Predicting Droughts in the Amazon Basin based on Global Sea Surface Temperatures

Dario Lepke

Contents

Introduction

Placeholder

Chapter 1

Related work

Placeholder

Chapter 2

EDA

Placeholder

2.1 EDA precipitation

2.2 Glyph plots

2.3 EDA SST

Chapter 3

Correlation analysis

Placeholder

3.1 Short Recap

3.2 Correlation of Sea Surface Temperature and Precipitation

3.2.1 Original Data

3.2.1.1 Timelag 0

3.2.1.2 Timelag 3

3.2.1.3 Timelag 6

3.2.1.4 Timelag 12

3.2.2 Deseasonalised Data

3.2.2.1 Timelag 0

3.2.2.2 Timelag 3

3.2.2.3 Timelag 6

3.2.2.4 Timelag 12

3.3 Summary

3.3.1 Original Data

3.3.2 Deseasonalised Data

Chapter 4

Clustering

In this chapter we will first summarize the main ideas of clustering and then apply it to the precipitation data. If not indicated otherwise the information is taken from Elements of Statistical Learning.

4.1 Main Idea Clustering

We can describe an object by a set of measurements or its similarity to other objects. Using this similarity we can put a collection of objects into subgroups or clusters. The objects in the subgroups should then be more similar to one another than to objects of different subgroups. This means inside the clusters we aim for homogeneity and for observations of different clusters for heterogeneity. With the clustering analysis applied to the precipitation data we want to study if there are distinct groups (regions) apparent in the CAB. So that if we later apply the regression models we predict the precipitation for each group and not for the whole region.

To explore the grouping in the data we need a measure of (dis)similarity. This measure is central and depends on subject matter considerations. We construct the dissimilarities based on the measurements taken for each month. We interpret this as a multivariate analysis where, each month is one variable. So given the area in the CAB (resolution 5°x5°), we have 612 cells and 432 months, resulting in a 612×432 data matrix. we want to cluster cells into homogen groups.

4.2 Clustering Methods

4.2.1 K-means

4.2.2 Kmeans characteristics

4.2.3 K-medoids

4.2.3.1 K-medoids characteristics

4.2.4 PCA

4.2.5 Gap statistic

4.3 Analyse clustering results

Chapter 5

LASSO Regression

Placeholder

5.1 The LASSO

5.2 Optimization

5.3 TODO here

Chapter 6

The fused lasso

6.1 General

```
# TODO
# Intro name and motivation fused lasso
# friedmann introduced name
# hastie combination lasso and fusion therefore fused lasso
# we use implementation from genlasso package
# then we describe the formula as given in the package description
# omptimization as in paper
```

```
# TODO add statistical learning with sparsity to references.
```

```
# TODO add to summary of LASSO defaults, that characteristic
# of LASSO
```

```
# TODO make reference to achieved results here
```

```
# TODO cite general LASSO properties here
# how it acts in situations where coefficients are grouped
# or highly correlated.
```

```
# TODO add plot of the graph given by our helper functions.
```

```
# TODO add graphic of fig.2 fused lasso
```

```
#TODO note the fused lasso paper here are
```

```
# TODO add efficient implementations of hte generlaized lasso dual path
# algorithm to references
```

```
# TODO ?check generalized dual path algorithm og paper?
```

```
# Efficient Implementations of the Generalized Lasso Dual Path
# Algorithm
```

As expected and seen in the results, the different LASSO models choose single SST regions as predictors as opposed to whole regions. Since the LASSO only regularizes the magnitude of coefficients but ignores their ordering.
We therefore use the so-called *fused lasso* as implemented in the *genlasso* package and the respective *fusedlasso* function. (R. Tibshirani et al. (2005), Taylor B. Arnold and Tibshirani (2020)) The fused lasso is a generalization of the lasso for problems with features that can be ordered in a meaningful way. It penalizes not only the coefficients' L_1-norm but also their differences given their ordering, introducing sparsity in both of them R. Tibshirani et al. (2005). In our case the fused lasso thus penalizes the differences of SST coefficients that are close to each other.

The fused LASSO as implemented in *genlasso* package solves the problem:

$$\min_{\beta} 1/2 \sum_{i=1}^{n} (y_i - x_i^T \beta_i)^2 + \lambda \sum_{i,j \in E} |\beta_i - \beta_j| + \gamma \cdot \lambda \sum_{i=1}^{p} |\beta_i|, \tag{6.1}$$

with x_i being the ith row of the predictor matrix and E is the edge set of an underlying graph. Regularizing $|\beta_i - \beta_j|$, penalizes large differences in close coefficients. In our case "close" means small distances as defined on 2-dimensional longitude/latitude grid. This grid defines a graph that can be used to compute the distances for each location. The third term $\gamma \cdot \lambda \sum_{i=1}^{p} |\beta_i|$, controls the sparsity of the coefficients. $\gamma = 0$ leads to complete fusion of the coefficients (no sparsity) and $\gamma > 0$ introduces sparsity to the solution, with higher values placing more priority on sparsity. $\hat{\beta}$ is computed as a function of λ, with fixed γ.

