Mini-Project, Times series

High-dimensional vector autoregressive time series modeling via tensor decomposition

Denis Duval, Dorian Gailhard

March 2023

Plan

- Introduction
- Summary of the paper
- Our implementation
- Results

Introduction

- The article focuses on autoregressive models and introduces a novel formalization of them, combining the matrices inside a tensor, which can then benefit from tensor decomposition to restrict the parameter space into three directions. It also adapts regularization methods to this new formalism.
- We implemented the two algorithms introduced by the authors, but only tested the first one.

Summary of the paper

The classical way of formalizing an VAR process problem is the following :

$$y_t = A^1 y_{t-1} + \cdots + A^P y_{t-P} + \epsilon_t, 1 \leq t \leq T$$

with $\{y_t\}$ being the observed time series (with $y_t \in \mathbb{R}^N$, $\{\epsilon_t\}$ being i.i.d innovations of mean 0 and finite variance, also in $\mathbb{R}^N$, and $\{A^i\}_{1 \leq i \leq P}$ being $N \times N$ transitions matrices, which are unknown).

Summary of the paper

The previous problem can be expressed with a tensor combining the transitions matrices, grouping them along a third dimension such that :

$$\forall (i,j,k) \in [1,N] \times [1,N] \times [1,P], \mathcal{A}_{ijk} = A^k_{ij}$$

Denoting (r_1, r_2, r_3) the multilinear ranks of this tensor, there exists the following Tucker decomposition :

$$\mathcal{A} = \mathcal{G} \times_1 \mathcal{U}_1 \times_2 \mathcal{U}_2 \times_3 \mathcal{U}_3$$

where $\mathcal{G} \in \mathbb{R}^{r_1 \times r_2 \times r_3}$ is the core tensor, and $\mathcal{U}_1 \in \mathbb{R}^{N \times r_1}$, $\mathcal{U}_2 \in \mathbb{R}^{N \times r_2}$ and $\mathcal{U}_3 \in \mathbb{R}^{P \times r_3}$ are factor matrices.

Summary of the paper

Ultimately, the VAR model can be rewritten as :

$$y_t = (\mathcal{G} \times_1 \mathcal{U}_1 \times_2 \mathcal{U}_2 \times_3 \mathcal{U}_3)_{(1)} x_t + \epsilon_t$$

where $\mathcal{A}_{(1)}$ is the mode-1 matricization of $\mathcal{A}$.
The mode-1 matricization of $\mathcal{X} \in \mathbb{R}^{p_1 \times p_2 \times p_3}$ is defined such that :

$$\forall 1 \leq i \leq p_1, 1 \leq j \leq p_2, 1 \leq k \leq p_3, \mathcal{X}_{(1)ijk} = \mathcal{X}_{i,(k-1)p_3+j}$$

which is in $\mathbb{R}^{p_1 \times p_2 p_3}$.

MAIN IDEA: Leverage the (HOSVD) Tucker decomposition to reduce the number of parameters

- Classic model: $N^2 P$ parameters
- Proposed model:
  $r_1 r_2 r_3 + (N - r_1) r_1 + (N - r_2) r_2 + (P - r_3) r_3$parameters

Summary of the paper

To estimate the unknown tensors, the authors introduce a multilinear low-rank least squares estimation algorithm which targets the minimization of the following loss :

$$L(\mathcal{G}, \mathcal{U}_1, \mathcal{U}_2, \mathcal{U}_3) = \frac{1}{T} \sum_{t=1}^{T} \| y_t - (\mathcal{G} \times_1 \mathcal{U}_1 \times_2 \mathcal{U}_2 \times_3 \mathcal{U}_3)_{(1)} x_t \|_2^2$$

Summary of the paper

Algorithm 1 Alternating least squares algorithm for $\widehat{\mathcal{A}}_{\text{MLR}}$

