and then explain how the problem (9) can be solved in this framework.

6.1 Consensus optimization

A general form consensus optimization problem is in the form $\min_z \sum_i f_i(z)$, $z \in \mathbb{R}^n$. We can define a set of *local variables* $x_i \in \mathbb{R}^{n_i}$ such that $\sum_i f_i(z) = \sum_i f_i(x_i)$. We follow the notation of [4] closely. Let $k = \mathcal{G}(i, j)$ which means that the j^{th} entry of x_i is z_k (or $(x_i)_j = z_k$) and define $\tilde{z}_i \in \mathbb{R}^{n_i}$ by $(\tilde{z}_i)_j = (x_i)_j$. Then the original unconstrained optimization problem is equivalent to the following constrained optimization problem:

$$\begin{aligned} &\min_{\{x_1, \ldots, x_N\}} \sum_i f_i(x_i) \\ &s.t. \;\; \tilde{z}_i = x_i \end{aligned} \tag{12}$$

Now, we can apply ADMM to the *augmented Lagrangian* of this problem. This results in the following ADMM updating steps at each iteration m:

$$\begin{aligned} x_i^{m+1} &:= \underset{x_i}{\operatorname{argmin}} \left(f_i(x_i) + (u_i^m)^t x_i + (\rho/2)\|x_i - \tilde{z}_i^m\|_2^2 \right) \\ z_k^{m+1} &:= (1/S_k) \sum_{\mathcal{G}(i,j)=k} (x_i^{m+1})_j \\ u_i^{m+1} &:= u_i^m + \rho(x_i^{m+1} - \tilde{z}_i^{m+1}) \end{aligned} \tag{13}$$

Here, S_k is the number of local variable entries that correspond to z_k, and u_i are the Lagrange multipliers.

To solve the optimization problem (9) or (8) using the method explained in the previous section we need to two address two questions: first, how to choose the local variables x_i, and second, how to solve the optimization problem for updating these variables (the first line of (13)).

Obviously, the global variable in the problem (8) is $z = h$. The global variable can be index by the spatial coordinate and the temporal step. In Figure 4, z is represented as a cube. We can decompose z into sub-cubes. An example of this decomposition is shown in the figure by white lines. It is easy to see that with this definition of x_i, the objective (9) decomposes as $\sum_i f_i(x_i)$ where $f_i(x_i) = -l(y_i|x_i) + \|\Lambda_i^t D_i x_i\|_1$.

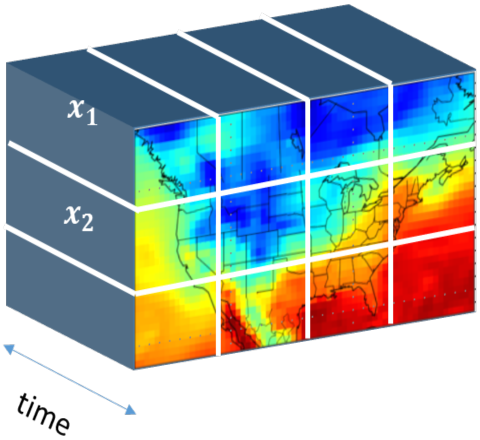

Figure 4: The cube represents the global variable z in space and time. The sub-cubes specified by the white lines are x_i.

By this definition of x_i, the update step for x_i is the following optimization problem: $x_i^{m+1} := \operatorname{argmin}_{x_i}(f_i(x_i) + (u_i^m)^t x_i + (\rho/2)\|x_i - \tilde{z}_i^m\|_2^2)$.

