Estimating time-varying networks with a state-space model

Shaowen Liu* Massimiliano Caporin† Sandra Paterlini‡

September 21, 2022

Abstract

Two state-space representations, also known as state-space models (SSMs), are proposed to estimate dynamic spatial relationships from time series data. At each time step, the weight matrix, which captures the latent state, is updated in the context of a spatial autoregressive process. Specifically, two types of SSMs are considered: the first one identifies the spatial effects in the form of multivariate regression, where the higher orders of the spatial matrix are introduced to the regression coefficient, while the second one updates the spatial matrix taking advantage of the likelihood function of a spatial autoregressive (SAR) model. Different filtering algorithms are proposed to estimate the latent state. The simulation results show that the first state-space representation performs better in both lower-dimensional and higher-dimensional cases, while the performance of the second representation is sensitive to the state dimension. In a real-world case study, the time-varying weight matrices are estimated with weekly

*Department of Statistical Sciences, University of Padova

†Department of Statistical Sciences, University of Padova; corresponding author, Via. C. Battisti, 241, 35121 Padova, Italy; ph. +39-049-827-4199, email: massimiliano.caporin@unipd.it.

‡Department of Economics and Management, University of Trento

credit default swap (CDS) data for 16 banks, and the results show that the methods can identify communities that are consistent with the country-driven partition.

Keywords: state-space model, dynamic network, spatial dependence, sequential filters.

JEL Codes: C32, C33 and C51.

1 Introduction

Panel spatial econometrics has seen increasing popularity and has been applied in a wide range of fields, from regional science to social science (Anselin, 2010). By specifying and estimating the latent connections among the observations collected from different subjects or regions, one could gain a more in-depth understanding of the underlying relationships or network structures. However, there are two main problems that arise when applying spatial econometric models in practice. First, the weight matrix has to be prespecified based on certain prior knowledge, and an inappropriate spatial weight matrix can introduce extra model uncertainties (LeSage and Pace, 2014; LeSage and Fischer, 2008; Getis and Aldstadt, 2004). Second, conventionally, in spatial models, the weight matrix is always assumed to be time invariant. This might be true for the cases where the geographical/physical distance between subjects is fixed and known a priori; but when it comes to other fields, such as social sciences, the relationship between subjects could be time varying. For instance, in the field of finance, the network effects across financial institutions, which are strongly connected to systemic risk, could evolve in a dynamic fashion over time (Billio et al., 2016; Alter and Beyer, 2014; Elliott et al., 2014; Diebold and Yılmaz, 2014; Billio et al., 2012) in response to a variety of causes, such as changing debt relationships between banks, or newly listed risky assets in the balance sheet of a bank.

The estimation of spatial model is well discussed in the literature, but, as mentioned

above, the weight matrix is assumed to be known or could be parametrised. Some commonly used estimation methods include maximum likelihood (Ord, 1975; LeSage and Pace, 2007) and, generalised method of moments (Kelejian and Prucha, 1999). There are also approaches for estimating the weight matrix on a bivariate basis, i.e. estimating the presence of each edge. As for estimating the full weight matrix, one of the early proposals is reported in Meen (1996), where the ordinary least squares (OLS) method is applied for the matrix estimation. Bhattacharjee and Jensen-Butler (2013) estimate the full weight matrix based on the assumption that the matrix is symmetric. In recent years, several papers have proposed estimating the full weight matrix by two-stage least squares estimation; see, among others, Ahrens and Bhattacharjee (2015) and Lam and Souza (2019). When it comes to estimating the time-varying full weight matrix, to our knowledge, there have been few such attempts. One of the related methods is the time-varying Vector Auto Regression (VAR) (Kimura et al., 2003; Haslbeck et al., 2020), which can also estimate the network matrices dynamically, but, rather than the simultaneity that is assumed in spatial models, it assumes a cross-lagged dependency.

To tackle these problems, we propose innovative methods for estimating the time-varying spatial dependencies across subjects. Specifically, we integrate a spatial model into a state-space model such that the weight matrix, as the hidden state, can be updated at each step. There are different specifications for spatial models, such as the SAR model, the spatial error model (SEM) and the spatial Durbin model (SDM) (LeSage and Pace, 2009). Without losing generality, we illustrate our method with a simple spatial autoregression, which includes only a spatial lag term. One could easily extend the methods by integrating them into spatial models characterised by different levels of complexity.