Initialize: $\mathcal{A}^{(0)}$

HOSVD: $\mathcal{A}^{(0)} \approx \mathcal{G}^{(0)} \times_1 \boldsymbol{U}_1^{(0)} \times_2 \boldsymbol{U}_2^{(0)} \times_3 \boldsymbol{U}_3^{(0)}$ with multilinear ranks (r_1, r_2, r_3)

repeat $k = 0, 1, 2, \ldots$

$$\boldsymbol{U}_1^{(k+1)} \leftarrow \arg\min_{\boldsymbol{U}_1} \sum_{t=1}^{T} \|\boldsymbol{y}_t - ((\boldsymbol{x}_t'(\boldsymbol{U}_3^{(k)} \otimes \boldsymbol{U}_2^{(k)})\mathcal{G}_{(1)}^{(k)\prime}) \otimes \boldsymbol{I}_N)\text{vec}(\boldsymbol{U}_1)\|_2^2$$

$$\boldsymbol{U}_2^{(k+1)} \leftarrow \arg\min_{\boldsymbol{U}_2} \sum_{t=1}^{T} \|\boldsymbol{y}_t - \boldsymbol{U}_1^{(k+1)}\mathcal{G}_{(1)}^{(k)}((\boldsymbol{X}_t\boldsymbol{U}_3^{(k)})' \otimes \boldsymbol{I}_{r_2})\text{vec}(\boldsymbol{U}_2')\|_2^2$$

$$\boldsymbol{U}_3^{(k+1)} \leftarrow \arg\min_{\boldsymbol{U}_3} \sum_{t=1}^{T} \|\boldsymbol{y}_t - \boldsymbol{U}_1^{(k+1)}\mathcal{G}_{(1)}^{(k)}(\boldsymbol{I}_{r_3} \otimes (\boldsymbol{U}_2^{(k+1)\prime}\boldsymbol{X}_t))\text{vec}(\boldsymbol{U}_3)\|_2^2$$

$$\mathcal{G}^{(k+1)} \leftarrow \arg\min_{\mathcal{G}} \sum_{t=1}^{T} \|\boldsymbol{y}_t - (((\boldsymbol{U}_3^{(k+1)} \otimes \boldsymbol{U}_2^{(k+1)})'\boldsymbol{x}_t)' \otimes \boldsymbol{U}_1^{(k+1)})\text{vec}(\mathcal{G}_{(1)})\|_2^2$$

$$\mathcal{A}^{(k+1)} \leftarrow \mathcal{G}^{(k+1)} \times_1 \boldsymbol{U}_1^{(k+1)} \times_2 \boldsymbol{U}_2^{(k+1)} \times_3 \boldsymbol{U}_3^{(k+1)}$$

until convergence

Finalize: $\widehat{\boldsymbol{U}}_i \leftarrow$ top r_i left singular vectors of $\widehat{\mathcal{A}}_{(i)}$ with positive first elements, $1 \le i \le 3$

$$\widehat{\mathcal{G}} \leftarrow [\![\widehat{\mathcal{A}}; \widehat{\boldsymbol{U}}_1', \widehat{\boldsymbol{U}}_2', \widehat{\boldsymbol{U}}_3']\!]$$

Summary of the paper

- But this algorithm needs the correct multilinear ranks of the tensor,
- which is why the authors propose a simple estimator based on the eigenvalues of an initial estimator.
- The initial estimator is defined as:

$$\widehat{\mathcal{A}}_{\mathrm{NN}} = \arg\min \frac{1}{T}\sum_{t=1}^{T} \|\boldsymbol{y}_t - \mathcal{A}_{(1)}\boldsymbol{x}_t\|_2^2 + \lambda\|\mathcal{A}_{(1)}\|_*.$$

$$\widehat{r}_i = \underset{1\le j\le p_i-1}{\arg\min} \frac{\sigma_{j+1}(\widehat{\mathcal{A}}_{(i)}) + c}{\sigma_j(\widehat{\mathcal{A}}_{(i)}) + c},$$