6.1.1 Implementation

The summary of the algorithm is taken from the paper proposing the implementation, Taylor B. Arnold and Tibshirani (2016) and the original paper introducing the algorithm R. J. Tibshirani and Taylor (2011). In the fused lasso setting the coefficients $\beta \in \mathbb{R}^p$ can be thought of as nodes of a given undirected Graph

G, with edge set $E \subset 1, ..., p^2$. Now lets assume that E has m edges which are enumerated $e_1, ..., e_m$. The fused lasso penalty matrix D is then $m \times p$, where each row corresponds to an edge in E. So when $e_l = (i, j)$, we write l_{th} row of D as

$$D_l = (0, ..., -1, ..., 1, ...) \in \mathbb{R}^p, \tag{6.2}$$

meaning D_l has all zeros except for the the i_{th} and j_{th} location.

(6.1) is solved by a dual path algorithm that was proposed by Taylor B. Arnold and Tibshirani (2016) for different use cases of the (sparse) fused lasso. They describe the dual path algorithm based on the notation of the generalized lasso problem R. J. Tibshirani and Taylor (2011):

$$\hat{\beta} = \underset{\beta \in \mathbb{R}^p}{\arg\min} \frac{1}{2} \|y - X\beta\|_2^2 + \lambda \|D\beta\|_1, \tag{6.3}$$

where $y \in \mathbb{R}^n$ is the vector of the outcome, $X \in \mathbb{R}^{n \times p}$ a predictor matrix, $D \in \mathbb{R}^{m \times p}$ denotes a penalty matrix, and $\lambda \geq 0$ is a regularization parameter. The dual path algorithm solves not the primal but the dual solution of the problem and computes the solution for a whole path instead of single values of λ. Hence the "dual" and "path" that make up the name. Taylor B. Arnold and Tibshirani (2016) argue that the strength of the original algorithm R. J. Tibshirani and Taylor (2011) lays in the fact that it applies to a unified framework in which D can be a general penalty matrix. Let's consider the case when $X = I$ and $rank(X) = p$ (this is called the "signal approximator" case), the dual problem of (6.3) is then:

$$\hat{u} \in \underset{u \in \mathbb{R}^\omega}{\arg\min} \frac{1}{2} \left\| y - D^T u \right\|_{\frac{2}{2}} \text{ subject to } \|u\|_\infty \leq \lambda. \tag{6.4}$$

The primal and dual solutions, $\hat{\beta}$ and $\hat{u}$ are related by:

$$\hat{\beta} = y - D^T \hat{u}. \tag{6.5}$$

While the primal solution is unique, this does not need to be the case for dual solution (note the element notation in (6.4). The dual path algorithm starts at $\lambda = \infty$ and computes the path until $\lambda = 0$. Conceptually the algorithm keeps track of the coordinates of the dual solutions it computed for each lambda $\hat{u}(\lambda)$. The solutions are equal to $\pm\lambda$, meaning they lie on the boundary of the region $[-\lambda, \lambda]$. Along the path it computes the critical values of λ, $\lambda_1 \geq \lambda_2, ...$, at which the coordinates of these solutions hit or leave the boundary.

There are two algorithms described in the paper and the various specialized implementations that can increase efficiency depending on the use cases. This

depends on X, and/or the special structure of D. Algorithm 1 handles the $X = I$ case and Algorithm 2 the general X case. As we introduced the dual in (6.4), it assumed $X = I$, which is not satisfied in our case. For the general X case the problem formulation can be rewritten so that the formula only changes D and y to $\tilde{D}$ and $\tilde{y}$ and then the same Algorithm can be applied. $\tilde{D} = DX^{+}$ and $\tilde{y} = XX^{+}y$, where X^{x} is the Moore-Penrose pseudoinverse of $X \in \mathbb{R}^{n \times p}$. Algorithm 2 therefore transforms X and y in a certain way and then applies Algorithm 1 to the transformed problem. Its also easy to see that in our case $p > n$ and X is column rank deficient. They solve this by adding a small fixed $l_2 $penalty to the original problem,which leads to:

$$\text{minimize}_{\beta \in \mathbb{R}^p} \frac{1}{2}\|y - X\beta\|_2^2 + \lambda\|D\beta\|_1 + \varepsilon\|\beta\|_2^2, \tag{6.6}$$

and this is the same as

$$\underset{\beta}{\text{minimize}} \frac{1}{2}\left\|y^* - \left(X^*\right)\beta\right\|_2^2 + \lambda\|D\beta\|_1, \tag{6.7}$$

with $y^* = (y, 0)^T$ and $X^* = \begin{bmatrix} x \\ \varepsilon \cdot I \end{bmatrix}$. Because $\text{rank}\left(X^*\right) = p$, again it is possible to apply one of the algorithms.

Arnold, Taylor B., and Ryan J. Tibshirani. 2020. *Genlasso: Path Algorithm for Generalized Lasso Problems.* https://CRAN.R-project.org/package=genlasso.

Arnold, Taylor B, and Ryan J Tibshirani. 2016. "Efficient Implementations of the Generalized Lasso Dual Path Algorithm." *Journal of Computational and Graphical Statistics* 25 (1): 1–27.

Tibshirani, Robert, Michael Saunders, Saharon Rosset, Ji Zhu, and Keith Knight. 2005. "Sparsity and Smoothness via the Fused Lasso." *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 67 (1): 91–108.

Tibshirani, Ryan J, and Jonathan Taylor. 2011. "The Solution Path of the Generalized Lasso." *The Annals of Statistics* 39 (3): 1335–71.