In particular, we propose two state-space representations for updating simultaneous dependencies. In the first specification, the hidden weight matrix is updated as a coefficient matrix, while, in the second one, the spatial autoregression is fully applied, and the matrix

is updated by building on the model likelihood. Simulations show that, although the first specification interprets the simultaneous effects differently from a spatial model, by introducing higher order of the weight matrix it could efficiently deliver a reasonable and robust model estimation. By contrast, the second model works well in low-dimensional cases, but does not perform as well in higher dimensions. Moreover, the simulations show that the filter performances require relative long time series for appropriate identification of the latent network dynamic. After comparing the two representations using simulated data, we consider a real-case study. Specifically, we apply the models on a credit default swaps (CDS) dataset and show that the country-driven communities can be successfully identified from the estimated matrices.

The paper is organised as follows. In Section 2, we introduce the two state-space representations and the corresponding filtering algorithms for state estimation. The simulation results are then reported in Section 3, while the sensitivity analyses are presented in Section 4. In Section 5, we present the empirical results for weekly CDS data. Section 6 concludes the paper.

2 The model and the state-space representations

We start with a SAR process,

$$y_t = \rho \tilde{W} y_t + \epsilon_t, \quad \epsilon_t \sim \mathcal{N}(0, \sigma_\epsilon^2 I_D), \tag{1}$$

where y_t is a $D \times 1$ column vector representing D subjects observed at time t, ϵ_t is a vector of independent and homoscedastic errors with variance σ_ϵ^2 and I_D denotes a $D \times D$ identity matrix. Moreover, $\tilde{W}$ is a $D \times D$ matrix that captures the static dependencies between each pair of time series. The diagonal entries of $\tilde{W}$ are zeros, and the matrix is usually row

normalised. In addition, $\tilde{W}y_t$ is usually called spatial lag, echoing the term "time lag"used in the autoregressive model (AR). Typically, ρ is a scalar coefficient with $|\rho| < 1$, which measures the average strength of the spatial relationships. The constraint on ρ together with the row-normalization of $\tilde{W}$ ensure the invertibility of $I - \rho\tilde{W}$. As discussed in Caporin and Paruolo (2015), ρ could be generalised to a diagonal matrix; in this case, model (1) becomes more flexible, as it allows for defining distinct diagonal entries instead of an unique ρ, such that the strength of the spatial effects can be specified or estimated for each subject.

To estimate a sequence of time-varying weight matrices, i.e. W_t, we propose setting up a model with a proper state-space representation whose observation equation has a form similar to model (1). The difference is that, in model (1), W is a constant; but in our model, W_t is regarded as the state and, thus, has a dynamic evolution. Specifically, we propose the following model (in state-space form),

$$\begin{cases} y_t = W_t y_t + \epsilon_t, & \epsilon_t \sim \mathcal{N}(0, \sigma_\epsilon^2 I_D) \\ x_t = x_{t-1} + \eta_t, & \eta_t \sim \mathcal{N}(0, \sigma_\eta^2 I_{D_x}) \end{cases}, \tag{2}$$

where $\rho\tilde{W}$ from (1) is replaced with W_t, denoting the scaled full weight matrix that is to be updated at each step. In the second equation (the transition equation), the state x_t is the vectorisation of the off-diagonal elements of W_t, and D_x denotes the dimension of the state $D_x = D \times (D-1)$. Finally, ϵ_t and η_t are the vectors of independent and homoscedastic noises, with a constant variance σ_ϵ^2 and σ_η^2, respectively.