Summary of the paper

When the problem becomes highly dimensional, a lot of the values of these three matrices become small. It is thus possible to enforce sparsity to reduce the number of unknown parameters and increase interpretability. The authors adapt the classical regularization methods with an algorithm targeting the following loss :

$$L_{sparse}(\mathcal{G}, \mathcal{U}_1, \mathcal{U}_2, \mathcal{U}_3) = L(\mathcal{G}, \mathcal{U}_1, \mathcal{U}_2, \mathcal{U}_3) + \lambda \|\mathcal{U}_1 \otimes \mathcal{U}_2 \otimes \mathcal{U}_3\|_1$$

with the constraints $\mathcal{G} \in \mathbb{R}^{r_1 \times r_2 \times r_3}$, $\mathcal{G}_{(i)}$ is row orthogonal for each i, and $\mathcal{U}_i^T \mathcal{U}_i = I_{r_i}$.

Summary of the paper

The authors then propose the following algorithm to enforce sparsity when estimating the real tensor :

Summary of the paper

Algorithm 2 ADMM algorithm for SHORR estimator

Initialize: $\mathcal{A}^{(0)}$

HOSVD: $\mathcal{A}^{(0)} \approx \mathcal{G}^{(0)} \times_1 \boldsymbol{U}_1^{(0)} \times_2 \boldsymbol{U}_2^{(0)} \times_3 \boldsymbol{U}_3^{(0)}$ with multilinear ranks (r_1, r_2, r_3).

repeat $k = 0, 1, 2, \ldots$

$$\boldsymbol{U}_1^{(k+1)} \leftarrow \underset{\boldsymbol{U}_1'\boldsymbol{U}_1 = \boldsymbol{I}_{r_1}}{\arg\min} \left\{ L(\mathcal{G}^{(k)}, \boldsymbol{U}_1, \boldsymbol{U}_2^{(k)}, \boldsymbol{U}_3^{(k)}) + \lambda \|\boldsymbol{U}_1\|_1 \|\boldsymbol{U}_2^{(k)}\|_1 \|\boldsymbol{U}_3^{(k)}\|_1 \right\}$$

$$\boldsymbol{U}_2^{(k+1)} \leftarrow \underset{\boldsymbol{U}_2'\boldsymbol{U}_2 = \boldsymbol{I}_{r_2}}{\arg\min} \left\{ L(\mathcal{G}^{(k)}, \boldsymbol{U}_1^{(k+1)}, \boldsymbol{U}_2, \boldsymbol{U}_3^{(k)}) + \lambda \|\boldsymbol{U}_1^{(k+1)}\|_1 \|\boldsymbol{U}_2\|_1 \|\boldsymbol{U}_3^{(k)}\|_1 \right\}$$

$$\boldsymbol{U}_3^{(k+1)} \leftarrow \underset{\boldsymbol{U}_3'\boldsymbol{U}_3 = \boldsymbol{I}_{r_3}}{\arg\min} \left\{ L(\mathcal{G}^{(k)}, \boldsymbol{U}_1^{(k+1)}, \boldsymbol{U}_2^{(k+1)}, \boldsymbol{U}_3) + \lambda \|\boldsymbol{U}_1^{(k+1)}\|_1 \|\boldsymbol{U}_2^{(k+1)}\|_1 \|\boldsymbol{U}_3\|_1 \right\}$$

$$\mathcal{G}^{(k+1)} \leftarrow \arg\min \Big\{ L(\mathcal{G}, \boldsymbol{U}_1^{(k+1)}, \boldsymbol{U}_2^{(k+1)}, \boldsymbol{U}_3^{(k+1)}) + \textstyle\sum_{i=1}^3 \varrho_i \|\mathcal{G}_{(i)} - \boldsymbol{D}_i^{(k)} \boldsymbol{V}_i^{(k)\prime} + (\mathcal{C}_i^{(k)})_{(i)}\|_{\mathrm{F}}^2 \Big\}$$

