Parallel Computing: Technology Trends
I. Foster et al. (Eds.)

doi:10.3233/APC200025

A Scalable Approach to Econometric Inference

Philip Nadler [a] Rossella Arcucci [a] and Yi-Ke Guo [a]
[a] *Data Science Institute, Imperial College London*

Abstract. We propose a novel approach combining vector autoregressive models and data assimilation to conduct econometric inference for high dimensional problems in cryptocurrency markets. We label this new model TVP-VAR-DA. As the resulting algorithm is computationally very expensive, it mandates the introduction of a problem decomposition and its implementation in a parallel computing environment. We study its scalability and prediction accuracy under various specifications.

Keywords. Econometrics, Inference, Cryptocurrencies, Data Assimilation, Parallel Computing, Problem Decomposition

1. Introduction

The ongoing digitisation of the economy has led to an abundance of data. Some institutions such as the Bank of England [8] argue that it is only a matter of time until the nature of economic data changes and will be as granular and continuous as is already the case for traffic and weather data. A very prominent example is the emergence of cryptocurrency markets, in which every single transaction is traceable on a publicly available ledger and is updated on a minute by minute bases [14,4], with the ledgers data size continuously growing. Hence, this new economic phenomenon provides an ideal example for the digitisation of economic data and also exposes the limitations of econometric inference at scale.

Due to these developments, economic modeling needs to take computational constraints more into account and incorporate sensible ways to speed up and enable larger scale analysis without the loss of accuracy. Whilst uncommon in economics, this resembles the approach in computational sciences in which researchers often face the problem of trade-offs between accuracy and computational efficiency. Our work therefore aims at bridging this gap by presenting a mathematical formulation of a generalisable economic model in which we explicitly account for scalability and parallelisation as well as evaluate and compare accuracy.

The model class we consider are vector autoregressive models with time varying parameters (TVP-VAR) [9]. TVP-VARs are time series models often used for economic policy analysis and forecasting, describing the interrelationship of economic variables dynamically over time and also model latent economic states whose structure economists exploit

This work is supported by Worley Parsons

for policy analysis [5]. Most conducted studies are small scale [15] because the inclusion of time varying latent states with multiple lagged coefficients leads to parameter matrices whose size increases with the square of the number of variables in the model. This makes CPU time requirements highly nonlinear with respect to the number of variables and thus calls for parallel computing methods [7].
Due to the fact that economists need to study and compare various model parameterisations such as lag length selection, it is important that such models can be computed in reasonable amounts of time without losing accuracy. We thus introduce a domain decomposition approach for significant performance gains. The parallelisation we introduce is on the mathematical formulation of the problem. We decompose the datasets in time windows with possible overlaps and we propose a mathematical model formulation based on domain decomposition. We present and study the performance of the parallel algorithm implemented on a distributed computing architecture. Also, the algorithm's scalability is studied taking into account the execution time.
To study the models performance, we use generated data as well as a dataset that consists of up to 121 identified exchanges on the Bitcoin blockchain. The data is available in multiple frequencies and spans multiple years. Each exchange is represented as an on-chain address cluster of hundreds of addresses. Thus for each exchange, multiple time series are of interest: the aggregate amount of Bitcoin they hold over time, the number of inflows and outflows, as well as transaction rate within a given timeframe. We further expand this with available off-chain data such as price and trading volume of exchanges. This allows for investigation of economic questions such as if a large inflow of Bitcoins has a negative effect of the price of a given exchange, i.e if the rules of supply and demand hold for individual exchanges.

