Core Matrix Regression and Prediction with Regularization

Dan Zhou
dz239@njit.edu
New Jersey Institute of technology
USA

Ajim Uddin
au76@njit.edu
New Jersey Institute of technology
USA

Zuofeng Shang
zshang@njit.edu
New Jersey Institute of technology
USA

Cheickna Sylla
cheickna.sylla@njit.edu
New Jersey Institute of technology
USA

Dantong Yu
dtyu@njit.edu
New Jersey Institute of technology
USA

ABSTRACT

Many finance time-series analyses often track a matrix of variables at each time and study their co-evolution over a long time. The matrix time series is overly sparse, involves complex interactions among latent matrix factors, and demands advanced models to extract dynamic temporal patterns from these interactions. This paper proposes a Core Matrix Regression with Regularization algorithm (CMRR) to capture spatiotemporal relations in sparse matrix-variate time series. The model decomposes each matrix into three factor matrices of row entities, column entities, and interactions between row entities and column entities, respectively. Subsequently, it applies recurrent neural networks on interaction matrices to extract temporal patterns. Given the sparse matrix, we design an element-wise orthogonal matrix factorization that leverages the Stochastic Gradient Descent (SGD) in a deep learning platform to overcome the challenge of the sparsity and large volume of complex data. The experiment confirms that combining orthogonal matrix factorization with recurrent neural networks is highly effective and outperforms existing graph neural networks and tensor-based time series prediction methods. We apply CMRR in three real-world financial applications: firm earning forecast, predicting firm fundamentals, and firm characteristics, and demonstrate its consistent performance superiority: reducing error by 23%-53% over other state-of-the-art high-dimensional time series prediction algorithms.

CCS CONCEPTS

• **Computing methodologies** → **Factorization methods**; **Regularization**; *Learning latent representations*; *Neural networks*; • **Information systems** → **Spatial-temporal systems**.

KEYWORDS

Tensor algorithm, matrix-variate time series prediction, matrix factorization, recurrent neural networks

ICAIF'22, November 2–4, 2022, New York, NY

ACM ISBN 978-1-4503-9376-8/22/11…$15.00
https://doi.org/10.1145/3533271.3561709

ACM Reference Format:
Dan Zhou, Ajim Uddin, Zuofeng Shang, Cheickna Sylla, and Dantong Yu. 2022. Core Matrix Regression and Prediction with Regularization. In *proceedings of ACM International Conference on AI in Finance (ICAIF'22).* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3533271.3561709

1 INTRODUCTION

Representing Spatio-temporal data as matrix-variate time series with the cross-sectional observations at each time in a matrix format is a powerful high-dimensional data organization method. It has a wide range of applications, including financial market trading [7, 27], environmental monitoring [3, 23], cross-selling customer product [24, 32], and transportation network [31]. The adoption of matrix-variate time series in such wide spectra of application domains has resulted from its high practicality. For example, firms periodically release their financial statements, including information such as total assets, liabilities, earnings, growth potential, etc. Representing such accounting data in a matrix-variate time series facilitates building sophisticated tensor-based models that can detect latent factors, dependency, causality, and interactions across firms over time. As shown in Figure 1, the cross-sectional observations in each time step form multi-dimensional arrays, i.e., a third-order tensor with dimensions of time × firms ×, and fundamentals. Table 1 shows the snapshot of a matrix (cross-sectional observations) from the third-order tensor at one time slice.

This paper proposes a novel Core Matrix Regression with Regularization (CMRR) to impute and predict matrix-variant time series stored in a 3rd-order tensor. We first slice the tensor into individual matrices along the time dimension, perform orthogonal matrix factorization on each matrix, and use the resulting core low-rank matrix for time-series prediction. The key idea of this design is to reduce the dimensionality of all components by separating the time-variant interactions from time-invariant factors and then design recurrent neural networks for non-linear prediction with time-variant interactions. We attribute the observations' temporal patterns and fluctuation to the joint interactions between column and row entities in the matrix context.

Our approach solves three important challenges in Spatio-temporal analysis using matrix variant time-series. First, previous efforts concerning matrix-valued time series [5, 28, 30, 33] either partition observations along the column or row direction based on the assumption of independent observations along the matrix rows and columns [30, 33], or flatten the matrix data into vectors to avoid the complexity of processing matrix [5, 28]. However, in real-world

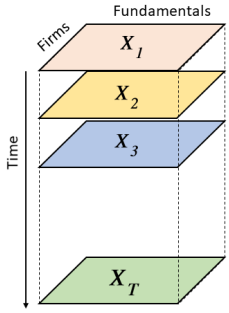


Figure 1: An Illustration of Matrix-variate Time Series

Table 1: Cross-sectional observations at one time step in a matrix-variate time series

Firms	Total Assets	Earning Growth	⋯	Market Price
Firm 1	$x_{1,1}$	$x_{1,2}$	⋯	$x_{1,n}$
Firm 2	$x_{2,1}$	$x_{2,2}$	⋯	$x_{2,n}$
⋮	⋮	⋮		⋮
Firm m	$x_{m,1}$	$x_{m,2}$	⋯	$x_{m,n}$

data, cross-sections of observation often interact and are interdependent. Previous approaches have the drawback of overlooking the interaction among entities and may incur a high bias in prediction. In contrast, our approach performs matrix decomposition on the entire cross-section of variables while fully preserving and utilizing its matrix structure.

Second, the matrix-valued time series model often suffers an overfitting problem in retaining the latent interactions among the columns and rows of a matrix. To solve this challenge, CMRR uses an orthogonal matrix factorization method to decompose the observations in matrix form to the left orthogonal basis (the hidden dimension for row entity), right orthogonal basis (the hidden dimension of column entity), and a core matrix with low rank containing the time-specific interaction weights between the left and right orthogonal basis. Using a core matrix allows us to develop a parsimonious model to capture the complex interactions within a matrix, thereby avoiding overfitting problems and achieving good generalization performance.

Third, the existing algorithms dealing with matrix variant data structures are not designed for handling missing values. Many real-world datasets have missing values due to measurement error, privacy, and random or systematic loss. Our algorithm performs sparse matrix factorization on the cross-sectional matrix at each time point. The low-dimensional embedding bypass the requirement of a complete matrix for future prediction.

