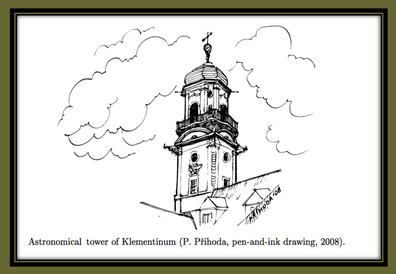
Application of MCMC to Changepoint Detection On Klementinum Temperature Series

Guneykan Ozgul[1], B. Sc. Computer Eng; Kemal Berk Kocabagli[2] B. Sc. Computer Eng
Bogazici University

ABSTRACT

The temperature data coming from Klementinum meteorology station in Prague is well known and many methods have been proposed to find a change point in it. An interesting and different approach to detect such a change point is through MCMC. In this project, we implemented an MCMC algorithm on both the original Klementinum data and a synthetic copy of it that we generated under our proposed model.



Astronomical tower of Klementinum (P. Přihoda, pen-and-ink drawing, 2008).

CONTACT

[2]Kemal Berk Kocabagli,
[1]Guneykan Ozgul
Bogazici University
Email:
kberkkocabagli@gmail.com,
guneykanozgul@gmail.com
Project GitHub link:
https://github.com/berk94/MCMC-Changepoint-Detection-Klementinum

INTRODUCTION

A change point can be defined as the point at which an abrupt change occurs in the generative parameters of a data sequence. Detection of such points is an important problem in various application areas such as finance, biometrics and robotics. It is also a popular subject among statisticians for many years and numerous different techniques are proposed until today.

PROBLEM STATEMENT

Suppose we observe some data that has exactly one noticeable change point, which forms a boundary between two different distributions the data comes from. It is difficult to estimate the parameters of these two distributions and the location of the change point efficiently.

OBJECTIVE

In this project, our objective is to analyze and implement a less usual approach to change point detection, which uses Bayesian Statistics and MCMC to increase efficiency. We modeled the problem as a two-phase linear jump. Our assumption is that the data comes from two different Gaussian distributions with linearly changing mean and a constant variance. The related work can be found in [1].

MODEL

Data points will be denoted by $Z_1, ..., Z_N$ where N is the number of data points and index of the change point will be denoted by r. To simplify results we also used the parameterization $\gamma = {}^{1}/_{\sigma^2}$. We used a two-phase linear model with a jump, assuming exactly one change point.

Our generative model is,

$$Z_i \sim \begin{cases} \mathcal{N}(\alpha_1 + \beta_1 i, 1/\gamma) & 1 \le i \le r \\ \mathcal{N}(\alpha_2 + \beta_2 (i-r), 1/\gamma) & r < i \le N \end{cases} \quad (1)$$

and

$$\mathcal{L}(\alpha_1) \sim \mathcal{N}(\nu_1, \xi_1), \quad \mathcal{L}(\alpha_2) \sim \mathcal{N}(\nu_2, \xi_2), \quad \mathcal{L}(r) \sim \mathcal{R}(1, ..., N)$$

$$\mathcal{L}(\beta_1) \sim \mathcal{N}(\eta_1, \zeta_1), \quad \mathcal{L}(\beta_2) \sim \mathcal{N}(\eta_2, \zeta_2), \quad \mathcal{L}(\gamma) \sim Ga(1, 1)$$

ALGORITHM

Markov chains are generated from the posterior distribution of these parameters using MCMC and desired results are derived from the chains obtained. The formulas for the posterior distributions can be found in [1].