2. The Economic Model

In order to conduct economic inference we combine a TVP-VAR with a Data Assimilation (DA) Framework. DA is an uncertainty quantification technique used to incorporate observational data into a prediction model ([2]) in order to improve numerical forecasted results. As previously discussed, TVP-VAR is a time series model for the analysis of economic systems using latent state variables. The model is outlined in Eq. 1 to 3. We propose a new model which combines TVP-VAR with DA, naming it TVP-VAR-DA. Due to the high dimensionality of the model and the number of state variables used to describe cryptocurrency markets, the TVP-VAR-DA is a large scale problem that should be solved in suitable acceptable time. It mandates the use of parallel computing environments. In this paper, we formally address the parallelism problem by defining the parallel TVP-VAR-DA model based on a problem decomposition approach. In fact, as claimed in [7], the partitioning problem (i.e, decomposability: to break the problem into small enough independent less complex subproblems) is a universal source of scalable parallelism.
To exemplify the model, the basic structure of the univariate TVP-VAR model is:

$$\tilde{y}_t = \tilde{x}_t \phi_t + \tilde{\varepsilon}_t \tilde{\sigma}_t \tag{1}$$

$$\phi_t = \phi_{t-1} + \tilde{v}_t \tag{2}$$

$$log(\tilde{\sigma}_t^2) = log(\tilde{\sigma}_{t-1}^2) + \tilde{\xi}_t \tag{3}$$

where scalar $\tilde{y}_t$ is the time t value of the dependent variable for $t = 1,..,T$, $\tilde{x}_t$ is a $1 \times q$ vector of predictors and lagged dependent variables and ϕ_t the coefficient vector of corresponding dimension. The errors follow the distributions: $\tilde{\varepsilon}_t \sim N(0,1)$, $\tilde{\nu}_t \sim N(0,\tilde{Q}_t)$ and $\tilde{\xi}_t \sim N(0,\tilde{R}_t)$. Eq. 1 describes the relationship of an economic variable with other series of interest. The evolution of this relationship over time is given by Eq. 2, whereas Eq. 3 models changes in the volatility of variables over time.

Generalisation In order to generalise the model in Eq. 1-3 to arbitrary lag length and number of variables under consideration, it is necessary to re-express the model in a more general matrix notation:

$$y_t = \Phi_1 y_{t-1} + ... + \Phi_l y_{t-l} + \mu_t + \varepsilon_t \sigma_t \tag{4}$$

where y_t is now a $q \times 1$ vector of variables we wish to study, μ_t a vector of means and Φ_l is a $q \times q$ coefficient matrix. σ_t is of the same dimension as y_t whereas ε_t is a diagonal matrix. To write this compactly, we define $X_t = [y'_{t-1},...,y'_{t-l},1]'$ and $\Phi = [\Phi_1,...,\Phi_l,\mu]'$. Define $K = (ql+1)$ as the product of variables q and lag length l including a constant. Thus X_t is of dimension $K \times 1$ and Φ of dimension $K \times q$.
Vectorisation of Φ is performed in order to include variable lag length which is necessary for policy analysis while preserving markovian properties of the resulting state-space model. Therefore define $x_t = I \otimes X_t$ using Kronecker product $\otimes$, where I is the identity matrix and define $\beta_t = vec(\Phi)$, where $vec()$ stacks the columns of a matrix. This allows us to rewrite the VAR in compact notation similar to the univariate case and express it in state space form:

$$y_t = x'_t \beta_t + \varepsilon_t \sigma_t \tag{5}$$

$$\beta_t = F\beta_{t-1} + \nu_t \tag{6}$$

$$log(\sigma_t^2) = log(\sigma_{t-1}^2) + \xi_t \tag{7}$$

where β_t is now of dimension $qK \times 1$ with corresponding transition matrix F and x'_t of dimension $q \times qK$, where $log(\sigma_t^2)$ in Eq. 7 scales accordingly. The terms ε_t, ν_t and ξ_t are zero mean errors and will be denoted in the linearization and algorithm section for clarity. Distributional assumptions remain the same with $\varepsilon_t \sim N(0,I)$, $\nu_t \sim N(0,Q_t)$ and $\xi_t \sim N(0,R_t)$ with the covariance matrices scaling accordingly. The above equations form a state space model with a stochastic volatility term, that can be approximated via a variety of filtering techniques (see e.g. [10], [3], [11]). Our TVP-VAR-DA methodology is able to produce point forecasts in high-dimensional settings, while also generating the history of latent state variables for analysis, taking into account the interdependencies of many variables, incorporating more information and thus reducing omitted variable bias. The solution to this model which takes into account parallelisation due to domain decomposition is described in the next section.