We tested our algorithm in Spatio-temporal analysis for three real-world financial datasets. These are firms earning per share (EPS) predictions from financial analyst forecast data, firms' fundamental predictions, and firms' characteristics predictions. These three datasets possess all the challenges mentioned above. For example, our financial analysts' earning forecast data sample has more than 90% missing values. The high missing rate comes from multiple reasons, including analysts intentionally skipping the report for a firm, their personal or professional interests toward firms, or their tendency to follow one firm, stop following, and re-follow a firm [26]. The use of financial data also provides a perfect example of factor interaction. A firm's performance depends on its internal factors such as revenues, assets, and liabilities as well as external macro-economic (market factors) conditions and the performance of related firms. In the cross-section of firms, these factors change over time and affect firms' performance. However, within the cross-section, each firm's inherent structure and embedding remain the same over the years. Therefore, our approach first separates the time-variant and time-invariant components and then applies recurrent neural networks to those highly volatile components while keeping the remaining embedding fixed and stable with regularization.

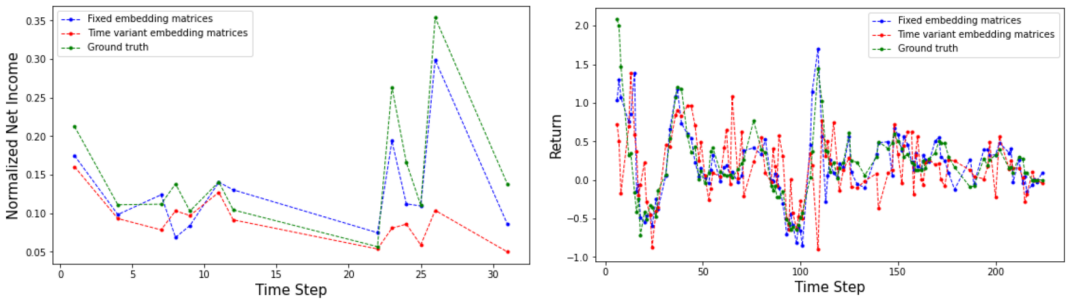
The experimental analysis demonstrates our algorithm's superiority in capturing these time-variant and time-invariant components. As Figure 2 shows, the periodic time series patterns reconstructed by our proposed matrix decomposition with two fixed embedding matrices U and V (blue curve) appear to fit the actual observations (green curve) better than the matrix-wise reconstruction by time-variant embedding matrices U and V (red curve) with different embedding at each time step. Furthermore, in EPS forecast data, CMRR reduces prediction error by 33% over LSTM, 34% over GRU, and 23% over TGCN. In corporate fundamentals prediction, CMMR outperforms GRU by 10%, LSTM by 8%, and DeepGLO by 17%. These performance improvements are consistent in terms of different performance metrics. These findings confirm our intuition that using the decomposition method to separate data signals into the time-variant/-invariant components indeed improves data reconstruction and prediction.

The main contributions of our paper are summarized as follows:

- We develop a novel matrix-variate time series prediction algorithm based on orthogonal matrix factorization and recurrent neural networks to efficiently learn intrinsic dependency and temporal dynamics among the high-dimensional matrix-valued time series.
- The adopted SGD performs an efficient element-wise matrix decomposition method and the associated back-propagations only on the observed data elements in each mini-batch while ignoring the remaining ones. As a result, our algorithm works exceptionally well even with a highly sparse dataset.
- We design two types of regularization, i.e., orthogonal and diagonal regularization to reduce the redundancy and colinearity of learned embedding and minimize the variance of prediction errors.
- We apply our proposed algorithm to solve real-world problems in finance and demonstrate our model improves the prediction accuracy and speeds up inference over current state-of-the-art techniques.

2 RELATED WORK

Decomposing a matrix (tensor) into a core matrix (tensor) to reduce the number of model parameters started to appear in recent literature [1, 7, 15, 27]. Previous works use vector and matrix-based methods to mitigate the complexity of the high-dimensional matrix-variate time series prediction tasks. The vector-based algorithms typically build models on the row and column vectors separately [5, 6, 18, 19, 28, 30, 33]. However, vectorizing the cross-sectional



(a) Net Income of Firms' Fundamentals

(b) Return of Firms' Characteristics

Figure 2: These figures demonstrate the advantage of the proposed CMRR. It presents the reconstructed time series of test data for two variables (a) net income from firms' fundamentals and (b) return from firms' characteristics. The blue line is the pattern reconstructed by our approach, the red line is the pattern reconstructed using simple matrix decomposition, and the green line is the ground truth.

observations in matrix format ignores the essential interactions within the different modes. In [1], the authors first use CP decomposition to factorize the input time series, design the time series forecasting algorithms only on the temporal embedding, and finally reconstruct the tensor to generate predictions. In [15], the authors reduce the size of input data by Tucker decomposition and apply RNNs to capture the temporal dynamics on the Tucker core. In [7, 27], authors build a matrix factorization model to capture the dynamics and co-movement of the matrix-valued time series with a low-dimensional latent factor matrix. In [29], authors propose TMRF, which views high-dimensional multivariate time series as a matrix with assumptions of linear temporal dependency among time points. Finally, DeepGLO proposed in [22] resolves the over-simplified temporal model and uses Temporal Convolution Networks to regularize and predict the basis of time series. DeepGLO gains significant improvement over TMRF for dense data. However, it loses the capability to handle sparsity as learning local properties demands complete observations.

Compared to the vector-based approach, our model attributes the temporal patterns and the fluctuation of the observations to the interactions between the cross-entities and cross-features in the matrix environment. In addition, our algorithm separates the time-variant interactions from time-invariant factors and focuses on predicting time-variant interactions with recurrent neural networks. As a result, it offers superiority over other matrix and tensor-based prediction models. Finally, compared to [22, 29] our model can handle spares matrix/tensor and successfully applied to Spatio-temporal settings with missing observations.

