LINMA 2120 - Seminars in Applied Mathematics
High dimensional time-series prediction with missing values

Quentin Lété

November 2017

1 Introduction

Let us first give a definition of time series to explain the formal context in which we work.

Definition 1. Time series A time series is a sequence of data indexed by the time.

A time series is thus a realization of a stochastic process which can be seen as a family of random variables indexed by the time.

We also define here a property of a stochastic process that will be useful later : the Markow property.

Definition 2. Markov property We say that $(Y_t)_{t\geq 0}$ is a Markov chain if for all $t > 0$

$$\pi(y_t|y_{1:t-1}) = \pi(y_t|y_{t-1})$$

where π is the probability.

This means that all the information about the past is completely carried out in the Y_t at each step. With Markov chains, the joint distribution of observations take a fairly simple form :

$$\pi(y_{1:t}) = \pi(y_1) \cdot \prod_{j=2}^{2} \pi(y_j|y_{j-1})$$

2 Classical methods

In this section, we present classical methods used for time series forecasting. These classical methods include AR (autoregressive) models and DLM (dynamic linear models) and are used for prediction. Let us explain sequentially these models.

Autoregressive models

As explained in [1], the idea of this model is to represent each observation as a noisy linear combination of previous observations. Formally, let $\mathcal{L}$ be the set of time indices that encodes the dependence between the observations over time. If $\mathbf{x}_t \in \mathbb{R}^k$ is the measurement at time t and if the lag $|\mathcal{L}| = p$, the AR(p) model prametrized by the coefficient matrix $\mathcal{W} = \{W^{(l)} \in \mathbb{R}^{k\times k} : l \in \mathcal{L}\}$ can be written as

$$\mathbf{x}_t = \sum_{l\in\mathcal{L}} W^{(l)}\mathbf{x}_{t-l} + \epsilon_t \tag{1}$$

where $\{\epsilon_t\}_{t\in\mathbb{Z}}$ is a Gaussian noise vector with for simplicity $\epsilon_t \sim \mathcal{N}(0, \sigma^2 I_k)$.

This moedel is motivated by a thorem due to Wold that states that a stationary process can be represented by a MA(∞) model, that is

$$\mathbf{x}_t = \sum_{i=1}^{\infty} \beta_i \epsilon_{t-i} + \epsilon_t$$

where $\sum_{i=1}^{\infty} \beta_i < \infty$. And if $\{\mathbf{x}_t\}_t^{\infty}$ is invertible, we also have that it can be represented by an AR(∞) model, that is

$$\mathbf{x}_t = \sum_{i=1}^{\infty} \alpha_i \mathbf{x}_{t-i} + \epsilon_t$$

With this theorem, it seems natural to use AR(p) models for prediction.

Dynamic Linear Models

DLM is a simpler model of a more general framework called state space models.

Definition 3. State space model A state space model consist of two time series : a $\mathbb{R}^p$-valued time series $\{\mathbf{x}_t\}$ and a a $\mathbb{R}^m$-valued time series $\{\mathbf{y}_t\}$ satisfying the following assumptions :

- $(\mathbf{x}_t)$ is a Markov chain
- Conditionnaly on $(\mathbf{x}_t)$, the $\mathbf{y}_t$'s are independent and the $\mathbf{y}_t$'s depends only on $(\mathbf{x}_t)$

The idea behind this model is that the seris $\mathbf{y}_t$ is determined by a latent process $\mathbf{x}_t$. The $\mathbf{x}_t$'s usually represent all the observable physical variables that have an influence on the variable of interest. For instance, if we try to predict the price of electricity in the future, the latent variables could be the temperature, the wind speed, the sunshine, ... Note that this $(\mathbf{x}_t)$ is assumed to be a Markov chain.

In general, a state space model consists in two equations : an **obvservation equation** which gives $\mathbf{y}_t$ in function of $\mathbf{x}_t$ at each step and an **evolution equation** which gives $\mathbf{x}_t$ in function of $\mathbf{x}_{t-1}$. Both equations are pertubed by noise. This can be written as

$$\begin{aligned} \mathbf{y}_t &= f_t(\mathbf{x}_t, v_t) \\ \mathbf{x}_t &= g_t(\mathbf{x}_{t-1}, w_t) \end{aligned}$$

To be complete we also have to specify the prior distribution for $\mathbf{x}_0$.

With that general state space model defined, it is possible to define the DLM which is a special case of it with linear functions and Gaussian noise.