3. The Parallel Model

The optimal values of parameters β_t and σ_t are obtained via the following DA methodology:

$$\beta_t, \sigma_t = \underset{\beta,\sigma}{\operatorname{argmin}}\, J(\beta, \sigma) \tag{8}$$

with

$$J(\beta, \sigma) = \left|\left|\beta - \beta_{t-1} + \nu_t\right|\right|_{Q_t^{-1}} + ||log\left(\frac{\sigma_{t-1}^2}{\sigma^2}\right) + \xi_t||_{R_t^{-1}} \tag{9}$$

In order to partition the problem, let $\Omega \subset \mathbb{R}^M$ denote the domain[2]. $DD(\Omega)$ then constitutes a partitioning with overlaps such that $DD(\Omega) = \{\Omega_i\}_{1,\ldots,p}$ with $\Omega_i = (x_{i,t}, y_{i,t})_{t=0,\ldots T_i}$ where $T_i < T$, $\Omega_i \subset \mathbb{R}^{r_i}$, $r_i \leq M$ and for $i = 1, \ldots, p$ is such that $\Omega = \cup_{i=1}^{p} \Omega_i$ with $\Omega_i \cap \Omega_j = \Omega_{ij} \neq \varnothing$. This allows us to restate the problem as:

$$J_i(\beta_i, \sigma_i) = ||\beta_i - \beta_{i,t-1} + \nu_{i,t}||_{Q_{i,t}^{-1}} + ||log\left(\frac{\sigma_{i,t-1}^2}{\sigma_i^2}\right) + \xi_{i,t}||_{R_{i,t}^{-1}} \tag{10}$$

where furthermore $y_{i,t}$, $x_{i,t}$ are the restrictions of the corresponding quantities in (5), (6), (7) and (8) on the subdomains which constitute the decomposition. In order to minimise the function in (10) on each subdomain, we pose $\nabla(J_i(\beta_i, \sigma_i)) = 0$ and we solve the normal equations which conduct to a modified version of the Kalman Filter [2,12]:

$$\beta_{i,t} = \beta_{i,t-1} + K_t(y_{i,t} - x_{i,t}\beta_{i,t-1}), \quad t = 1, \ldots, T_i \tag{11}$$

$$log(\sigma_{i,t}^2) = log(\sigma_{i,t-1}^2) + K_{i,t}^*(y_{i,t}^* - log(\sigma_{i,t-1}^2)), \quad t = 1, \ldots, T_i \tag{12}$$

where

$$K_{i,t} = Q_{i,t-1}x_{i,t}'(x_{i,t}Q_{i,t-1}x_{i,t}' + \sigma_{i,t})^{-1} \tag{13}$$

and

$$K_{i,t}^* = R_{i,t-1}(R_{i,t-1})^{-1}, \quad y_{i,t}^* = log((y_{i,t} - x_{i,t}\beta_{i,t-1})^2) - ln(\xi^2) \tag{14}$$

as described in detail in Algorithm 1 applied to each subdomain Ω_i where for ease of notation we drop subdomain subscript i. The Algorithm depicts the main steps and also includes a condition $\bar{\sigma}$, in which $log(\sigma_{i,t}^2)$ can be modeled as time varying or constant over time.

4. Dataset

We conduct our experiments on two datasets. The first one is generated artificially in order to check model specifications and to generate more well-behaved data. The second dataset consists of on-chain as well as off-chain data of the cryptocurrency market.