3 METHODOLOGY

3.1 Problem Definition

Given the Spatio-temporal data in the form of a third-order tensor $\mathcal{X} = \{X_1, X_2, \cdots, X_T\} \in \mathbb{R}^{m \times n \times T}$ that consists of T cross-sectional observation matrices at each time step t: $X_t \in \mathbb{R}^{m \times n}$. At each time step t, the prediction task takes lag-p slices of historical observations, $M_t := \{X_{t-p+1}, X_{t-p+2}, \ldots, X_t\} \in \mathbb{R}^{m \times n \times p}$, as input and predicts $X_{t+i}, 1 \leq i \leq I$ for up to I time steps in the future. The objective is to minimize the mean squared error (MSE) in prediction:

$$\mathcal{L} = \sum_{i=1}^{I} \frac{1}{\sum_{j,k}(W_{t+i})_{jk}} \left\| W_{t+i} \odot (\hat{X}_{t+i} - X_{t+i}) \right\|_F^2 \tag{1}$$

$W_{t+i} \in \{0,1\}^{m \times n}$ is the index matrix of the available real observations at time $t+i$. $\hat{X}_{t+i}$ contains the predicted values that are derived from the RNNs outputs.

3.2 Architecture

An overview of the proposed CMRR with its three modules is presented in Figure 3. First, we view the spatio-temporal observations as a third-order tensor and attempt to extract the common basis of column space and row space of the lag-p matrix $X_{t'} \in M_t$ by applying orthogonal decomposition module **(Section 3.3)** on each of p matrices and factorizing it into three matrices $U \in \mathbb{R}^{m \times r}$, $V \in \mathbb{R}^{n \times r}$, $S_{t'} \in \mathbb{R}^{r \times r}$, where U and V are the same across all lag-p matrices of M_t. The key idea of this design is to apply matrix decomposition to first separate the time-variant interactions from time-invariant factors, reduce the dimensionality (complexity) of all components, and focus on predicting time-variant interactions with non-linear recurrent neural networks. Next, CMRR architecture employs a Gated Recurrent Unit (GRU) module **(Section 3.4)** to scan through p input matrices and extract temporal dynamics and predict the interactions in the next time step $\hat{S}_{t+1}$. Finally, the output module **(Section 3.5)** takes $\hat{S}_{t+1}$, U and V to reconstruct the future cross-section $\hat{X}_{t+1}$. To ensure the orthogonal matrix factorization works properly, we design a regularized reconstruction loss that consists of a reconstruction term, an orthogonal regularization term, and a diagonal regularization term.

3.3 Orthogonal Factorization on A Sequence of Matrices

Singular Value Decomposition (SVD) is a commonly used approach for principal component analysis, latent features learning [16], low-rank matrix approximate, and data imputation [4]. However, as the singular values in SVD only represent the variance of principal components in SVD, it requires expanding the capacity of the middle

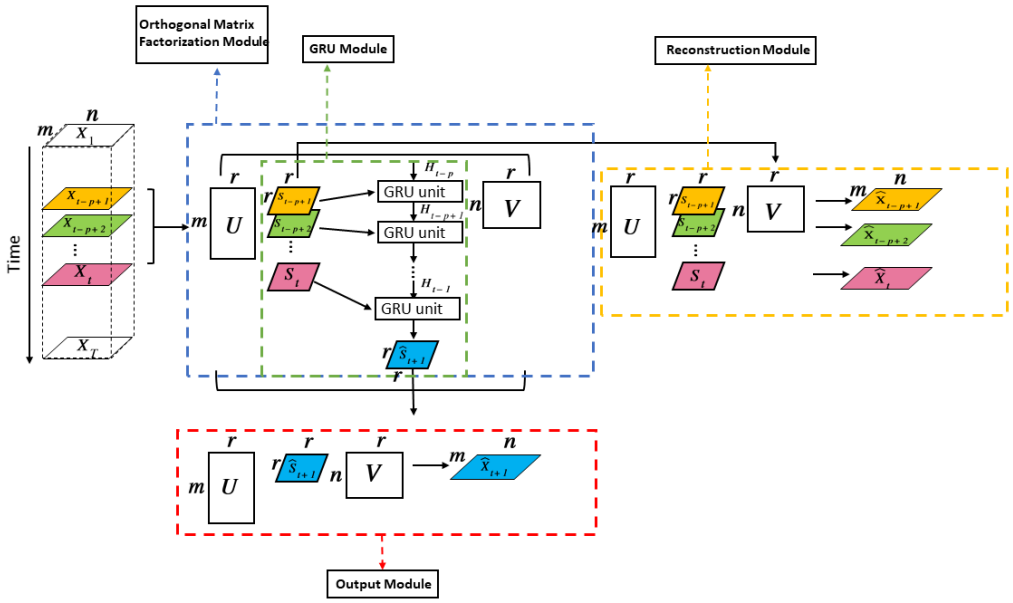


Figure 3: Core Matrix Regression with Regularization (CMRR) Model Architecture

matrix S_t to model interactions between the columns of U and V along the time horizon. Another challenge of using the SVD-based algorithm is its high computational complexity. To address this problem, we propose an orthogonal matrix factorization method that approximates the estimation of SVD and employs Stochastic Gradient Descent (SGD) based optimization for fast convergence in training.

For each dense matrix X_t and its lag-p M_t, we define the following orthogonal matrix co-factorization among lag-p matrices:

$$\begin{aligned} \min_{U,V,S_{t'}} & \sum_{t'=t-p+1}^{t} \|X_{t'} - US_{t'}V^\top\|_F \\ s.t.\ & U^\top U \text{ is a diagonal matrix} \\ & V^\top V \text{ is a diagonal matrix} \end{aligned} \tag{2}$$

Here, $U \in \mathbb{R}^{m\times r}$ contains the left orthogonal basis (the hidden dimension of row entity), $V \in \mathbb{R}^{n\times r}$ contains the right orthogonal basis (the hidden dimension of column entity), and $S_{t'} \in \mathbb{R}^{r\times r}$ encodes interactions. Different from SVD, $S_{t'}$ is not necessarily a diagonal matrix in our core matrix co-factorization because we need all elements to encode the time-variant components of all lag-p matrices. All matrices have a rank of r.[1]

Similar to the assumption in [27], we assume that matrix-valued time series has time-variant and time-invariant factors, and therefore, we fix the left orthogonal matrix U where each row of dimension r represents a row entity (for example, a firm in the EPS dataset) and the right orthogonal matrix V where each row of dimension r represent a column feature (for example, a financial indicator in the EPS dataset). When we reconstruct the lag-p matrices for each time $t : 1 \le t \le T$, the overlapping of the lag-p windows ensures that all U and V are identical across time. The implementation is simple: the T iterations of matrix reconstructions share the same set of trainable parameters in factor matrices U and V. The time-variant information falls in the core matrix $S_{t'}$ that encodes the interactions between the column and row latent factors and drives the structural changes in the observation matrix. In our model, we attribute the temporal patterns and dynamics to the joint interactions between the column and row entities in the networked environment, while the latent attribute of column and row entities will not have prominent alternations over time. In addition, we use RNN to model non-linear temporal changes in the central interaction matrices.