Definition 4. DLM A dynamic linear model (DLM) is specified by a Normal prior distribution for the p-dimensional state vector at time t = 0,

$$\mathbf{x}_0 \sim \mathcal{N}_p(m_0, C_0)$$

together with a pair of equations for each time $t \geq 1$,

$$\begin{aligned} \mathbf{y}_t &= F_t \mathbf{x}_t + v_t, & v_t &\sim \mathcal{N}_m(0, V_t) \\ \mathbf{x}_t &= G_t \mathbf{x}_{t-1} + w_t, & v_t &\sim \mathcal{N}_p(0, W_t) \end{aligned}$$

where F_t and G_t are known matrices of order $m \times p$ and $p \times p$ respectively.

Forecasting

Based on this model, how can we predict the future observation ? The problem is solved by use of the **Kalman filter**. This filter is based on the idea that, due to the markovian property of the state vector and the assumption of conditional independence of the observations, the density of the state and the observation knowing their previous values can be computed recursively. Moreover, because the equations are linear and the noise and prior distribution are Gaussian, it is possible to prove that the random vector $(\mathbf{x}_0, ..., \mathbf{x}_t, \mathbf{y}_0, ..., \mathbf{y}_t)$ follows a mutlivariate Gaussian distribution. Therefore, only the mean and covariance matrix has to be computed. The Kalman filter gives the recursion equation for them :

Theorem 1. *Let $\mathcal{D}_t$ be the information provided by the first t observations $\mathbf{y}_1, ..., \mathbf{y}_t$. Then, if*

$$\mathbf{x}_{t-1} \sim \mathcal{N}(m_{t-1}, C_{t-1}),$$

where $t \geq 1$, then

(a) the one-step-ahead predictive density of $\mathbf{x}_t$ given $\mathcal{D}_{t-1}$ is Gaussian with parameters

$$a_t = E(\mathbf{x}_t|\mathcal{D}_{t-1}) = G_t m_{t-1}$$
$$R_t = Var(\mathbf{x}_t|\mathcal{D}_{t-1}) = G_t C_{t-1} G_t' + W_t$$

(b) the one-step-ahead predictive density of Y_t given $\mathcal{D}_{t-1}$ is Gaussian with parameters

$$f_t = E(\mathbf{y}_t|\mathcal{D}_{t-1}) = F_t a_t$$
$$Q_t = Var(\mathbf{y}_t|\mathcal{D}_{t-1}) = F_t R_t F_t' + V_t$$

(c) the filtering density of $\mathbf{x}_t$ given $\mathcal{D}_t$ is Gaussian with parameters

$$m_t = E(\mathbf{x}_t|\mathcal{D}_t) = a_t + R_t F_t' Q_t^{-1} e_t$$
$$C_t = Var(\mathbf{x}_t|\mathcal{D}_t) = R_t - R_t F_t' Q_t^{-1} F_t R_t$$

where $e_t = \mathbf{y}_t - f_t$ is the forecast error.

This theorem gives us everything we need to predict the behaviour of the observations for a time series modeled by a DLM.

It can be seen that the complexity of this algorithm is $\mathcal{O}(kn^2T + k^3T)$. This rather high complexity can become prohibitive when dealing with high-dimensional data. For instance, it is noted in [2] that using a common package in R for DLM, problems with $n \geq 32$ cannot be solved.

Another problem with the two classical methods that we have described is that it is not clear how missing data would be dealed with. Yet, missing data are common in many application involving time series and can be due to occlusions or misfunction of sensor, among others.

We will now present the novel approach to make predictions about high-dimensional time series published in [2].

3 Temporal Regularized Matrix Factorization

This section aims to describe the method developed in [2].

Framework

The authors place themselves in a framework similar to the state space model defined in definition 3. The difference is that here $\mathbf{x}$ is not supposed to respect the Markov property. Moreover, the observation equation is assumed to be linear. Formally, this can be described by the following equations :

$$\mathbf{y}_t = F\mathbf{x}_t + \eta_t \tag{2}$$

$$\mathbf{x}_t = M_\Theta(\{\mathbf{x}_{t-l} : l \in \mathcal{L}\}) + \epsilon_t \tag{3}$$

The vector noise η_t and ϵ_t are supposed to be Gaussian. These equations are parametrized by sets : $\mathcal{L}$ that contains the number of previous state vectors on wich $\mathbf{x}_t$ depends and Θ that contains the weights associated to each vector in time.

Matrix factorization formulation

The primary obsevation made by the authors is that when we stack all $\mathbf{x}_t$ in a matrix X and if we do the same with the $\mathbf{y}_t$ in the matrix Y, we can see that $Y \approx FX$. Of course, noise has been omitted in this equation and this is why it is not an equality.