[2]The training data set is defined as $\mathscr{D} = \{(x_i, y_i), i = 1, \ldots, M\}$

Algorithm 1 The TVP-VAR-DA algorithm $A(\Omega_i, p)$ based on domain decomposition for each subdomain Ω_i, where $DD(\Omega) = \{\Omega_i\}_{1,\ldots,p}$

Input $\Omega_i = (x_{i,t}, y_{i,t})_{t=0,\ldots T}$
Initialise Priors Q_0, R_0, ξ, F, β_0, σ_0
specify $\bar{\sigma}$
for for t=1...T **do**
Prediction Step
    $\beta_{t-1} = F\beta_{t-1}$ ▷ Predicted Mean
    $Q_{t-1} = FQ_{t-1}F'$ ▷ Predicted Variance
    $\eta_{t-1} = y_t - x_t\beta_{t-1}$ ▷ Forecast Error
Stochastic Volatility
    if $\sigma_t \neq \bar{\sigma}$ **then** ▷ Define Constant or Stochastic Volatility
        Construct $y_t^* = log((y_t - x_t\beta_{t-1})^2) - ln(\xi^2)$
        $K_t^* = R_{t-1}(R_{t-1})^{-1}$ ▷ Gain Matrix
        $log(\sigma_t^2) = log(\sigma_{t-1}^2) + K_t^*(y_t^* - log(\sigma_{t-1}^2))$ ▷ Posterior Mean of $log(\sigma_t^2)$
        $R_t = (I - K_t^*)R_{t-1}$ ▷ Posterior Variance of $log(\sigma_t^2)$
    else
        $\sigma_t = \bar{\sigma}$
    end if
Updating Step
    $f_t = x_tQ_{t-1}x_t' + \sigma_t$ ▷ Forecast Variance
    $K_t = Q_{t-1}x_t'(x_tQ_{t-1}x_t' + \sigma_t)^{-1}$ ▷ Gain Matrix
    $\beta_t = \beta_{t-1} + K_t(y_t - x_t\beta_{t-1})$ ▷ Posterior Mean of β_t
    $Q_t = (I - K_tx_t)Q_{t-1}$ ▷ Posterior Variance of β_t
end for

Dataset 1 In order to verify the correct tracking of the state equation we create an artifical dataset that is generated according to the following underlying specifications:

$$y_t = X_t'\gamma_t + e_{1,t} \tag{15}$$

$$\gamma_t = \gamma_{t-1} + e_{2,t} + e_{3,t} \tag{16}$$

Where γ is the time varying state variable, and e_1, e_2, e_3 are $\sim i.i.d\, N(0, e_i)$ white noise processes. Data matrix X_t is generated in a similar fashion as standardised i.i.d process. γ_t and y_t are generated as outlined above.
The dimensions of the dataset are created to match the real dataset and are of problem size $M = 1100$, $M = 8200$, $M = 27900$ and $M = 655600$ respectively, these correspond to the number of entries in x_t for each timestep which resemble combinations of variables and lag lengths included in the model.

Dataset 2 The main dataset of interest is the cryptocurrency market dataset. There are two types of data, one labeled on-chain data is the data derived directly from the Bitcoin blockchain. In short, the blockchain can be described as a distributed ledger system in which a multitude of nodes receive and process transactions created by other actors. Every transaction leads from one public address to another. All nodes in the network then synchronise the state of the ledger to form a global consensus on which address owns how much Bitcoin. The network is pseudonymous: all addresses and their balance are

visible through time, although it is not directly visible by whom the address is controlled. Some researchers such as [13] created and verified heuristics in which it is possible to link addresses to entities such as exchanges and other services. Interested readers are referred to other sources such as [14] or [6]. Based on this heuristic the on-chain dataset consists of 121 exchanges which are partially identified on the blockchain by the authors of https://www.walletexplorer.com/. Given this we calculate its hourly balance and the number of inflows as well outflows of Bitcoins. The second part of that set which we label off-chain data consists of the prices as well as volume of Bitcoin on selected exchanges which were identified in the first dataset. These sets constitute matrices x_t and y_t in Eq. 5 whereas the latent state variable β_t give an economic interpretation of the relationship of the underlying dynamics of off-chain and on-chain data. Data is available up to minute frequency and spans from 2014 until early 2019. Due to the noisy nature of the data we focus on hourly frequency to show the scalability properties of our model. For all experiments we add constants for numerical stability in the algorithm and interpolate missing values to make forecasting results more comparable. Fig. 1 displays the on-chain Bitcoin balance as well as the off-chain trading volume of the Kraken.com exchange for a selected period.