Notice that ρ_t is not estimated directly in (2) but by further specification, and both $\tilde{W}_t$ and ρ_t can be recovered from W_t. In this paper, we define the scalar ρ_t as the maximum row sum of W_t, i.e.

$$\rho_t = \max_i \sum_{j=1}^{D} W_{i,j,t},$$

such that the normalised weight matrix $\tilde{W}_t$ can be obtained by $\tilde{W}_t = W_t/\rho_t$. The maximum

row sum of $\tilde{W}_t$ is always equal to 1. The constraint $|\rho_t| < 1$ is still required to achieve the invertibility of $I - \rho_t \tilde{W}_t$. However, we find that it is not necessary to implement this constraint specifically for model identification. As shown in the simulation analysis, ρ_t always converges with W_t, thus excluding the possibility of exploding patterns, associated with singularities in the $I - \rho_t \tilde{W}_t$ sequence.

The structure of out model does not exclude residuals correlation. Applied works often suggest that spatial dependence could be also present among residuals, and this might be captured by adopting a SEM, where the spatial dependence matrix is usually known. In our setting, we are forced to introduce a hypothesis of orthogonality among the innovations for identification reasons. In fact, it would be, in general, impossible to disentangle a latent time varying spatial dependence typical of a SAR-type model from the correlation among innovations; the latter would be at least partly captured by the dynamic evolution of W_t that would thus proxy for the possible presence of correlation among the ϵ_t.

Searching for the optimal parameters for model (2) is not straightforward. On the one hand, one could simplify the model and regard the observation equation as a regression. This option might lead to biased estimates, as we will see below, but the model, in its simplest form, could easily be solved using a Kalman filter (KF). On the other hand, one could update W_t strictly according to the spatial model. The challenge along this direction is that the state is high-dimensional and only interacts with the covariance matrix of the observations. In the following subsections, both options are investigated by designing two alternative state-space representations, named $\mathcal{M}_1$ and $\mathcal{M}_2$.

2.1 The state-space representation $\mathcal{M}_1$

The first attempt to estimate the time-invariant weight matrix within a spatial error model by an OLS estimator was made by Meen (1996). Although OLS estimation for SAR is biased and inconsistent (Anselin, 2013; Kelejian and Prucha, 2002; Lee, 2002), the method provides

a simple solution for the state-space model (2) in the sense that the observation equation can be regarded as a normal multivariate regression and W_t can be updated dynamically as the coefficient matrix.

To better understand the relationship between the coefficient matrix in the context of least squares and the weight matrix in the spatial model, we resort to a linear model representation. In fact, the spatial model (1) can be more properly represented in the form of a linear model, i.e.,

$$y_t = Wy_t + \epsilon_t, \quad \epsilon_t \sim \mathcal{N}(0, R), \tag{3}$$

where R is a general covariance matrix that measures the correlations among the disturbance term. Thus, the spatial effects, which are driven solely by the weight matrix in a spatial model, might be captured by two matrices here, i.e. W and R in (3). To estimate time-varying spatial effects with an SSM while allowing for the presence of correlation among innovations at each time step t, one possible solution is to regard both matrices W_t and R_t as the state (thus allowing for possible time variation in R), such that they can be estimated dynamically. However, this would double the number of parameters without giving a clear solution to the possible joint effect between the two sources of contemporaneous interdependence. In fact, we might expect W_t to also capture the spatial effects that would be represented by R_t, *as we commented in the previous section when touching the identification issue.*

To tackle the problem, we first consider the following series expansion of the SAR model (Debreu and Herstein, 1953; Horn and Johnson, 2012)

$$y_t = (I - W)^{-1}\epsilon_t = (I + W + W^2 + W^3...)\epsilon_t. \tag{4}$$

Here the spatial effect is decomposed into elements with different orders of W, which represent the different orders of the spillover effect. For example, W^2 measures the indirect effect

of second-order neighbours (LeSage, 2008). Thus, the simultaneous effect can be interpreted as a combination of the direct effects, addressed by W, and the indirect effects, addressed by the power of W (i.e., W^k, with $k > 1$).