We solve this using the PDIP method. To this end we first need to compute the dual of the problem. It can be shown that the dual of this problem is:

$$\begin{aligned} &\min_{\nu} f_i^*(-D_i^t \nu_i) \\ &s.t.\ \ |(\nu_i)_j| \leq (\Lambda_i)_j \end{aligned} \tag{14}$$

Algorithm 1 ADMM for sparse estimation of variance of spatio-temporal data

```
**Input:** data $y$, mapping $\mathscr{G}(i,j)$, $\rho, \lambda_t, \lambda_s$
Initialization: $x_i^0 = z^0 = u_i^0 = \mathbf{0}$.
**for** $m = 1, 2, ...$ **do**
  **for** $i = 1$ **to** $N_{sub-cubes}$ **do**
    compute $\nu_i$ from (14)
    compute $w_i$ from (15)
    set $x_i^m := w_i$
  **end for**
  Compute $z^m$ from (13)
  Compute $u_i^m$ from (13)
**end for**
```

where:

$$
\begin{aligned}
f_i^*(q) = &\sum_j (q_i)_j (w_i)_j - (w_i)_j - (y_i)_j^2 e^{-(w_i)_j} - \\
&(\rho/2)((w_i)_j - (\alpha_i^m)_j)^2 \\
&\alpha_i^m = \hat{z}_i^m - u_i^m \\
(w_i)_j = &\mathscr{W}\left(\frac{(y_i)_j^2}{\rho} \exp\left[\frac{1 - q_j - \rho(\alpha_i^m)_j}{\rho}\right]\right) - \\
&\frac{1 - q_j - \rho(\alpha_i^m)_j}{\rho}
\end{aligned}
\tag{15}
$$

In these equations, $\mathscr{W}$ is the *Lambert function* [5]. To use the PDIP method, we also need to compute the gradient and Hessian of $f_i^*(-D_i^t \nu_i)$. This involves computing the derivatives of the Lambert functions. The primal solution x_i can be obtained from the dual solution ν_i in (14) by setting $x_i = w_i$ where w_i is defined in the last equation of (15).

The complete ADMM algorithm for estimating the variances is represented in 1. All the computations in the three updating steps (13) can be performed in parallel. The number of rows and columns of the sub-cubes should be chosen so that the updating of x_i could be performed in one processor. We choose $3 \times 3 \times 521$ sub-cubes.

This algorithm divides the main optimization problem into several sub-problems and in each iteration solves these sub-problems and then performs a consensus step to make the solutions of these sub-problems agree with each other. These sub-problems can be solved independently which makes this algorithm appealing for parallelization: in each iteration, each of these sub-problems can be sent into a separate computer and then all the solutions can be sent into a single computer which performs the consensus step. Therefore, in each iteration, the computation time will be equal to the time of solving each sub-problem plus the time of communicating the solutions to the master computer and then performing the consensus step. Since each sub-problem is small, with parallelization, the computation time in each iteration will be small. In addition, our experiments with several values of λ_t and λ_s showed that the algorithm converges in few hundreds iterations. Solving each sub-problem on a machine with four 3.20GHz Intel i5-3470 cores takes less than 3 seconds on average, and so for example if we assume that communication time is 10 seconds and the algorithm converges in 300 iterations, with parallelization on $N_{sub-cubes}$ machines, the algorithm will converge in about 1 hour. Assuming that we use $N_{sub-cubes}$ machines and that the convergence rate of the algorithm is independent of the grid size, this time will be independent of the grid size.

If we perform these computations on a single machine, the computation time grows linearly with $N_{sub-cubes}$. For example, for the data in a grid over the united states and using $3 \times 3 \times 521$ sub-cubes each iteration of the algorithm will take about 20 minutes on a single machine and so with 300 iterations it will take several days to converge. Given that we need to compute the solution for several values of the parameters λ_t and λ_s, this computation time is not feasible.

Therefore, this algorithm is only useful if we can parallelize the computation over several machines. In the next section, we describe another algorithm which makes the computation feasible on a single machine.

6.2 Linearized ADMM

In this section, we describe *Linearized ADMM algorithm* [10] which, as we will see, makes the computation on a single machine feasible.