To eliminate co-linearity in the column vectors of U and V, we enforce orthogonality and normalization of the column vectors in matrix U and V and minimize the dimensionality of the left and right factor matrices. The objective function of orthogonal regularization is defined as:

$$\mathcal{L}_{ortho} = \|U^\top U \odot (\mathbf{1}\mathbf{1}^\top - I)\|_{2,2} + \|V^\top V \odot (\mathbf{1}\mathbf{1}^\top - I)\|_{2,2} \tag{3}$$

where $\mathbf{1}\mathbf{1}^\top$ is the matrix of all ones. The diagonal regularization term in Eqn (3) is the $\ell_{p,q}$ matrix norm. The $\ell_{2,2}$ norm enforces the element-wise convergence to zero in a matrix. The regularization term penalizes the inter-dependency (correlation or collinearity) among the latent features, i.e., the column vectors $U_{:,i}$ and $U_{:,j}$ in the left orthogonal matrix U. Given U and V, we can calculate the interaction matrix at each time t' as follows:

$$S_{t'} = U^\top X_{t'} V \tag{4}$$

We introduce another diagonal regularization on matrix $S_{t'}$ and make the decomposition approximate SVD and encourage interactions to be only active in the diagonal locations, i.e., $U_{i,i}$ and $V_{j,j}$. The diagonal regularization is defined as follows:

$$\mathcal{L}_{diagnal} = \left\|S_{t'} - diag(S_{t'})\right\|_F^2 \tag{5}$$

[1]We can easily extend the definition to the co-factorization of sparse matrices by only considering the mean square errors of observed entries while ignoring all missing ones (Eqns. 1 and 10).

where $S_{t'} = U^\top X_{t'} V, t-p+1 \le t' \le t$ and $diag(S_{t'})$ is the diagonal matrix of $S_{t'}$.

One constraint with the standard SVD is that it requires a complete matrix for factorization and suffers the slow convergence and long computation time for a large matrix. This paper adopts an element-wise orthogonal matrix factorization approach to reconstruct matrices from the partial observation set Ω while ignoring the remaining missing entries. To calculate matrix $S_{t'}$ from a sparse matrix, we first decompose the matrix $X_{t'}$ to the sum of its observed elements. For each matrix $X_{t'}$ in history sequence $M_t = \{X_{t-p+1}, X_{t-p+2}, \cdots, X_t\}$, let $\Omega_{X_{t'}}$ be the observation set of matrix $X_{t'}$, the sparse matrix $X_{t'}$ can be decomposed to the sum of matrices, each of which contains one observed entry: $X_{t'} = \sum_{i\in\Omega_{X_{t'}}} \sum_{j\in\Omega_{X_{t'}}} x_{ij} E_{ij}$, where x_{ij} is the i, j-th element of $X_{t'}$, and E_{ij} is an $m \times n$ matrix whose i, j-th entry is one and the remaining ones are zero. The element-wise expression of $S_{t'}$ is: where $U_{i,:} \in \mathbb{R}^{1\times r}$ is the i-th row of matrix U, $V_{j,:} \in \mathbb{R}^{1\times r}$ is the j-th row of matrix V and $\otimes$ represents the outer product. $W_{t'} \in \{0,1\}^{m\times n}$ is the index matrix of the cross sectional observations $X_{t'}$ and $\{W_{t'}\}_{i,j} = 1$ if $\{X_{t'}\}_{i,j}$ is observed, otherwise, $\{W_{t'}\}_{i,j} = 0$. The Hadamard product $\odot$ selects the observed entries.

Finally, given the lag-p sequence M_t at each time step t, the orthogonal matrix factorization module (OMF) generates the outputs:

$$OMF(M_t) = U, V, S_{t-p+1}, S_{t-p+2}, \cdots, S_t \tag{6}$$

Noted that CMRR uses the same factor matrices U and V for all training matrices between time $1 \le t \le T$.

3.4 Gated Recurrent Unit (GRU) Module

Given the output from orthogonal matrix factorization (6), in the next step we apply recurrent neural networks to estimate the temporal dynamics from $S_{t-p+1}, S_{t-p+2}, \cdots, S_t$. The standard recurrent neural networks work well with the inputs and hidden states in the vector form. It requires vectorizing the matrix first, and as a result, it can not retain the matrix structure. We adopt the method proposed in [20] to extend the Gated Recurrent Unit (GRU) to retain the matrix structure and generate matrix output. For each core matrix $S_{t'}$ in $\{S_{t-p+1}, S_{t-p+2}, \cdots, S_t\}$, we use the same GRU to process each column of core matrix $S_{t'}$. The matricized GRU is presented in the following:

$$\begin{aligned}
Z_{t'} &= sigmoid(W_Z S_{t'} + Y_Z H_{t'-1} + B_Z) \\
R_{t'} &= sigmoid(W_R S_{t'} + Y_R H_{t'-1} + B_R) \\
\widetilde{H}_{t'} &= tanh(W_H S_{t'} + Y_H (R_{t'} \odot H_{t'-1}) + B_H) \\
H_{t'} &= (1 - Z_{t'}) \odot H_{t'-1} + Z_{t'} \odot \widetilde{H}_{t'}
\end{aligned} \tag{7}$$

where $S_{t'} \in \mathbb{R}^{r\times r}$ and $H_{t'} \in \mathbb{R}^{r\times r}$ denote the input core matrix and the hidden state matrix at the time step t'; $Z_{t'}, R_{t'} \in \mathbb{R}^{r\times r}$ represent the update gate and reset gate, respectively; $\widetilde{H}_{t'} \in \mathbb{R}^{r\times r}$ is the memory content for updating the hidden state. And for $\alpha = \{Z, R, H\}$, $W_\alpha \in \mathbb{R}^{r\times r}$ represents the weight matrices corresponding to input matrix $S_{t'}$ within different gates; $Y_\alpha \in \mathbb{R}^{r\times r}$ represents the weight matrices corresponding to latent hidden states; B represents the corresponding bias vectors. Moreover, sigmoid$(\cdot)$ and tanh$(\cdot)$ denote the logistic sigmoid activation function and tangent activation function, respectively; $\odot$ represents the Hadamard product.