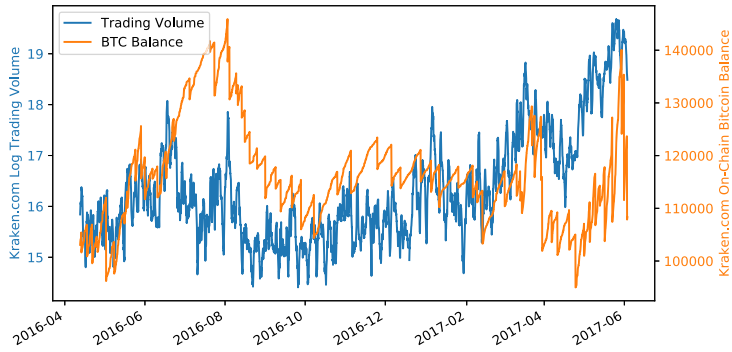

Figure 1. Example of on-chain and off-chain data: The amount of Bitcoin a set of addresses associated to an exchange on the blockchain hold as well as the trading volume on the exchange itself

5. Computational Time and Scalability

We evaluate the performance of Algorithm 1 on *Dataset 1* as we know this does not affect the generality of our study. We computed the values of execution time and we evaluated the scale-up factor. The scale-up factor for a problem decomposition of the function (10) is defined as [1]:

$$S_p(M, p_M) = \frac{T_{p_M}}{T_p}, \tag{17}$$

where p denotes the number of running processors, M denotes the problem size and p_M is the minimum number of processors used for the problem of size M. Table 1 shows the values of the execution time of Algorithm 1 for a problem of size $M = 1100, 8200, 27900, 65600$ created to match the real dataset which resemble combinations of variables and lag lengths included in the model. The experiments are run on a cluster with multiple 2.40GHz Intel Core i7-6700 CPUs and 256GB RAM available.

Problem Size	$M = 1100$	$M = 8200$	$M = 27900$	$M = 65600$
Processors	T_p (seconds)	T_p (seconds)	T_p (seconds)	T_p (seconds)
2	10.47×10^0	19.70×10^1	-	-
4	4.93×10^0	99.33×10^0	98.61×10^1	50.60×10^2
8	2.46×10^0	53.53×10^0	50.65×10^1	25.55×10^2
16	1.25×10^0	29.26×10^0	31.78×10^1	13.63×10^2
32	0.99×10^0	18.34×10^0	14.41×10^1	10.87×10^2

Table 1. Generated data table including stochastic volatility displaying computational time. Experiments ran on Imperial Cluster CX2

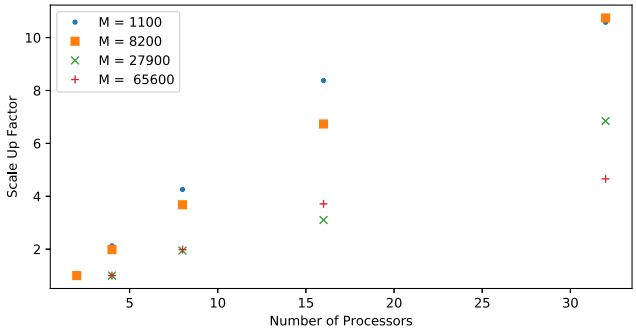

Figure 2. Values of scale-up factor for problem size $M = 1100, 8200$ with $p_M = 2$ and $M = 27900, 65600$ with $p_M = 4$