Consider the following optimization problem:

$$\min_x f(x) + g(Dx) \tag{16}$$

where $x \in \mathbb{R}^n$ and $D \in \mathbb{R}^{m \times n}$. Each iteration of the linearized ADMM algorithm for solving this problem has the following form:

$$\begin{aligned} x^{k+1} &:= \mathbf{prox}_{\mu f}\big(x^k - (\mu/\rho)D^T(Dx^k - z^k + u^k)\big) \\ z^{k+1} &:= \mathbf{prox}_{\rho g}\big(Dx^k + u^k\big) \\ u^{k+1} &:= u^k + Dx^{k+1} - z^{k+1} \end{aligned} \tag{17}$$

where $z, u \in \mathbb{R}^m$ and the *proximal operator* is defined as follows:

$$\mathbf{prox}_{\alpha f}(u) = \min_x \ \alpha \cdot f(x) + \frac{1}{2}\|x - u\|^2 \tag{18}$$

This algorithm belongs to the general class of *proximal algorithms* for solving convex optimization problems. For more details about these algorithms see [10].

The optimization problem (8) can be put into the form (16) as follows:

$$\begin{aligned} f(x) &:= \sum_k f_k(x_k) := \sum_k x_k + y_k^2 e^{-x_k} \\ g(z) &:= \sum_l g_l(z_l) := \sum_l \lambda_l |z_l| \\ z &= Dx \end{aligned} \tag{19}$$

where y_k is the k^{th} entry of the vector whose entries are y_{ijt}, and λ_l is the l^{th} entry of the vector $\Lambda^t = (\lambda_t \mathbf{e}_{n_t}^t | \lambda_s \mathbf{e}_{n_s}^t)$ (see Section 5).

To perform the steps in (17), we need to evaluate $\mathbf{prox}_{\mu f}$ and $\mathbf{prox}_{\rho g}$. The proximal algorithms are feasible only if these proximal operators can be evaluated efficiently which, as we show next, is the case for our problem.

Let $\big(\mathbf{prox}_{\mu f}(u)\big)_k$ be the k^{th} entry of $\mathbf{prox}_{\mu f}(u)$. From the *separable sum* property of the proximal operators we have (see [10], section 2.1):

$$\bigg(\mathbf{prox}_{\mu f}(u)\bigg)_k = \mathbf{prox}_{\mu f_k}(u_k) \tag{20}$$

Similarly, evaluating $\mathbf{prox}_{\rho g}$ reduces to evaluating the proximal operators of scalar functions. We have:

$$\mathbf{prox}_{\mu f_k}(u_k) = \min_{x_k} \ \mu\big(x_k + y_k^2 e^{-x_k}\big) + \frac{1}{2}(x_k - u_k)^2 \tag{21}$$

By setting the derivative with respect to x_k of the function to zero we obtain:

$$\mathbf{prox}_{\mu f_k}(u_k) = \mathscr{W}\left(\frac{y_k^2}{\mu}\exp\left(\frac{1-\mu u_k}{\mu}\right)\right) + \frac{1-\mu u_k}{\mu} \tag{22}$$

where $\mathscr{W}$ is the Lambert function.

Next we compute $\mathbf{prox}_{\mu g_l}(u_l)$. The function $\rho\lambda_l|z_l| + 1/2(z_l - u_l)^2$ is not differentiable. However, at the optimal solution we have: $\rho \cdot \lambda_l \cdot \partial\big(|z_l|\big) = u_l - z_l$, where $\partial\big(|z_l|\big)$ is the sub-gradient of $|z_l|$. This results in the following solution:

$$\mathbf{prox}_{\rho g_l}(u_l) = \begin{cases} u_l - \rho\lambda_l & if \quad u_l > \rho\lambda_l \\ 0 & if \quad |u_l| \le \rho\lambda_l \\ u_l + \rho\lambda_l & if \quad u_l < -\rho\lambda_l \end{cases} \tag{23}$$

Therefore, both proximal operators in (17) can be evaluated easily and so we can use the linearized ADMM algorithm to solve the optimization problem (8).

7 Results

As it was mentioned before, the algorithm proposed in Section 6.1 is appropriate only if we parallelize it over multiple machines and so we do not pursue it further here. All the results reported in this section are obtained using the linearize ADMM algorithm of Section 6.2. We applied this algorithm on a subset of the ERA-40 dataset. The data is the 2 meter temperature measured daily at 12 p.m from August 31 of 1992 to 2002. To reduce the noise we first computed the weekly average of this data. To further reduce the size of the data, we will only analyze the data of the locations inside a rectangle extended from (58°,226°) to (22°,302°). This rectangle covers the united states. All the computations were performed on a machine with four 3.20GHz Intel i5-3470 cores.