Given the core matrices $\{S_{t-p+1}, S_{t-p+2}, \cdots, S_t\}$ of lag-p time steps, the output of GRU module is $H_t \in \mathbb{R}^{r\times r}$. Where, H_t is the hidden state matrix in time step t and the predicted core matrix of the $t+1$-th time step and denoted by $\hat{S}_{t+1}$.

$$H_t = GRU(\{S_{t-p+1}, S_{t-p+2}, \cdots, S_t\}) \tag{8}$$

3.5 Output Module

After getting the predicted core matrix $\hat{S}_{t+1} = H_t \in \mathbb{R}^{r\times r}$, we predict the snapshot $\hat{X}_{t+1}$ in the future time step by multiplying H_t with the left orthogonal matrix $U \in \mathbb{R}^{m\times r}$ and right orthogonal matrix $V \in \mathbb{R}^{n\times r}$ as follow:

$$\hat{X}_{t+1} = U H_t V^\top \tag{9}$$

3.6 Training

Given a time series of cross-sectional observations denoted as a third-order tensor $\mathcal{X} = \{X_1, X_2, \cdots, X_T\} \in \mathbb{R}^{m\times n\times T}$ with the cross-section observation at each time step t is a matrix $X_t \in \mathbb{R}^{m\times n}$ for $t = 1, \cdots, T$. For each time step t, we stack the snapshots of p prior time steps, i.e., the lag of p slices M_t to predict the snapshot X_{t+1} at the next time step. For each snapshot $X_{t'}$ in M_t, the objective function is defined as:

$$\begin{aligned}
\mathcal{L} = & \frac{1}{\sum_{j,k}(W_{t+i})_{jk}} \left\| W_{t+1} \odot (\hat{X}_{t+1} - X_{t+1}) \right\|_F^2 \\
& + \lambda_r \sum_{t'=t-p+1}^{t} \frac{1}{\sum_{j,k}(W_{t'})_{jk}} \left\| W_{t'} \odot (X_{t'} - U S_{t'} V^\top) \right\|_F^2 \\
& + \lambda_o (\|U^\top U \odot (\mathbf{1}\mathbf{1}^\top - I)\|_{2,2} + \|V^\top V \odot (\mathbf{1}\mathbf{1}^\top - I)\|_{2,2}) \\
& + \lambda_d \sum_{t'=t-p+1}^{t} \left\| S_{t'} - diag(S_{t'}) \right\|_F^2
\end{aligned} \tag{10}$$

where $W_t \in \{0,1\}^{m\times n}$ is the index matrix. The Hadamard product $\odot$ indicates that we only consider the reconstruction and prediction errors originating from the observed values in X_t. The second term is the reconstruction error of the lag-p slices, the third term is the orthogonal regularization term enforcing orthogonality, the fourth term denotes diagonal regularization that encourages $S_{t'}$ to be a diagonal matrix. Equation (10) contains three regularization coefficients, μ_r, μ_o, and μ_d that are determined by grid-search and cross-validation.

4 EXPERIMENT

We conducted experiments in three Real-world datasets to evaluate our algorithm based on three aspects: 1) efficiency in matrix time series prediction in both time and accuracy compared to other state-of-the-art time series techniques; 2) the ability to capture the complex interactions among different dimensions; 3) the efficiency and scalability in the sparse and dense dataset.

4.1 Experimental Setup

4.1.1 Data. To show the applicability and generalization capability of CMRR, we perform our experiments on three public datasets with different levels of sparsity. The data shape and number of observed entries for each of the datasets are reported in Table 2. First, the financial analyst **EPS** forecast. The quarterly financial

Table 2: Statistics of Data

Datasets	Shape	#Obs. entries	Obs. ratio	train-val-test
EPS	(180,173,32)	42,867	4.30%	50%-25%-25%
Fundamentals	(180,19,32)	10,8486	99.13%	50%-25%-25%
Characteristics	(964,93,227)	20,351,004	100.00%	58%-10%-32%

analysts' EPS forecast data are collected from Thomson Reuters I/B/E/S and are available at https://wrds-www.wharton.upenn.edu/. This data is highly sparse, with almost 96% missing values. To make our analysis more consistent and meaningful, we remove analysts who have been in business for less than four years or made less than 200 predictions in their entire careers. Second, firms accounting **Fundamentals**. We select 19 accounting variables following established literature [2, 12, 14, 25]. Quarterly accounting fundamental variables are collected from COMPUSTAT and CRSP. For analysts' EPS forecast and firm fundamental datasets, we consider 180 firms for the period from Q1-2010 to Q4-2017. Third, firm **Characteristics** data. This data comprises 93 firms' characteristics for 964 large firms listed in North America. The data is the same as that one used in [9, 10]. These variables are the documented drivers (factors) of the stock returns, including accounting figures and ratios, risk and liquidity measures, and momentum factors. The monthly data for accounting characteristics are collected for the period from 2000 to 2019. For the experiments, we split all data into three folds: training/validation/test splits along the temporal dimension.

4.1.2 Baseline methods. In the experiment, we evaluate the performance of CMRR with other state-of-the-art models designed for high-dimensional time series prediction. These includes: **Tensor Graph Convolutional Network (TGCN)** [15] is a hybrid method of tensor graph convolutional networks and recurrent networks. **CP Forecast** [11] is a method for temporal forecasting based on CP decomposition. **DeepGLO** [22] is a multivariate time series prediction algorithm based on matrix factorization and temporal convolution networks. **TRMF** [29] is a multivariate time series prediction algorithm based on matrix factorization with a temporal regularization in an autoregressive framework. **Long Short-term memory (LSTM)** [13] and **Gated Recurrent Unit (GRU)** [8] are commonly used gated recurrent neural network models. For the four multivariate time series prediction algorithms DeepGLO, TRMF, LSTM, and GRU, we apply separate models for each firm in the EPS, firm's fundamentals, and characteristics datasets.