6. Forecast Comparison

This section evaluates the performance of the model in terms of forecasting. Mean squared forecast errors (MSFE) as well as mean absolute forecast errors (MAFE) are used to compare model quality in-sample. This is based on direct point forecasts, evaluating the residuals of predicted and realised values:

$$MSFE = \sum_{n=0}^{N} \left(\frac{\sum_{\tau=\tau_0}^{T-h} (y_{t,n}^r - \hat{y}_{t,n})^2}{T-h-\tau_0+1} \right) \quad (18) \qquad MAFE = \sum_{n=0}^{N} \left(\frac{\sum_{\tau=\tau_0}^{T-h} \left| y_{t,n}^r - \hat{y}_{t,n} \right|}{T-h-\tau_0+1} \right) \quad (19)$$

where $h = 1$ is the forecast horizon, evaluated before updating the state parameters and $\tau_0 = 1$, the starting date of the forecasting exercise. $y_{t,n}^r$ represents the actual realisation of a variable, while $\hat{y}_{t,n}$ represents the corresponding point forecast. The results in the tables are reported as averages over all included time series respectively, indicated by index n.

Fig. 3 shows the MSFE for one of the experiments using real data. Comparing both axes shows how after the TVP-VAR-DA assimilation the forecasting error decreases by multiple magnitudes, validating the predictive performance of the model.

The results for all experiments are reported in Table 2 and 3. We plot the ratio of forecasting errors before $\hat{y}_{t|t-1}$ and after $\hat{y}_{t|t}$ assimilation of observations, corresponding to forecasts generated via parameters in line 5 and 19 in Algorithm 1. The increase in forecasting accuracy is observable across all problem sizes for both generated and real

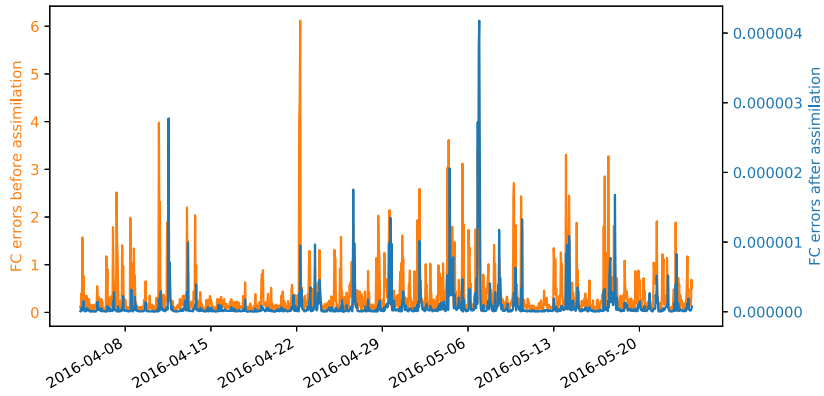

Figure 3. Comparison of forecasting errors before and after assimilation, depicting the average squared forecasting error at each timestep for problem size $M = 1100$

Problem Size	$M = 1100$		$M = 8200$		$M = 27900$		$M = 65600$	
Processors	MSFE	MAFE	MSFE	MAFE	MSFE	MAFE	MSFE	MAFE
2	$2.26e^{-5}$	$1.98e^{-4}$	$3.65e^{-6}$	$8.49e^{-5}$	-	-	-	-
4	$3.38e^{-5}$	$3.55e^{-4}$	$6.94e^{-6}$	$1.64e^{-4}$	$3.27e^{-6}$	$1.18e^{-4}$	$1.44e^{-6}$	$7.97e^{-5}$
8	$5.18e^{-5}$	$7.02e^{-4}$	$8.87e^{-6}$	$2.95e^{-4}$	$3.82e^{-6}$	$2.07e^{-4}$	$1.77e^{-6}$	$1.45e^{-4}$
16	$6.65e^{-5}$	$1.26e^{-3}$	$1.13e^{-5}$	$5.31e^{-4}$	$4.27e^{-6}$	$3.47e^{-4}$	$2.11e^{-6}$	$2.25e^{-4}$
32	$7.33e^{-5}$	$2.08e^{-3}$	$1.12e^{-5}$	8.02^{-4}	$3.97e^{-6}$	$4.95e^{-4}$	$2.24e^{-6}$	$3.35e^{-4}$

Table 2. Generated data table including stochastic volatility displaying accuracy ratios before and after assimilation