7.1 Convergence

Figure 5 shows the value of the loss function (9) as a function of iteration for $\lambda_t = 1$ and $\lambda_s = 0$. We used the following rule to determine when to stop the optimization: the optimization was stopped if the value of the loss did not improve by at least 0.1% in 1000 trials. As we can see, the algorithm converged in about 2000 iterations. This took about 11 minutes. Our experiments showed that the convergence speed depends on the value of λ_t and λ_s. Also, if we use the solution obtained for smaller values of these parameters as the initial value for the larger values (*warm start*), the converges speed improves.

7.2 Model selection

One common method for choosing the penalty parameters in the Lasso problems is to find the solution for a range of the values of these parameters and then choose the values which minimize a model selection criterion. However, such analyses needs the computation of the degrees of freedom (df). Several previous work have investigated the df in generalized lasso problems [7, 14, 17]. However, all these studies have considered the linear regression problem and, to the best of our knowledge, the problem of computing the df for generalized lasso with general objective function has not been considered yet.

Another approach is to choose the set of values which minimize an estimate of the expected prediction error obtained by k-fold cross-validation [12] . Although this method is applicable for our problem, it needs k times more computation.

In this paper, we use a heuristic method for choosing λ_t and λ_s: we compute the optimal solution for a range of values of these parameters and choose the values which minimize $\mathscr{L}(\lambda_t, \lambda_s) = -l(y|h) + \sum\|D_{total}h\|$. This objective is a compromise between the negative log likelihood ($-l(y|h)$) and the complexity of the solution ($\sum\|D_{total}h\|$). For smoother solutions the value of $\sum\|D_{total}h\|$ will be smaller but with the cost of larger $-l(y|h)$.

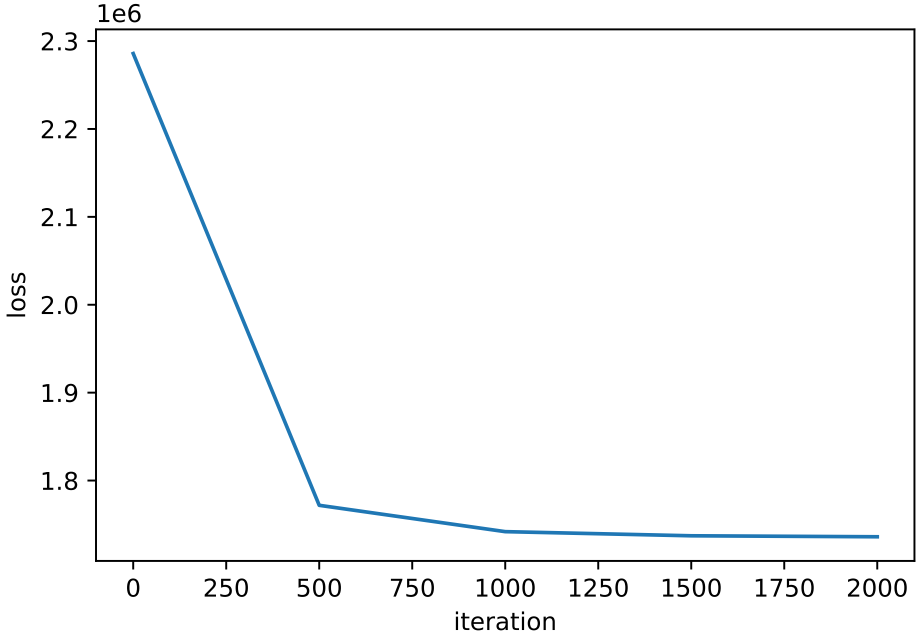

Figure 5: Loss as a function of iteration for $\lambda_t = 1$ and $\lambda_s = 0$.