4.1.3 Hyper-parameter settings. We implement CMRR in PyTorch [21]. For all experiments, we use the Adam optimizer [17] with the default configuration. We use MSE as the loss function and set early stopping on the validation loss with $patience = 10$. We tune the hyper-parameters for CMRR and select the proper matrix rank of the orthogonal matrix factorization. We use the grid search to choose the coefficients λ_r, λ_o, λ_d of the reconstruction regularization term, orthogonal regularization term and diagonal regularization term within [$1e^{-5}$, $1e^{-4}$, $1e^{-3}$, 0.01, 0.1, 0.5, 1, 10, 100]. The python implementation of our algorithm and experiments are available at https://anonymous.4open.science/r/CMRR-BCCE/.

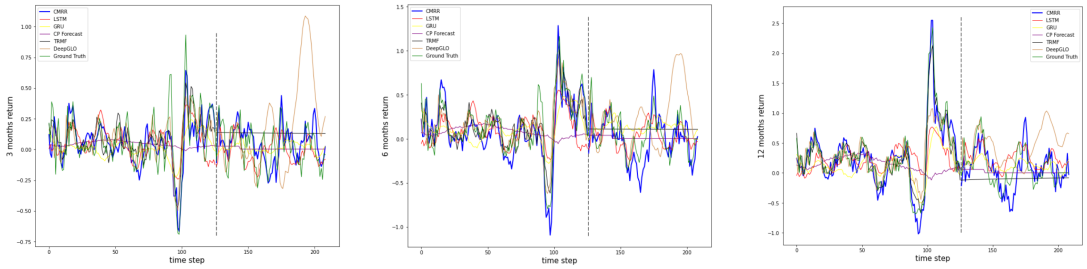
4.2 Time Series Prediction Evaluation

The prediction result is reported in Table 3. The EPS data is highly sparse, with only a 4.30% observation ratio. We Apply the sparse version of CMRR. The result shows that CMRR exceeds all baseline methods. Specifically, compared with GRU, LSTM, CP Forecast, DeepGLO, TRMF, and TGCN, CMRR lowers the RMSE by 34%, 33%, 53%, 48%, 35%, 23%, respectively. Among all methods, CMRR with core matrix factorization and TGCN enhanced by graph convolutional networks perform better than the four multivariate models: LSTM, GRU, DeepGLO, and TRMF. This observation is because CMRR and TGCN have a clear advantage of capturing the structure in the dataset and various interactions among different modes in a matrix or a tensor. Despite capturing the interactions among different modes, CP forecast shows inferior prediction performance than all other methods. The EPS data is sparse and noisy, and the linear fit of the CP factorization is not perfect; as a result, it incurs bias. TGCN performs graph convolutions on the affinity matrices along with each tensor mode. When datasets are noisy and do not have any auxiliary networks, TGCN uses Pearson correlation to build networks among each tensor mode. Such a procedure can introduce information redundancy and leads to significant variances in the generalization of the downstream prediction tasks. As a result, in all three performance metrics, CMRR outperforms TGCN.

The firms' Fundamental dataset has significant variances in the variables and measurement distributions. We first apply the min-max normalization to normalize the data per variable. As reported, the proposed CMRR is superior to the baselines in all three performance metrics with 2-36%, 1-64%, and 1-70% improvement in terms of RMSE, MAE, and R^2, respectively. Finally, the Characteristics dataset is complete without any missing values, and we directly apply dense matrix factorization in CMRR. Similar to the other two datasets, the experiment suggests that the CMRR is superior to all the baselines in all three evaluation metrics, particularly with 2-68% improvement in RMSE, 7-74% improvement in MAE, and 5-81% improvement in R2. The result confirms our model's effectiveness in capturing the implicit relations within different dimensions and learning the temporal patterns embedded in data.

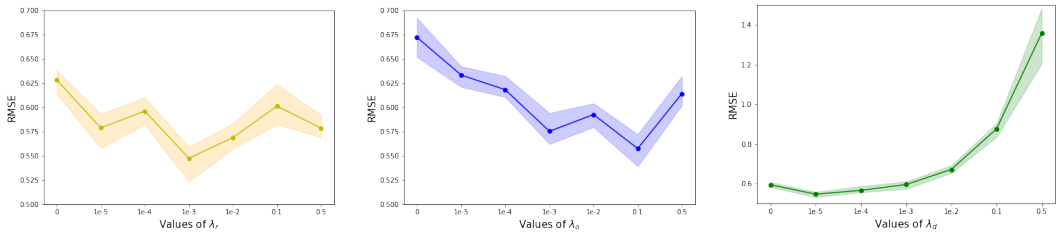
4.3 Visualization of Reconstruction Accuracy

We plot the future value prediction results versus ground truth on the Firms' Characteristics in Figure 4. The result shows the temporal patterns of the three features in Firms' Characteristics data: three-month, six-month, and twelve-month returns. The green curve is the original ground truth, and the blue curve is the time trend extracted via the CMRR prediction result. The vertical dashed line indicates in-sample (left of the line) tests and out-of-sample tests (right of the line). Compared with all baselines, our model fits better to the ground truth. The excellent fit in the in-sample training data demonstrates the orthogonal matrix factorization in our model is efficient in data imputation and reconstruction. Despite slightly deviating from the actual data in the test dataset, our model extracts the time patterns and captures the general trends and the prominent peaks and valleys in the ground-truth temporal patterns. It outperforms the standard RNN models and the multivariate time series prediction models that largely ignore the joint interactions among different firms.