Inspired by the expansion form in (4), we propose augmenting the state-space model (2) with higher orders of W_t. This modification allows us to take into account the indirect effects and has benefits with regard to the estimation of W_t. The proposed state-space representation, denoted $\mathcal{M}_1$, becomes then as follows

$$\begin{cases} y_t = \sum_{k=1}^{K} W_t^k y_t + \epsilon_t, & \epsilon_t \sim \mathcal{N}(0, \sigma_\epsilon^2 I_D) \\ x_t = x_{t-1} + \eta_t, & \eta_t \sim \mathcal{N}(0, \sigma_\eta^2 I_{D_x}) \end{cases}, \tag{5}$$

where k is the order of W_t and $k = 1, 2, ..., K$. We note that R in (3) is not introduced in this state-space representation, as we expect that W_t and its higher orders in (5) can capture also the interdependence among disturbances. Thus, ϵ_t is assumed to be independent and identically distributed. We stress that such a model might be misspecified; this holds when data combine a time varying W_t with correlations in the residuals, i.e. R is not diagonal. Nevertheless, the introduction of powers of W_t would be of help in capturing the interdependence among observations when this is governed by two different forces.

The state-space representation (5) cannot be solved by the Kalman filter (KF) when $K > 1$, as the state-space is non-linear. We suggest using the Stochastic Ensemble Kalman Filter (SEKF) (Burgers et al., 1998; Houtekamer and Mitchell, 1998), which is a Monte Carlo method based on the framework of the Kalman filter.

A.1 and A.2 include pseudo-algorithms for KF and SEKF, respectively. Here, we briefly describe the filtering process specifically adopted for the state-space representation (5) under SEKF. First, we generate N initial samples from a multivariate normal distribution, i.e. $x_0^{(i)} \sim \mathcal{N}(\boldsymbol{\mu_0}, \sigma_0^2 I_{D_x})$ $(i = 1, ..., N)$, where $\boldsymbol{\mu_0}$ and σ_0^2 are the initial guesses for the state mean

and variance, respectively. We assume the same initial state mean, i.e. $\boldsymbol{\mu_0} = \mathbf{1}\mu_0$, with $\mathbf{1}$ being a vector of ones, and μ_0 being a scalar. The initial state variance is a diagonal matrix, as the states are assumed to be independent. Thus, each sample $x_0^{(i)}$ is a vector of dimension $D(D-1) \times 1$, which is then transformed into a sample weight matrix $W_0^{(i)}$. Similar to KF, each iteration of SEKF includes a forecast step and an update step. In the forecast step, we forecast the i-th state $\tilde{x}_t^{(i)}$ and the i-th observation $\tilde{y}_t^{(i)}$ by propagating the sample $x_{t-1}^{(i)}$, which is obtained from the previous step through the state equation and the observation equation given in (5), respectively. In the update step, the Kalman gain is estimated from sample forecasts, as both the covariance matrix $Cov(\tilde{x}_t, \tilde{y}_t)$ and the variance $Var(\tilde{y}_t)$ can be calculated from $\tilde{x}_t^{(i)}$ and $\tilde{y}_t^{(i)}$. Then, the sample forecasts $\tilde{x}_t^{(i)}$ are updated to $x_t^{(i)}$ with the Kalman gain, and the state mean x_t is equal to the sample mean, i.e. $\frac{1}{N}\sum_i^N x_t^{(i)}$. Note that, as SEKF propagates the samples through the state space and estimates the first two moments of the state at each iteration, it is capable of dealing both with linear and non-linear state-space representations, with the only requirement being that the sample state should be properly propagated. Like KF, SEKF also assumes that the distribution of the states is Gaussian.

In the following, we also label the representation adopting the maximum order of W_t, such as $\mathcal{M}_{1,K=1}$ for model $\mathcal{M}_1$ with $K = 1$, $\mathcal{M}_{1,K=2}$ for model $\mathcal{M}_1$ with $K = 2$, etc.

2.2 The state-space representation $\mathcal{M}_2$

The second approach we propose relies on the likelihood function of SAR.

Ord (1975) was the first to propose the use of maximum likelihood for the estimation of a spatial model; maximum likelihood asymptotic properties are summarised in Anselin (2013). In our case, we use the model likelihood within a filtering algorithm. We first rewrite the

state-space model (2) in a reduced form,

$$\begin{cases} y_t = (I_D - W_t)^{-1}\epsilon_t, & \epsilon_t \sim \mathcal{N}(0, \sigma_\epsilon^2 I_D) \\ x_t = x_{t-1} + \eta_t, & \eta_t \sim \mathcal{N}(0, \sigma_\eta^2 I_{D_x}) \end{cases}, \tag{6}$$

and denote it as representation $\mathcal{M}_2$.