We computed the optimal solution for all the combinations of the following sets of values: $\lambda_t \in \{1, 5, 10, 20\}$, $\lambda_s \in \{0, .1, 1, 5, 10\}$. The value of $\mathscr{L}(\lambda_t, \lambda_s)$ is shown in Figure 6. This function is minimized at $\lambda_t = 5$ and $\lambda_s = 1$. All the analyses in the next section are performed on the solution for these values.

7.3 Analysis of trend of temperature volatility

Figure 7 shows the detrended data, the estimated standard deviation and the yearly average of these estimates for two cities in the united states: Bloomington (left) and San Diego (right). The estimated SD captures the periodic behavior in the variance of the time-series. In addition, the number of linear segments changes adaptively in each time window depending on how fast the variance is changing.

The yearly average of the estimated SD captures the trend in the temperature volatility. For example, we can see that in Bloomington, there is a small positive trend. To determine how the volatility has changed in each location, we subtract the average of the estimated variance in 1992 from the average in the following years and compute their sum. The value of this change in the variance in each location is depicted in the right panel of Figure 8. The left panel of this figure, shows the average estimated variance in each location.

It is interesting to note that the trend in volatility is almost zero over the oceans. The most positive trend can be observed in the south-east and the most negative trend has happened in the north-east.

Finally, Figure 9 shows the histogram of the changes shown in the right panel of Figure 8. As we can see, there are two peaks in this histogram: one for locations with no trend (which includes the points on the oceans), and one peak for locations with positive trend. The locations with negative trend form the long tail of the histogram.

8 Discussion

In this paper, we proposed a new method for estimating the variance of spatio-temporal data. The main idea is to cast this problem as a constrained optimization problem where the constraints enforce

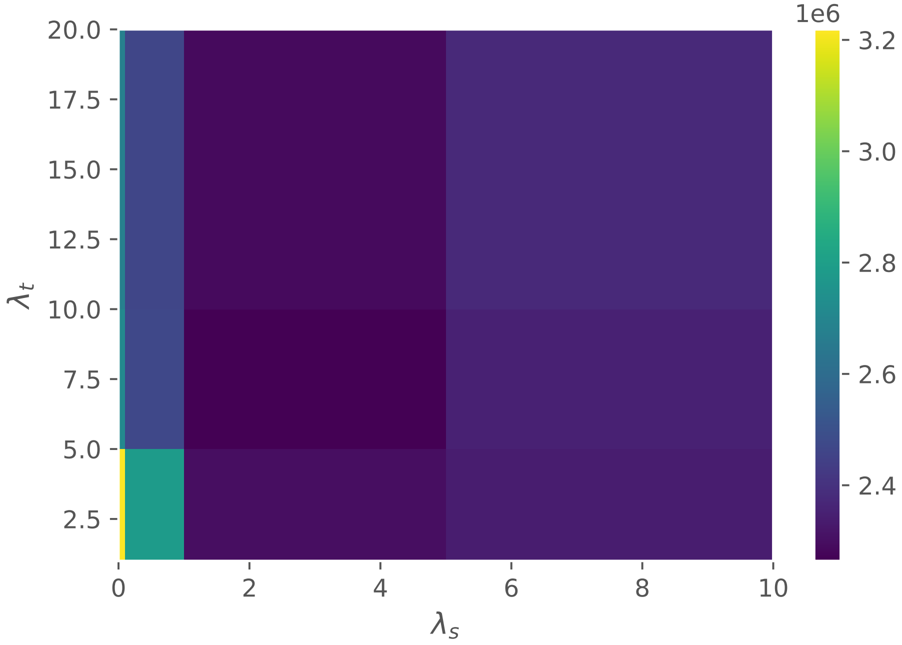

Figure 6: Value of $\mathscr{L}(\lambda_t, \lambda_s)$ for several values of λ_t and λ_s.