(1) Generate a candidate r' for the new value of the parameter r from $R\{1, ..., N\}$.
(2) Accept the candidate r' from step 1 with a probability $\alpha(\boldsymbol{x}^{(n)}, \boldsymbol{x}')$ that will be specified later; i.e. $r^{(n+1)} = r'$ if accepted else $r^{(n+1)} = r^{(n)}$.
(3) Generate $\alpha_1^{(n+1)}$ and $\alpha_2^{(n+1)}$ from the conditional distribution $f_{\alpha_1,\alpha_2\,|\,\beta_1,\beta_2,\gamma,r,\boldsymbol{z}}$, where the values $\beta_1 = \beta_1^{(n)}$, $\beta_2 = \beta_2^{(n)}$, $\gamma = \gamma^{(n)}$ and $r = r^{(n+1)}$ are given.
(4) Generate $\beta_1^{(n+1)}$ and $\beta_2^{(n+1)}$ from the conditional distribution $f_{\beta_1,\beta_2\,|\,\alpha_1,\alpha_2,\gamma,r,\boldsymbol{z}}$, where the values $\alpha_1 = \alpha_1^{(n+1)}$, $\alpha_2 = \alpha_2^{(n+1)}$, $\gamma = \gamma^{(n)}$ and $r = r^{(n+1)}$ are given.
(5) Generate $\gamma^{(n+1)}$ from the conditional distribution $f_{\gamma\,|\,\alpha_1,\alpha_2,\beta_1,\beta_2,r,\boldsymbol{z}}$, where the values $\alpha_1 = \alpha_1^{(n+1)}$, $\alpha_2 = \alpha_2^{(n+1)}$, $\beta_1 = \beta_1^{(n+1)}$, $\beta_2 = \beta_2^{(n+1)}$ and $r = r^{(n+1)}$ are given.

Iteration Logic

We pick a random change point. If that change point is good enough based on the current estimated parameters, then we keep that r for the next iteration. In addition, we update the alpha, beta and gamma values at each iteration subsequently, fixing all parameters but one at a time.

DATA

Figure 1. Synthetic data created from our proposed model

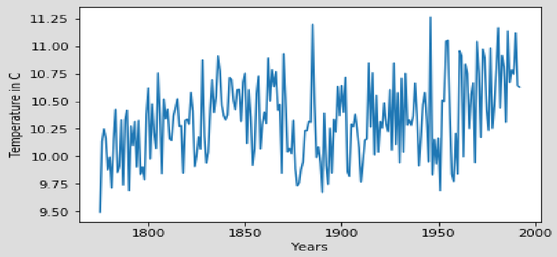
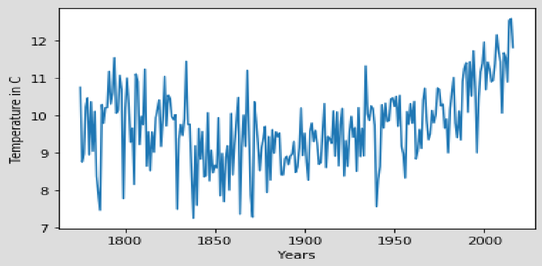


Figure 2. Original data coming from Klementinum station



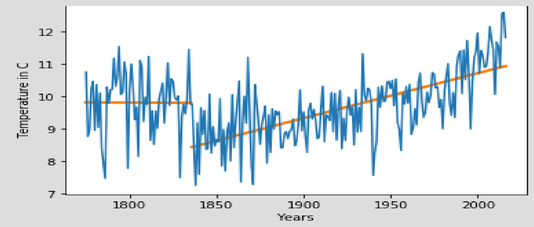
RESULTS

After observing the original Klementinum data, we generated a similar dataset using our proposed model with r=1873, alpha1=alpha2=10, beta1=beta2=0.005, gamma=3.

The following table shows the results that our algorithm converged to. We set a threshold of 40 for convergence. If the change point remains the same for 40 iterations, our algorithm terminates. We did not choose a small number like 10 to avoid convergence to a wrong change point (initialization bias) or a large number like 100 as it might lead our algorithm to disregard a correct change point.

goodEnough=40	alpha1	alpha2	beta1	beta2	gamma	r
Synthetic data	10.09	10.04	0.005	0.005	8.83	1873
Original data	9.81	8.44	-0.0003	0.014	1.43	1836

Figure 3. Prediction of our algorithm on the original dataset



CONCLUSIONS

We tested a couple of synthetic datasets and in the cases where us humans could not detect a noticeable change point, the algorithm failed, too. This occurred, for example, if the end point of the first phase was aligned with the start point of the second phase and the slopes were of the same sign.

Please note that our algorithm might not find the exact same change point and parameters in each run since it runs on Monte Carlo logic. However, the results will be very close.

REFERENCES

1. Jarom´ır Antoch, David Leg´at: Application of MCMC to change point detec- tion. In: Applications of Mathematics, Vol. 53 (2008), No. 4, 281–296.