Problem Size	$M = 1100$		$M = 8200$		$M = 27900$		$M = 65600$	
Processors	MSFE	MAFE	MSFE	MAFE	MSFE	MAFE	MSFE	MAFE
2	$7.11e^{-6}$	$8.07e^{-4}$	$1.46e^{-5}$	$3.36e^{-3}$	-	-	-	-
4	$1.11e^{-5}$	$8.22e^{-4}$	$1.46e^{-5}$	$3.35e^{-3}$	$1.14e^{-5}$	$2.55e^{-3}$	$1.60e^{-6}$	$2.26e^{-3}$
8	$1.95e^{-5}$	$8.52e^{-4}$	$1.47e^{-5}$	$3.35e^{-3}$	$1.15e^{-5}$	$2.56e^{-3}$	$3.39e^{-6}$	$6.96e^{-3}$
16	$5.29e^{-5}$	$9.32e^{-4}$	$1.81e^{-5}$	$3.38e^{-3}$	$1.46e^{-5}$	$2.59e^{-3}$	$1.07e^{-6}$	$1.37e^{-2}$
32	$9.11e^{-5}$	$1.06e^{-3}$	$2.1e^{-5}$	$3.42e^{-3}$	$1.73e^{-5}$	$2.63e^{-3}$	$5.09e^{-6}$	$9.21e^{-3}$

Table 3. Real data table including stochastic volatility displaying accuracy ratios before and after assimilation

data, although more pronounced for the MSFE metric. By introducing the domain decomposition we see a slight decrease in accuracy when then number of processors increase but still improve forecasts by similar orders of magnitude. We use results of adjunct subdomains with no overlap to provide a lower bound for accuracy. The relative increase in errors due to domain decomposition is decreasing in larger scale problems compared to small ones. In Table 2 it is observable that, across all processors specifications, with increasing problem size the MSFE ratio decreases consistently, meaning that the larger the problem, the larger the increase in forecasting accuracy after assimilation. A similar patterns holds for the MAFE ratios. In contrast, Table 3 shows higher forecasting error ratios across all problem sizes except for the smallest. The forecasting ratio is performing better in the case of synthetic data since the data generation algorithm

is specified in such a way that it aligns with model assumptions of the TVP-VAR-DA, whereas using real cryptocurrency data, the errors are more pronounced due to the more erratic nature of the data.

7. Economic Results

As a case study, we analyse latent state parameters over the summer of 2015 and 2016 which represent the interaction of on-chain and off-chain movements of the Kraken.com exchange. Figure 4 displays selected entries from state vector β_t over time. In particular the predictive effect of price and trading volume changes on Bitcoin flows. In the top figure it is observable that in August 2015 changes in trading volume are associated with a decrease in bitcoin balance and thus outflow of bitcoins whereas in the same period for 2016 this relationship has nearly vanished. This is evidence that over time the on-chain activity has become more decoupled from actual price action on exchanges, which might be driven by other factors such as sentiment. It it also observable that around the change from August to September price changes have a significant positive effect on the inflow of Bitcoin, providing evidence for seasonal cycles in how the flow of Bitcoins affect exchanges.