for $i \in \{1, 2, 3\}$ **do**

$$\boldsymbol{D}_i^{(k+1)} \leftarrow \underset{\boldsymbol{D}_i = \mathrm{diag}(\boldsymbol{d}_i)}{\arg\min} \|\mathcal{G}_{(i)}^{(k+1)} - \boldsymbol{D}_i \boldsymbol{V}_i^{(k)\prime} + (\mathcal{C}_i^{(k)})_{(i)}\|_{\mathrm{F}}^2$$

$$\boldsymbol{V}_i^{(k+1)} \leftarrow \underset{\boldsymbol{V}_i'\boldsymbol{V}_i = \boldsymbol{I}_{r_i}}{\arg\min} \|\mathcal{G}_{(i)}^{(k+1)} - \boldsymbol{D}_i^{(k+1)} \boldsymbol{V}_i' + (\mathcal{C}_i^{(k)})_{(i)}\|_{\mathrm{F}}^2$$

$$(\mathcal{C}_i^{(k+1)})_{(i)} \leftarrow (\mathcal{C}_i^{(k)})_{(i)} + \mathcal{G}_{(i)}^{(k+1)} - \boldsymbol{D}_i^{(k+1)} \boldsymbol{V}_i^{(k+1)\prime}$$

end for

$$\mathcal{A}^{(k+1)} \leftarrow \mathcal{G}^{(k+1)} \times_1 \boldsymbol{U}_1^{(k+1)} \times_2 \boldsymbol{U}_2^{(k+1)} \times_3 \boldsymbol{U}_3^{(k+1)}$$

until convergence

Algorithm 3 ADMM subroutine for sparse and orthogonal regression

Initialize: $\boldsymbol{B}^{(0)} = \boldsymbol{W}^{(0)}$, $\boldsymbol{M}^{(0)} = \boldsymbol{0}$

repeat $k = 0, 1, 2, \ldots$

$$\boldsymbol{B}^{(k+1)} \leftarrow \arg\min_{\boldsymbol{B}'\boldsymbol{B} = \boldsymbol{I}} \left\{ n^{-1} \|\boldsymbol{y} - \boldsymbol{X}\mathrm{vec}(\boldsymbol{B})\|_2^2 + \kappa \|\boldsymbol{B} - \boldsymbol{W}^{(k)} + \boldsymbol{M}^{(k)}\|_{\mathrm{F}}^2 \right\}$$

$$\boldsymbol{W}^{(k+1)} \leftarrow \arg\min_{\boldsymbol{W}} \left\{ \kappa \|\boldsymbol{B}^{(k+1)} - \boldsymbol{W} + \boldsymbol{M}^{(k)}\|_{\mathrm{F}}^2 + \lambda \|\boldsymbol{W}\|_1 \right\}$$

$$\boldsymbol{M}^{(k+1)} \leftarrow \boldsymbol{M}^{(k)} + \boldsymbol{B}^{(k+1)} - \boldsymbol{W}^{(k+1)}$$

until convergence

Summary of the paper

Other results obtained by the paper, that we didn't focused on:

- Theoretical results on the MLR and SHORR estimators
- Theoretical results on the Consistency of the rank selection
- Theoretical connection with factor models
- Testing these theoretical results through experiments

Our implementation

Our code only relies on itself and somes basic functions of the main libraries PyTorch, Matplotlib etc. What parts of the paper have been implemented and tested?