To solve the filtering problems associated with $\mathcal{M}_2$, we consider Sequential Monte Carlo (SMC), which estimates the posterior distribution by importance sampling. The simplest form of SMC is the Standard Particle Filter (SPF), which assumes that the proposal distribution is identical to the prior distribution, i.e. $q_t(x_t|x_{t-1}) = p(x_t|x_{t-1})$, where $q_t(x_t|x_{t-1})$ denotes the proposal density at step t, and $p(x_t|x_{t-1})$ and $p(y_t|x_t)$ denote the prior density and the likelihood, respectively. Thus, the importance weights are proportional to the likelihoods, i.e.,

$$w_t^{(j)} \propto \frac{p(y_t|x_t^{(j)})p(x_t^{(j)}|x_{t-1}^{(j)})}{q_t(x_t^{(j)}|x_{t-1}^{(j)})} = p(y_t|x_t^{(j)}),$$

where $w_t^{(j)}$ denotes the importance weight of particle j at step t. In the case of $\mathcal{M}_2$, applying the SPF is straightforward as its log-likelihood function is well-studied, i.e.

$$l(W_t|y_t) = -\frac{D}{2}\log(2\pi) - \frac{D}{2}\log(\sigma_\epsilon^2) + \log|A_t| - \frac{1}{2\sigma_\epsilon^2}(A_t y_t)^T(A_t y_t), \tag{7}$$

where $A_t = I_D - W_t$, and $|A_t|$ denotes the determinant of A_t.

The SPF has, however, some shortcomings. In fact, the particle filter suffers from the so-called curse of dimensionality (Snyder et al., 2008). Even for a small network matrix, such as $D = 10$ or $D = 20$, the state dimension can be in the order of hundreds, which is a daunting task for the SPF. In the literature, different methods have been proposed to improve the performance of the particle filter in high-dimensional cases; see Khan et al. (2005) and Andrieu et al. (2010), among many others. Here, we choose to apply the variational

sequential Monte Carlo (VSMC), which has been recently introduced by Naesseth et al. (2018). VSMC estimates the state with a variational method, where the Kullback-Leibler (KL) distance monitors the divergence between the posterior distribution of the state and the proposed distribution. Different from KF and SEKF, which are online methods, VSMC is often defined as an offline method, as it optimises the filter over the whole dataset.

In this paper, we use a proposal distribution $q_t(x_t|x_{t-1}) = \mathcal{N}(\alpha_t + x_{t-1}, \sigma_q^2 I_{D_x})$, where α_t, a $D_x \times 1$ vector, is the only parameter to be optimised, while the variance σ_q^2 in the proposal distribution is set to be equal to the state variance σ_η^2 in the prior distribution. The optimisation process for the state-space representation (6) works as follows; to avoid confusion, from now on, we use the notation $a_{m,t}$, where $m = 1, ..., M$ and $t = 1, ..., T$, which indicate the optimisation step and the filtering step, respectively. First, we generate a series of initial proposal parameters $\{a_{m=0,t=1,...,T}\}$. Each element of vector $a_{0,t}$ is generated from a normal distribution $\mathcal{N}(0, 0.01)$. Then, we start the optimisation process, which includes M iterations. In each optimisation step, the sequence $\{a_{m,t=1,...,T}\}$ is optimised to $\{a_{m+1,t=1,...,T}\}$ by calculating the gradients of the surrogate evidence lower bound (ELBO), which can be obtained empirically by averaging the sample weights in SMC, i.e. $\sum_{t=1}^{T} \log\left(\frac{1}{N}\sum_{j=1}^{N} w_t^{(j)}\right)$. In particular, the weight of the j-th particle at step t is measured by

$$w_t^{(j)} \propto \frac{p(y_t|x_t^{(j)})p(x_t^{(j)}|x_{t-1}^{(j)})}{q_t(x_t^{(j)}|x_{t-1}^{(j)}, \alpha_{m,t})}.$$

In the end, the final state mean is estimated by running a SPF with optimal parameter $\{\alpha_{m=M,t=1,...,T}\}$. We refer to A.3 for details of the VSMC algorithm.