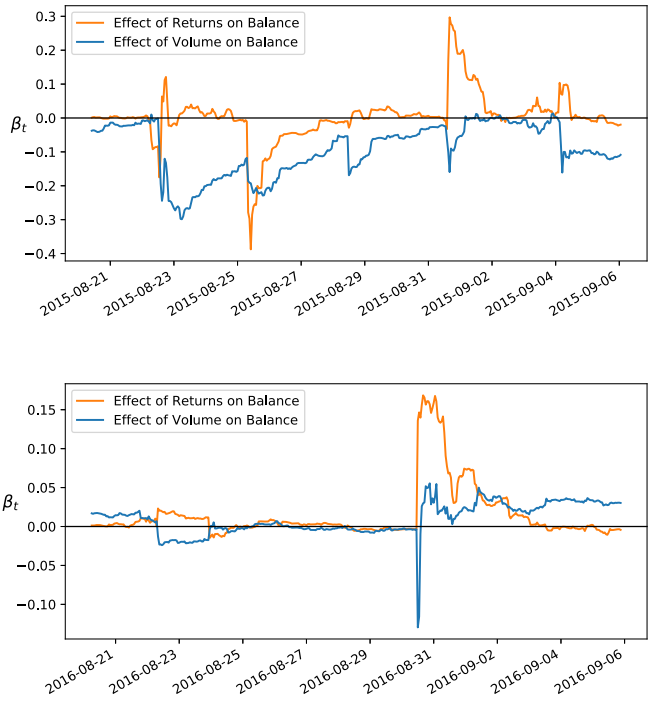

Figure 4. 2015 and 2016 hourly evolution of state variables of Kraken bitcoin blockchain balance and its relation to Kraken price and trading Volume changes.

8. Conclusion

We introduced a new model type that is capable of doing econometric inference at scale by leveraging a data assimilation approach. We show how already in the formulation of the problem we can take into account domain decomposition and parallelisation, showing how similar to computational sciences, economists can increase computational feasibility without sacrificing too much accuracy. We compared model performance and showed that the model generated latent economic variables which help to analyze economic phenomena such as the interaction of entities in cryptocurrency markets. Future work can include additional parallelisations of the algorithm or inference techniques which are unfeasible in standard environments, such as doing a fully Bayesian treatment of the TVP-VAR-DA model, as well as doing real-time forecasting and inference with high frequency data at scale.

References

[1] R. Arcucci, L. D'Amore, L. Carracciuolo, G. Scotti, and G. Laccetti. A decomposition of the tikhonov regularization functional oriented to exploit hybrid multilevel parallelism. *International Journal of Parallel Programming*, 45(5):1214–1235, 2017.

[2] M. Asch, M. Bocquet, and M. Nodet. *Data assimilation: methods, algorithms, and applications*, volume 11. SIAM, 2016.

[3] R. Bhar and D. Lee. Comparing estimation procedures for stochastic volatility models of short-term interest rates. *Available at SSRN 1160659*, 2009.

[4] R. Böhme, N. Christin, B. Edelman, and T. Moore. Bitcoin: Economics, technology, and governance. *Journal of Economic Perspectives*, 29(2):213–38, 2015.

[5] F. Canova and L. Gambetti. Structural changes in the us economy: Is there a role for monetary policy? *Journal of Economic dynamics and control*, 33(2):477–490, 2009.

[6] K. Christidis and M. Devetsikiotis. Blockchains and smart contracts for the internet of things. *Ieee Access*, 4:2292–2303, 2016.

[7] G. C. Fox, R. D. Williams, and G. C. Messina. *Parallel computing works!* Elsevier, 2014.

[8] A. G. Haldane. Will big data keep its promise? *Speech at the Bank of England Data Analytics for Finance and Macro Research Centre, King's Business School*, 2018.

[9] J. D. Hamilton. *Time series analysis*, volume 2. Princeton university press Princeton, NJ, 1994.

[10] A. Harvey, E. Ruiz, and N. Shephard. Multivariate stochastic variance models. *The Review of Economic Studies*, 61(2):247–264, 1994.

[11] S. J. Julier and J. K. Uhlmann. New extension of the kalman filter to nonlinear systems. In *Signal processing, sensor fusion, and target recognition VI*, volume 3068, pages 182–194. International Society for Optics and Photonics, 1997.

[12] R. E. Kalman. A new approach to linear filtering and prediction problems. *Journal of basic Engineering*, 82(1):35–45, 1960.

[13] S. Meiklejohn, M. Pomarole, G. Jordan, K. Levchenko, D. McCoy, G. M. Voelker, and S. Savage. A fistful of bitcoins: characterizing payments among men with no names. In *Proceedings of the 2013 conference on Internet measurement conference*, pages 127–140. ACM, 2013.

[14] S. Nakamoto et al. Bitcoin: A peer-to-peer electronic cash system. 2008.

[15] G. E. Primiceri. Time varying structural vector autoregressions and monetary policy. *The Review of Economic Studies*, 72(3):821–852, 2005.