- Rank estimation and rank selection consistency
- MLR estimator algorithm and comparison with OLS and RRR estimators
- Testing on the synthetic dataset and the UCI Air Quality Data

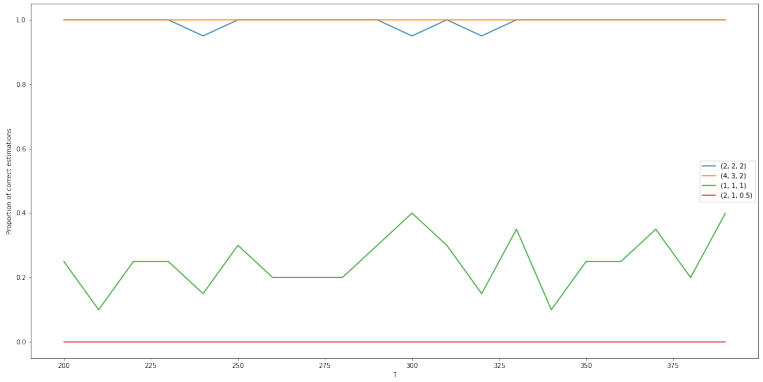
Our implementation

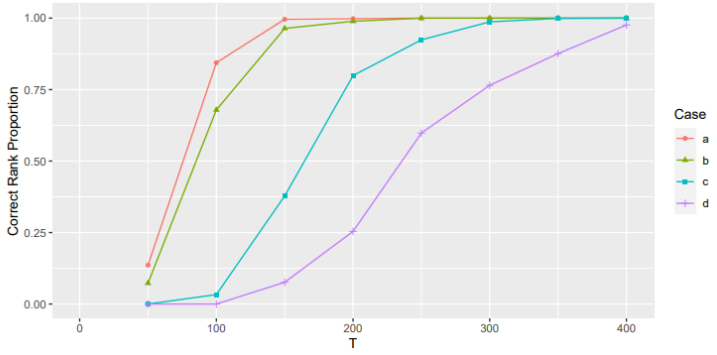
Problematics encountered during the implementation:

- SVD and inverse computations
- How to carry out each update : closed-form or SGD?
- For the second algo : updates require quite deepful nested iterations

Results

- Rank estimation and rank selection consistency





Results

- MLR estimator algorithm and comparison with OLS and RRR estimators

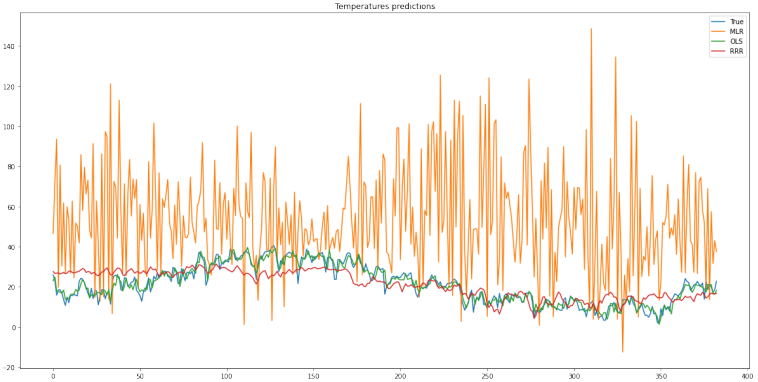


Figure: Temperature

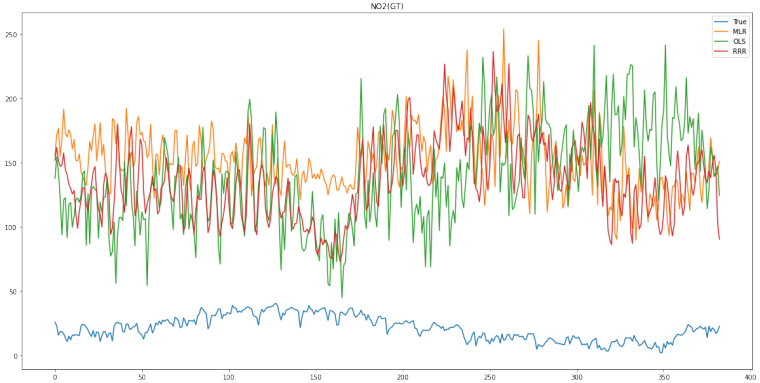


Figure: NO2 concentration