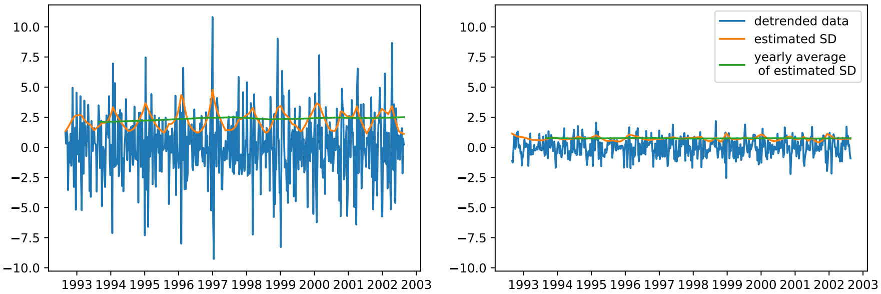

Figure 7: Detrended data and the estimated SD for Bloomington (left) and San Diego (right).

smooth changes in the variance for neighboring points in time and space. In particular, the solution is piecewise linear in time and piecewise constant in space. The resulting optimization is in the form of a generalized LASSO problem with high-dimension, and so applying the PDIP method directly is infeasible. We therefore developed two ADMM-based algorithms to solve this problem: the consensus ADMM and linearized ADMM.

The consensus ADMM algorithm converges in few hundreds of iterations but each iteration takes much longer than the linearized ADMM algorithm. The appealing feature of the consensus ADMM algorithm is that if it is parallelized on enough number of machines the computation time per iteration remains constant as the problem size increases. The linearized ADMM algorithm, on the other hand converges in few thousands of iterations but each iteration is performed in split second. However, since the algorithm converges in many iterations it is not very appropriate for parallelization. The reason is that after each iteration the solution computed in each machine should be broadcast to the master machine and this operation takes some time which depends on the speed of the network

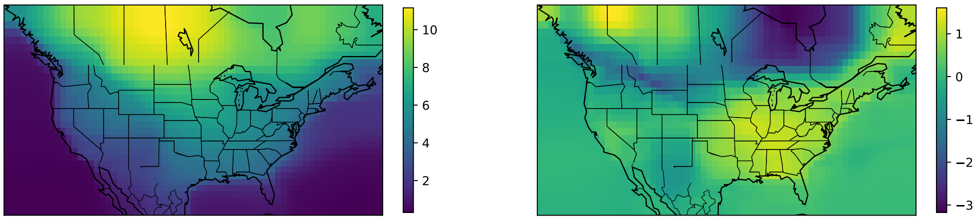

Figure 8: Left: the average of the estimated variance in all locations in the grid over the united states. Right: the change in the variance from 1992 to 2002.

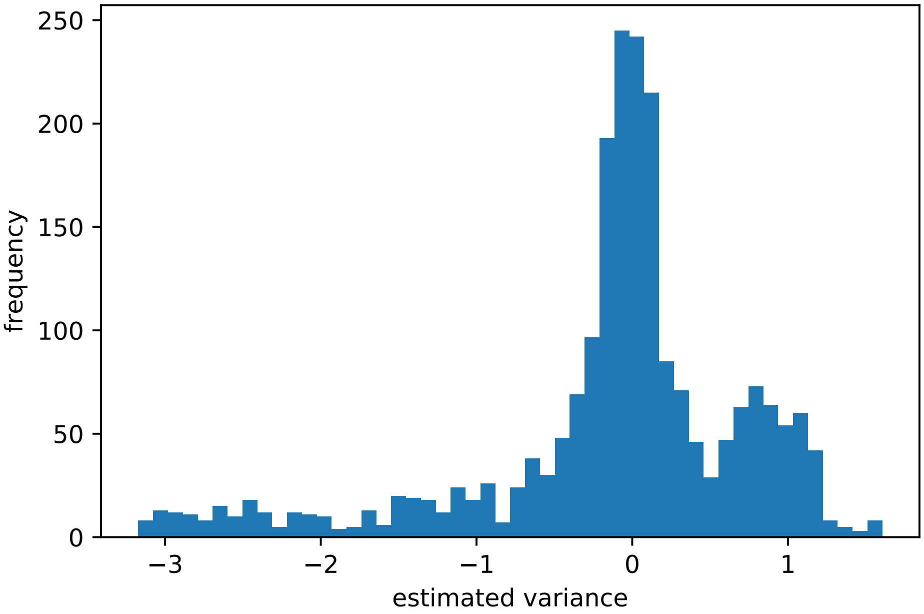

Figure 9: Histogram of the change in the estimated variances.

connecting the slave machines to the master. A direction for future research would be to combine these two algorithms in the following way: the problem should be split into the sub-problems (as in the consensus ADMM) but each sub-problem can be solved using linearized ADMM.