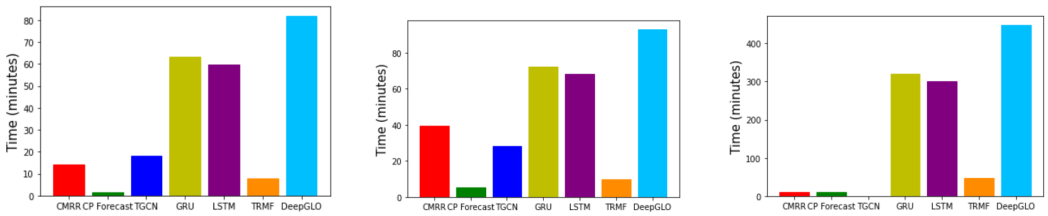
(a) Three Month Return (b) Six Month Return (c) Twelve Month Return

Figure 4: Visualization of future value prediction on Firms' Characteristics



(a) Coefficient of Reconstruction Error Term **(b) Coefficient of Orthonormality Regularization Term** **(c) Coefficient of Diagonal Regularization Term**

Figure 5: RMSE performance on the EPS data for varying the value of (a) coefficient of reconstruction error term λ_r, (b) coefficient of orthonormality regularization λ_o, and (c) coefficient of diagonal regularization λ_d (all from 1e-5 to 0.5).



(a) dataset=EPS **(b) dataset=Fundamentals** **(c) dataset=Characters**

Figure 6: Running time of different time series prediction algorithms

4.4 Experiment on Regularization Term

We also analyze the impact of the three regularization terms on algorithm performance and present the result in Figure 5. λ_r controls the penalty on reconstruction error of the orthogonal matrix factorization. λ_o controls the penalty on orthogonal regularization for left and right orthogonal matrices, and λ_d controls the penalty on diagonal regularization of the interaction matrix. We repeat the experiment ten times and collect the best, average and the worst results and estimate the error distribution. Figure 5(a) depicts the RMSE values for varying λ_r when fixing λ_o=0.1, λ_d=1e-5. Figure 5(b) displays the RMSE values for varying λ_o when fixing λ_r=0.01, λ_d=1e-5. Figure 5(c) presents the RMSE values for varying λ_d when fixing λ_r=0.01, λ_o=0.1.

All three curves demonstrate that the model performance has three regions (overfitting, good-fitting, and underfitting). All models first improve when we increase the regularization strength, indicating models are overfitting the observations. Once all models reach the elbow point, they deteriorate as excessive regularization causes the underfitting problem. The orthogonal regularization improves the model significantly and indicates the importance of eliminating co-linearity and covariance on the embedding dimensions. On the other hand, the reconstruction coefficient ensures that the learned embedding must retain the original data information while predicting the future value. The diagonal regularization indeed improves the model performance. But the improvement is relatively moderate. When we increase λ_d to a large value, the core matrix decomposition generates a diagonal matrix for modeling interactions. The CMRR decomposition deteriorates into an SVD that incurs a high bias in the matrix co-factorization along the time dimension.

Table 3: Tensor Completion Results of SafeGraph

Data	Model	RMSE	MAE	R^2
EPS	GRU	0.8254	0.4685	0.4621
	LSTM	0.8132	0.4724	0.4778
	CP Forecast	1.1746	0.7635	0.1389
	DeepGLO	1.0477	0.6811	0.5487
	TRMF	0.8503	0.5447	0.4302
	TGCN	0.7123	0.4523	0.5283
	CMRR	**0.5476**	**0.3403**	**0.7637**
Fundamentals	GRU	0.1506	0.0864	0.7260
	LSTM	0.1479	0.0793	0.7357
	CP Forecast	0.2119	0.1569	0.4573
	DeepGLO	0.1636	0.1031	0.6846
	TRMF	0.1425	0.0778	0.7554
	TGCN	0.1386	0.0567	0.7681
	CMRR	**0.1358**	**0.0561**	**0.7781**
Characteristics	GRU	0.1945	0.1339	0.5793
	LSTM	0.1892	0.1297	0.6017
	CP Forecast	0.5361	0.4592	0.2207
	DeepGLO	0.5807	0.4205	0.1204
	TRMF	0.2421	0.1684	0.3480
	TGCN*	-	-	-
	CMRR	**0.1847**	**0.1207**	**0.6352**

Note: *The characteristics data is too big and TGCN shows out-of-memory error when applying Graph Convolution. Therefore, we did not report the TGCN result.

4.5 Running Time

Figure 6 shows the running time of all five algorithms on three datasets. The result shows that the CMRRs of both sparse and dense versions are much faster than GRU, LSTM, and DeepGLO and comparable to CP, TGCN, and TRMF. We fix the learning rate, lr=1e-4 for all models. The reported time elapsed for all algorithms already incorporates the early stopping criteria with the fixed patience=10. We run all experiments on a single machine with 20 cores/2 threads, equipped with an Intel(R) Xeon(R) Gold 6138 2.00GHz CPU and 93GB RAM.

5 CONCLUSION

This paper designed an innovative approach to effectively model the intrinsic dependence of the cross-sectional observations and confirm its central role in future time prediction algorithms. The proposed CMRR algorithm benefits from several regularization methods in reducing data complexity and separating the data signal into time-variant and time-invariant components and consequently improves the efficacy and effectiveness of the recurrent network module to extract meaningful temporal patterns. The algorithm is highly efficient because of the adoption of SGD to handle the problem of sparse data. This work also sheds light on future studies involving sparse matrix-variate financial data, including bond yield and price prediction or predicting the probability of mergers and acquisitions.

REFERENCES

[1] Miguel Araújo, Pedro Ribeiro, and Christos Faloutsos. 2018. TensorCast: Forecasting Time-Evolving Networks with Contextual Information. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*. International Joint Conferences on Artificial Intelligence Organization, 5199–5203.

[2] Ryan T Ball and Eric Ghysels. 2018. Automated Earnings Forecasts: Beat Analysts or Combine and Conquer? *Management Science* 64, 10 (2018), 4936–4952.

[3] V. Banzon, T. M. Smith, T. M. Chin, C. Liu, and W. Hankins. 2016. A long-term record of blended satellite and in situ sea-surface temperature for climate monitoring, modeling and environmental studies. *Earth System Science Data* 8, 1 (2016), 165–176.

[4] Robert Bell, Yehuda Koren, and Chris Volinsky. 2007. Modeling Relationships at Multiple Scales to Improve Accuracy of Large Recommender Systems. In *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. New York, NY, USA, 95–104.

[5] Dimitris Bertsimas and Martin S Copenhaver. 2018. Characterization of the equivalence of robustification and regularization in linear and matrix regression. *European Journal of Operational Research* 270, 3 (2018), 931–942.