3 Simulations

To study the performance of the two state-space representations and of the associated filtering algorithms, we first generate the series $\{y_t, W_t\}$ and then examine whether the hidden matrices can be correctly estimated from the observation $\{y_t\}$. Specifically, two scenarios for the data generation are considered (see Algorithms 1 and 2 below). In the first scenario, W_t is designed to have some given patterns such that we could visualise the similarity between the generated and estimated weight matrices. In the second scenario, to mimic real-world instances, we let W_t evolve stochastically. In each scenario, two different matrix sizes are tested, i.e. $D = 10$ and $D = 20$. Note that, in both data generating processes, the observations are generated according to a SAR model. In theory, this would give $\mathcal{M}_2$ some advantage, as it integrates the same SAR process, while $\mathcal{M}_1$ implies a different way for estimating W.

As discussed in the previous section, $\mathcal{M}_1$ is estimated by SEKF, and $\mathcal{M}_2$ is estimated by VSMC. The performance of the filter can be influenced by its sample/particle size. After a preliminary testing (available upon request, see Section 4 for a sensitivity analysis), we decided to use the following parameter values, which lead to an optimal balance between the performance of the filters and their computational cost.

For SEKF, we apply $N = 10000$ samples. As for VSMC, we set the number of optimisation iterations as $M = 100$ (Algorithm 6) and the number of particles $N = 10$ (Algorithm 7). After obtaining the optimised proposal parameter α_M, the state is estimated by running the particle filter once with optimal α_M. For the final state estimation, we use $N = 10000$ particles.

3.1 Scenario 1: The patterned W_t

In the first scenario, we assume that W_t is sparse and the non-zero entries only exist in the blocks along the diagonal of the matrix. This is a baseline model that can allow to point out also visually the performance of the proposed methods. To that end, the initial weight matrix W_0 is generated in a specific block form

$$W_{10\times10} = \begin{bmatrix} W_{5\times5} & 0 \\ 0 & W_{5\times5} \end{bmatrix}, \quad W_{20\times20} = \begin{bmatrix} W_{5\times5} & 0 & 0 & 0 \\ 0 & W_{5\times5} & 0 & 0 \\ 0 & 0 & W_{5\times5} & 0 \\ 0 & 0 & 0 & W_{5\times5} \end{bmatrix}, \tag{8}$$

where $W_{D\times D}$ denotes the $D \times D$ weight matrix, and the entries for each diagonal block $W_{5\times5}$ are generated randomly by

$$\begin{cases} W_{ij} \sim U(0, 0.3), & i \neq j \\ W_{ij} = 0, & i = j \end{cases}, \tag{9}$$

where $U(a, b)$ denotes the uniform distribution between a and b. As there are four non-zero entries on each row of W_0 and the row sum must be less than one, we choose $b = 0.3$ such that the average row sum is expected to be 0.6.

The series of observations $\{y_t\}$ and the latent states $\{x_t\}$ are generated according to Algorithm 1, where $T = 500$, $\sigma_\epsilon^2 = 1$ and $\sigma_\eta^2 = 1 \times 10^{-2}$. x_t and x_0 are the vectorisation of the off-diagonal entries of W_t and W_0, respectively. In this scenario, W_t is, in fact, a series of perturbed W_0 such that the pattern is kept invariant.

We consider $B = 100$ runs for $D = 10$ and $D = 20$, separately. In fact, the evaluation for even a single run is based on a large set of data (500 observations and a big tensor containing the series of the hidden matrices, such as $20 \times 20 \times 500$ when $D = 20$). We apply

Algorithm 1 : The data-generating process for Scenario 1

Generate initial weight matrix W_0 with (8) and (9).
for $t = 1, ..., T$ **do**
  $x_t = x_0 + \eta_t, \eta_t \sim \mathcal{N}(0, \sigma_\eta^2 I_{D_x})$
  $y_t = (I - W_t)^{-1}\epsilon_t, \epsilon_t \sim \mathcal{N}(0, \sigma_\epsilon^2 I_D)$
end for
Output: $\{x_t, y_t\}, t = 1, ..., T$

the state-space representation $\mathcal{M}_1$ and $\mathcal{M}_2$ on each set of data and compare the estimated weight matrices, denoted as $\hat{W}_t$, with the generated matrices W_t. As input for the filters, we set the initial parameter as follows: $\mu_0 = 1/D$, $\sigma_0 = 1/D$, $\sigma_\epsilon^2 = 1$ and $\sigma_\eta^2 = 1 \times 10^{-6}$. Notice that a very small value is chosen for the state variance, which is different from the value applied in the data generating process, as, with smaller variance, the estimation is expected to be rather smooth and stable.