We applied the linearized ADMM algorithm to the surface temperature data on a grid over the united states, for years 1992-2002. The results showed that in many locations the variance of the temperature has increased about 1 unit in 10 years.

The goal of this paper, however, is not to make any conclusions about the trend in the variance because we solved the problem only for a grid over the united states and for 10 years of the data. A thorough analysis, needs the full solution over the globe and for a longer time period. The goal of the paper, was to propose the idea of estimating the trend in variance of spatio-temporal signals using generalized lasso and to investigate the algorithms for solving the resulting optimization problem.

References

[1] M. S. Andersen, J. Dahl, and L. Vandenberghe. CVXOPT: A Python package for convex optimization, version 1.1. 6. *Available at cvxopt. org 54*, 2013.

[2] T. Bollerslev. Generalized autoregressive conditional heteroskedasticity. *Journal of Econometrics*, 31(3): 307–327, Apr. 1986. ISSN 0304-4076.

[3] S. Boyd and L. Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[4] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers. *Foundations and Trends in Machine Learning*, 3(1): 1–122, 2011. ISSN 1935-8237.

[5] R. M. Corless, G. H. Gonnet, D. E. G. Hare, D. J. Jeffrey, and D. E. Knuth. On the LambertW function. *Advances in Computational Mathematics*, 5(1):329–359, Dec. 1996.

[6] J. Hansen, M. Sato, and R. Ruedy. Perception of climate change. *Proceedings of the National Academy of Sciences*, 109(37), Sept. 2012.

[7] Q. Hu, P. Zeng, and L. Lin. The dual and degrees of freedom of linearly constrained generalized lasso. *Computational Statistics & Data Analysis*, 86:13–26, June 2015.

[8] C. Huntingford, P. D. Jones, V. N. Livina, T. M. Lenton, and P. M. Cox. No increase in global temperature variability despite changing regional patterns. *Nature*, 500(7462):327–330, Aug. 2013. ISSN 0028-0836.

[9] S. Kim, K. Koh, S. Boyd, and D. Gorinevsky. ℓ_1 Trend Filtering. *SIAM Review*, 51(2):339–360, May 2009. ISSN 0036-1445. doi: 10.1137/070690274. URL `http://epubs.siam.org/doi/abs/10.1137/070690274`.

[10] N. Parikh and S. Boyd. Proximal Algorithms. *Foundations and Trends® in Optimization*, 1(3):127–239, Jan. 2014.

[11] A. Rhines and P. Huybers. Frequent summer temperature extremes reflect changes in the mean, not the variance. *Proceedings of the National Academy of Sciences*, 110(7):E546–E546, Feb. 2013.

[12] R. Tibshirani. Regression Shrinkage and Selection via the Lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, 58(1):267–288, 1996.

[13] R. J. Tibshirani. *The Solution Path of the Generalized Lasso*. PhD Thesis, Stanford University, 2011.

[14] R. J. Tibshirani and J. Taylor. Degrees of freedom in lasso problems. *The Annals of Statistics*, 40(2): 1198–1232, 2012.

[15] S. M. Uppala, P. W. KÅllberg, A. J. Simmons, U. Andrae, and e. al. The ERA-40 re-analysis. *Quarterly Journal of the Royal Meteorological Society*, 131(612):2961–3012, Oct. 2005.

[16] Y.-X. Wang, J. Sharpnack, A. J. Smola, and R. J. Tibshirani. Trend Filtering on Graphs. *Journal of Machine Learning Research*, 17(105):1–41, 2016. URL `http://jmlr.org/papers/v17/15-147.html`.

[17] P. Zeng, Q. Hu, and X. Li. Geometry and Degrees of Freedom of Linearly Constrained Generalized Lasso. *Scandinavian Journal of Statistics*, 44(4):989–1008, Nov. 2017. ISSN 0303-6898.