[6] Gary Chamberlain and Michael Rothschild. 1982. *Arbitrage, Factor Structure, and Mean-Variance Analysis on Large Asset Markets.* Working Paper 996. National Bureau of Economic Research.

[7] Elynn Y. Chen, Ruey S. Tsay, and Rong Chen. 2020. Constrained Factor Models for High-Dimensional Matrix-Variate Time Series. *J. Amer. Statist. Assoc.* 115, 530 (2020), 775–793.

[8] Junyoung Chung, Caglar Gulcehre, Kyunghyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. In *NIPS 2014 Workshop on Deep Learning, December 2014.*

[9] Guillaume Coqueret. 2021. Persistence in factor-based supervised learning models. *The Journal of Finance and Data Science* (2021).

[10] Guillaume Coqueret and Tony Guida. 2020. *Machine Learning for Factor Investing: R Version.* CRC Press.

[11] Daniel M. Dunlavy, Tamara G. Kolda, and Evrim Acar. 2011. Temporal Link Prediction Using Matrix and Tensor Factorizations. *ACM Trans. Knowl. Discov. Data* 5, 2, Article 10 (feb 2011), 27 pages.

[12] Eugene F Fama and Kenneth R French. 2006. Profitability, investment and average returns. *Journal of financial economics* 82, 3 (2006), 491–518.

[13] Felix A. Gers, Nicol N. Schraudolph, and Jürgen Schmidhuber. 2003. Learning Precise Timing with Lstm Recurrent Networks. *J. Mach. Learn. Res.* 3, null (mar 2003), 115–143.

[14] Kewei Hou, Mathijs A Van Dijk, and Yinglei Zhang. 2012. The implied cost of capital: A new approach. *Journal of Accounting and Economics* 53, 3 (2012), 504–526.

[15] Baoyu Jing, Hanghang Tong, and Yada Zhu. 2021. Network of Tensor Time Series. In *Proceedings of the Web Conference 2021* (Ljubljana, Slovenia). 2425–2437.

[16] Dohyun Kim and Bong-Jin Yum. 2005. Collaborative filtering based on iterative principal component analysis. *Expert Systems with Applications* 28, 4 (2005), 823–830.

[17] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9.*

[18] Clifford Lam and Qiwei Yao. 2012. Factor modeling for high-dimensional time series: Inference for the number of factors. *The Annals of Statistics* 40, 2 (2012), 694 – 726.

[19] Clifford Lam, Qiwei Yao, and Neil Bathia. 2011. Estimation of latent factors for high-dimensional time series. *Biometrika* 98, 4 (10 2011), 901–918.

[20] Aldo Pareja, Giacomo Domeniconi, Jie Chen, Tengfei Ma, Toyotaro Suzumura, Hiroki Kanezashi, Tim Kaler, and Charles E Leisersen. 2020. Evolvegcn: Evolving graph convolutional networks for dynamic graphs. *AAAI* (2020).

[21] Adam Paszke and et al. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (Eds.). Curran Associates, Inc., 8024–8035.

[22] Rajat Sen, Hsiang-Fu Yu, and Inderjit S. Dhillon. 2019. Think Globally, Act Locally: A Deep Neural Network Approach to High-Dimensional Time Series Forecasting. In *Neural Information Processing Systems (NIPS).*

[23] Shikhar Srivastava and Stefan Lessmann. 2018. A comparative study of LSTM neural networks in forecasting day-ahead global horizontal irradiance with satellite data. *Solar Energy* 162 (2018), 232–247. https://doi.org/10.1016/j.solener.2018.01.005

[24] Chenhao Su, Sheng Gao, and Si Li. 2020. GATE: Graph-Attention Augmented Temporal Neural Network for Medication Recommendation. *IEEE Access* 8 (2020), 125447–125458.

[25] Ajim Uddin, Xinyuan Tao, Chia-Ching Chou, and Dantong Yu. 2020. Nonlinear Tensor Completion Using Domain Knowledge: An Application in Analysts' Earnings Forecast. In *2020 International Conference on Data Mining Workshops (ICDMW).* IEEE.

[26] Ajim Uddin, Xinyuan Tao, Chia-Ching Chou, and Dantong Yu. 2021. Are missing values important for earnings forecasts? A machine learning perspective. *Quantitative Finance* (2021), 1–20.

[27] Dong Wang, Xialu Liu, and Rong Chen. 2019. Factor models for matrix-valued high-dimensional time series. *Journal of Econometrics* 208, 1 (2019), 231–248. Special Issue on Financial Engineering and Risk Management.

[28] Jianxin Yin and Hongzhe Li. 2012. Model selection and estimation in the matrix normal graphical model. *Journal of Multivariate Analysis* 107 (2012), 119–140. https://doi.org/10.1016/j.jmva.2012.01.005

[29] Hsiang-Fu Yu, Nikhil Rao, and Inderjit S Dhillon. 2016. Temporal Regularized Matrix Factorization for High-dimensional Time Series Prediction. In *Advances in Neural Information Processing Systems*.

[30] Junlong Zhao and Chenlei Leng. 2014. STRUCTURED LASSO FOR REGRESSION WITH MATRIX COVARIATES. *Statistica Sinica* (2014).

[31] Ling Zhao, Yujiao Song, Chao Zhang, Yu Liu, Pu Wang, Tao Lin, Min Deng, and Haifeng Li. 2020. T-GCN: A Temporal Graph Convolutional Network for Traffic Prediction. *IEEE Transactions on Intelligent Transportation Systems* 21, 9 (2020), 3848–3858.

[32] Pengpeng Zhao, Anjing Luo, Yanchi Liu, Fuzhen Zhuang, Jiajie Xu, Zhixu Li, Victor S. Sheng, and Xiaofang Zhou. 2020. Where to Go Next: A Spatio-Temporal Gated Network for Next POI Recommendation. *IEEE Transactions on Knowledge and Data Engineering* (2020), 1–1.

[33] Hua Zhou and Lexin Li. 2014. Regularized matrix regression. *Journal of the Royal Statistical Society. Series B, Statistical methodology* 76, 2 (March 2014), 463—483. https://doi.org/10.1111/rssb.12031