With this observation, our problem becomes a problem of matrix factorization which has been widely studied in the past. For instance, this is a key element in some recommender systems and the use of matrix factorization in this context to perform machine learning has been used in the well-known Netflix prize in 2007. In the context of the Netflix prize, the incomplete matrix of ratings was decomposed into a matrix for the users and a matrix for the items. Those two matrices were learned by minimizing an mean square error on the data.

A similar approach can be used in our context. If we denote by $\mathbf{f}_i^\top$ the i$^{\text{th}}$ row of the matrix F and by Ω the set of observed entries, we could learn the X and F factors as

$$\min_{F,X} \sum_{(i,t)\in\Omega} (Y_{it} - \mathbf{f}_i^\top \mathbf{x}_t)^2 + \lambda_f \mathcal{R}_f(F) + \lambda_x \mathcal{R}_x(X) \tag{4}$$

where $\mathcal{R}_f(F)$ and $\mathcal{R}_x(X)$ are regularization factors on F and X respectively. These factors are important to prevent overfitting and to take into accounts possible dependecies between the different entries of the matrices.
The link that has just been established between the general framework and matrix factorization approaches is important but it still does not tell us how to use equation 3 to account for the temporal dependencies. We also need to ddescribe how forecasting could take place in this framework.
In classical MF applications like the one mentionned earlier for recommender systems, the regularization factors are often taken as the norm of the matrix (for example the Frobenius norm) to penalize high values of the larned elements. However, here, this choice would not be a good idea as it would not take into account the temporal dependecies that naturally exists in the $\mathbf{x}_t$s. In [2], the authors propose a choice of regularization factors which remdies that problem.

4 Temporal regularization

The goal in this section is to ddescribe a choice of regulization factor that is called $\mathcal{T}_M(X|\Theta)$ that will promote the relationship between the $\mathbf{x}_t$s given in (3). Recall that Θ in our model contains the information that links the $\mathbf{x}_t$s together. The idea proposed is to take the log-likelihood of the $\mathbf{x}_t$s knowing Θ. This is formalized by the following equation :

$$\mathcal{T}_M(X|\Theta) = -\log \mathbb{P}(\mathbf{x}_1, ..., \mathbf{x}_T|\Theta)$$

It can be seen that the lower this factor is, the more plausible it is to observe this set of $\mathbf{x}_t$s according to (3). Note that the logarithm in this equation is there for computational reasons only. If Θ is given, then we can use directly $\mathcal{T}_M(X|\Theta)$ as the regularization factor for X. When it is not given, a regularization for Θ should be added to 4 which becomes

$$\min_{F,X,\Theta} \sum_{(i,t)\in\Omega} (Y_{it} - \mathbf{f}_i^\top \mathbf{x}_t)^2 + \lambda_f \mathcal{R}_f(F) + \lambda_x \mathcal{T}_M(X|\Theta) + \lambda_\theta R_\theta(\Theta) \tag{5}$$

It is important to note that with this formulation, Θ can thus be learned from the data like F and X. This is another important avantage that TRMF has compared to other approaches like graph-based regularization. Once Θ is learned, we can use it in our model and use equation (3) to predict that next latent vector $\mathbf{x}_t$, which can in turn be used to predict $\mathbf{y}_t$ as $F\mathbf{x}_t$.
Another feature that was missing from the classical method was the possibility to deal missing entries. With the TRMF technique, we can simply estimate the missing entries of Y as $Y_{it} = \mathbf{f}_i^\top \mathbf{x}_t$. This is one big advantage of matrix factirization and it is central to many of its application like recommender systems.
Finally, the MF framework can also be used to perform clustering. Again, this is similar to what has already been proposed in classical use of MF. The $\mathbf{f}_i$ vector can be interpreted as the latent embedding factor of the i^{th} time series in Y and thus clusering algorithms can directly be applied to these vectors.

Extensions

This general TRMF framework can be conveniently extended to account for different aplications dependent extensions. In this section, we prensent 3 useful ones.

Known fetures

In the classical machine leaning framework, every item is associated with a time invariant feature vector. In the study of time series, it is common that in addition to the time dependent data, some known features are available on the objects and it is important to be able to incorporate them in the model to improve performances. Our TRMF framework can be easily extended to account for this additional feature vector for item i denoted $\mathbf{a}_i$ as follows :

$$\min_{F,X,\Theta} \sum_{(i,t)\in\Omega} (Y_{it} - \mathbf{a}_i^\top F\mathbf{x}_t)^2 + \lambda_f \mathcal{R}_f(F) + \lambda_x \mathcal{T}_M(X|\Theta) + \lambda_\theta R_\theta(\Theta) \tag{6}$$

This formulation has two main advantages :

- A new time series $Y_{i't}$ can know be estimated without knowledge of any additional observation up to time T. We can simply and directily estimate it as $Y_{i't} = \mathbf{a}_i'^\top F\mathbf{x}_t$ if the feature vector is available
- If $\mathbf{a}_i \in \mathbb{R}^d$, this has also the effect of reducing the dimensions of F from $n \times k$ to $d \times k$ which can improve the efiiciency of the algorithm.