First, we evaluate the outcome by visualising the recovered pattern from a single run in case of $D = 10$. Figure 1 presents the generated matrix W_t and the estimated matrices $\hat{W}_t$ at the final step ($t = 500$). As described before, we generate the sparse weight matrix $W_{10\times10}$ with two communities, which could be easily identified in Panel (a). Similar patterns could also be observed in the other three panels (b, c and d); this implies that both models are effective, at least, in terms of extracting some key features of the hidden matrix.

Second, to compare the performance, we compute the mean squared errors (MSE), which represents the distance between the generated and the estimated weight matrices, defined as follows:

$$MSE(W_t, \hat{W}_t) = \frac{1}{D^2}\sum_{i=1}^{D}\sum_{j=1}^{D}(W_{i,j,t} - \hat{W}_{i,j,t})^2.$$

Panels (a) and (b) in Figure 2 report the average MSE at each step for $D = 10$ and $D = 20$, respectively. As the filter SEKF for $\mathcal{M}_1$ is an online method, we notice it requires around 100 steps to converge when $D = 10$ and around 200 steps when $D = 20$. The curves for $\mathcal{M}_1$ with a larger K (i.e., $K > 3$), which are available upon request, exhibit similar properties.

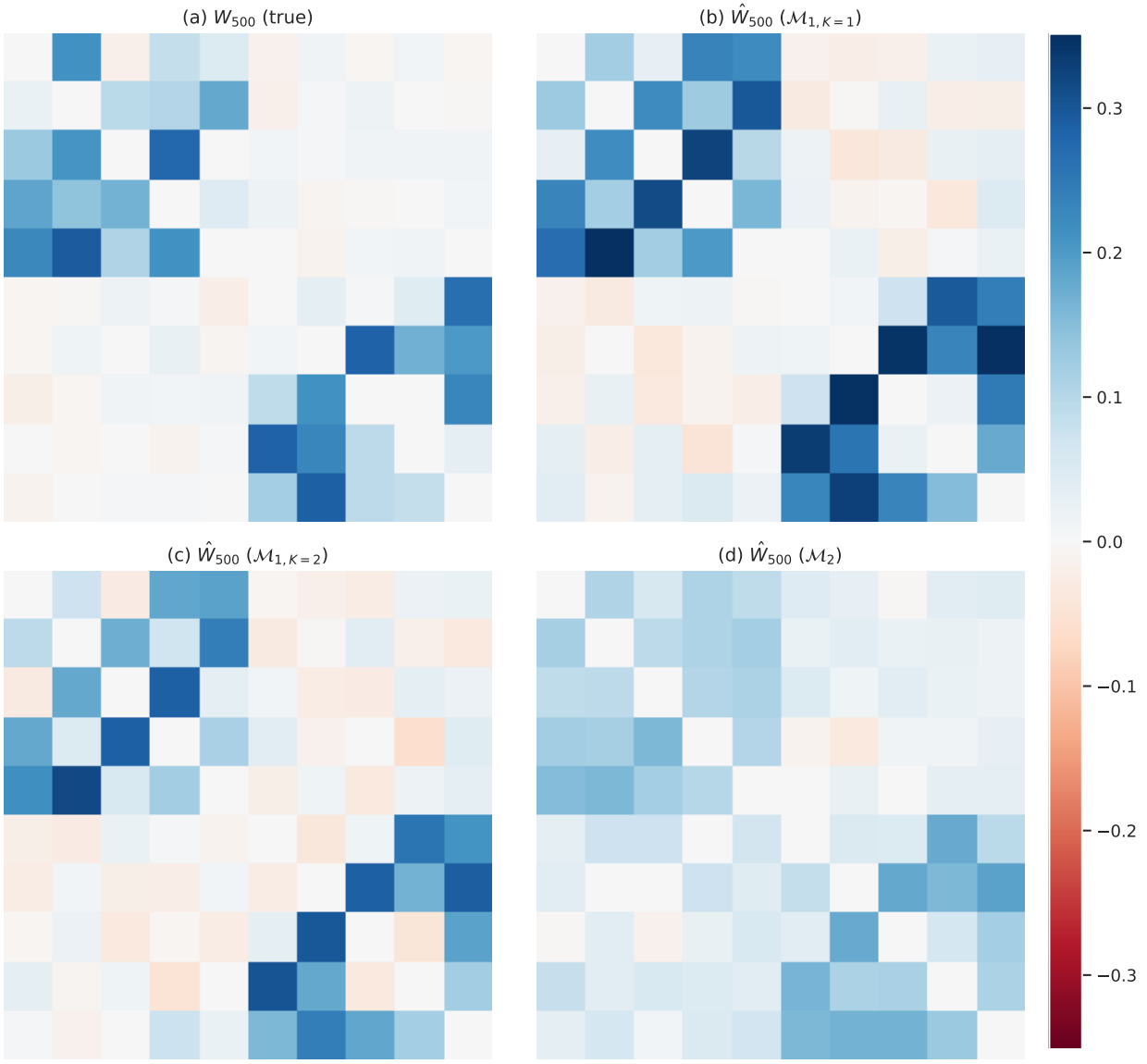

Figure 1: (a) the 500th generated weight matrix; (b) the 500th weight matrix estimated using $\mathcal{M}_{1,K=1}$; (c) the 500th weight matrix estimated using $\mathcal{M}_{1,K=2}$; (d) the 500th weight matrix estimated using $\mathcal{M}_2$.

In contrast, as $\mathcal{M}_2$ is solved by an offline method, the estimates (the blue line) are more stable over time.

Further, we summarise the quantiles of MSEs for each state-space model in Panels (c) and (d) in Figure 2. The MSEs obtained during the warm-up period (i.e., 200 steps) are