Graph information among time series

Sometimes, a graph based information is available on the time series. This graph encodes relationship between them. The goal is to define a regularization on F that will tend to promote the structure defined by the graph. Given a graph G on the time series with weights G_{ij} on the edge $i-j$, we can take the following graph regularization factor :

$$\mathcal{G}(F|G,\eta) = \frac{1}{2}\sum_{i\sim j} G_{ij}\|\mathbf{f}_i - \mathbf{f}_j\| + \frac{\eta}{2}\sum_i \|\mathbf{f}_i\|$$

Our problem then becomes simply :

$$\min_{F,X,\Theta} \sum_{(i,t)\in\Omega} (Y_{it} - \mathbf{f}_i^\top \mathbf{x}_t)^2 + \lambda_f \mathcal{G}(F|G,\eta) + \lambda_x \mathcal{T}_M(X|\Theta) + \lambda_\theta R_\theta(\Theta) \tag{7}$$

Temporal-regularized tensor factorization

It is also possible to extend our framework to tensor factorization with evolution over time. To take again the example of recommender system, it is sometimes useful to model the variation over time of the preferences of a user or to model some kind of fashion effect associated to a movie. In the classial MF framework, we thus incorporate time labelled data. If P denotes the matrix of latent enbeddings of the users and Q of the movies, our model can be extended as :

$$\min_{P,Q,X,\Theta} \sum_{(i,j,t)\in\Omega} (Y_{ijt} - \langle \mathbf{p}_i, \mathbf{q}_j, \mathbf{x}_t \rangle)^2 + \lambda_p \mathcal{R}_p(P) + \lambda_q \mathcal{R}_q(Q) + \lambda_x \mathcal{T}_M(X|\Theta) + \lambda_\theta R_\theta(\Theta) \tag{8}$$

where $\langle \mathbf{p}_i, \mathbf{q}_j, \mathbf{x}_t \rangle$ is defined as $\sum_r p_{ir} q_{jr} x_{tr}$

Autoregressive Temporal Regularization

TRMF have been described until here using a very general evolution equation (3). In this section, we will apply the TRMF framework on a auroregressive model for the latent embeddings. Let us recall equation (1) which defines the AR model for the time series $\mathbf{x}_t$ parametrized by the lag set $\mathcal{L}$ and the weights $W^{(l)}$:

$$\mathbf{x}_t = \sum_{l\in\mathcal{L}} W^{(l)} \mathbf{x}_{t-l} + \epsilon_t$$

We would like to define a regularization factor on X that promotes this model for the $\mathbf{x}_t$. A natural choice would be the following :

$$\mathcal{T}_{AR}(X|\mathcal{L}, \mathcal{W}, \eta) = \frac{1}{2} \sum_{t=m}^{T} \left\| \mathbf{x}_t - \sum_{l\in\mathcal{L}} W^{(l)} \mathbf{x}_{t-l} \right\|^2 + \frac{\eta}{2} \sum_t \|\mathbf{x}_t\|^2$$

where $m := 1 + \max(\mathcal{L})$ and the last term has been added to ensure strong convexity.
As explained previously, the TRMF framework allows us to learn the weight matrices $W^{(l)}$ when they are not known. However, as $W^{(l)} \in \mathbb{R}^{k\times k}$, the number of weights to be learned would be $|\mathcal{L}|k^2$ which can be large and lead to overfitting. We could reduce the amount of weights by assuming that the $W^{(l)}$ are diagonal. This reduces the number of weights to $|\mathcal{L}|k$. This correseponds to assuming that each element of the observation vector $\mathbf{y}_t$ is only influenced by the corresponding elements in the observations $\{\mathbf{y}_{t-l}\}_{l\in\mathcal{L}}$

References

[1] O. Anava, E. Hazan, and A. Zeevi, "Online time series prediction with missing data," in *Proceedings of the 32nd International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, F. Bach and D. Blei, Eds., vol. 37. Lille, France: PMLR, 07–09 Jul 2015, pp. 2191–2199. [Online]. Available: http://proceedings.mlr.press/v37/anava15.html

[2] H. Yu, N. Rao, and I. S. Dhillon, "Temporal regularized matrix factorization," *CoRR*, vol. abs/1509.08333, 2015. [Online]. Available: http://arxiv.org/abs/1509.08333