omitted, so there are $100 \times 300 = 30000$ MSEs available for each model. The representation $\mathcal{M}_1$ is evaluated for value of K up to five. Similar evidence is observed for $D = 10$ (Panels (a) and (c)) and $D = 20$ (Panels (b) and (d)).

First, Panels (c) and (d) indicate that $K = 2$ is an optimal truncating point for $\mathcal{M}_1$. Compared to $\mathcal{M}_{1,K=1}$, the performance of $\mathcal{M}_{1,K=2}$ improves significantly. But as K further increases, the performance stops improving; in contrast, the median MSE begins mildly targeting the opposite direction. The possible reason for this is that the third or higher order of W_t could only deliver marginal information, which is completely offset by the additional noise being introduced due to the increased model complexity.

Second, in terms of the median value, we see that $\mathcal{M}_2$ works reasonably well in case of $D = 10$, where its median MSE achieves a value nearly as low as the median MSE of $\mathcal{M}_{1,K=2}$. However, in case of a higher dimension ($D = 20$), the performance of $\mathcal{M}_2$ looks worse. Furthermore, $\mathcal{M}_2$ also shows a much wider quantile range, which can be linked to the extreme complexity of particle filters in high dimensions.

Besides similarity, it is also important to examine other properties of the estimated weight matrices, such as the scaling of the matrix, and see how they evolve over time. Figure 3 displays the average value of the Frobenius norm and the maximum row sum ρ_t of the weight matrices. The curves show that, in all cases, the $\mathcal{M}_{1,K=1}$ estimates (the green line) deviate the most from the generated data (the grey line); in contrast, the estimation from the representations involving the higher order of weight matrix, i.e. $\mathcal{M}_{1,K=2}$ and $\mathcal{M}_2$, are much closer to the true values. Thereby, this evidence suggests another advantage of including the second order of W_t for $\mathcal{M}_1$, which is that it gives better control over the scale of the estimated matrices.

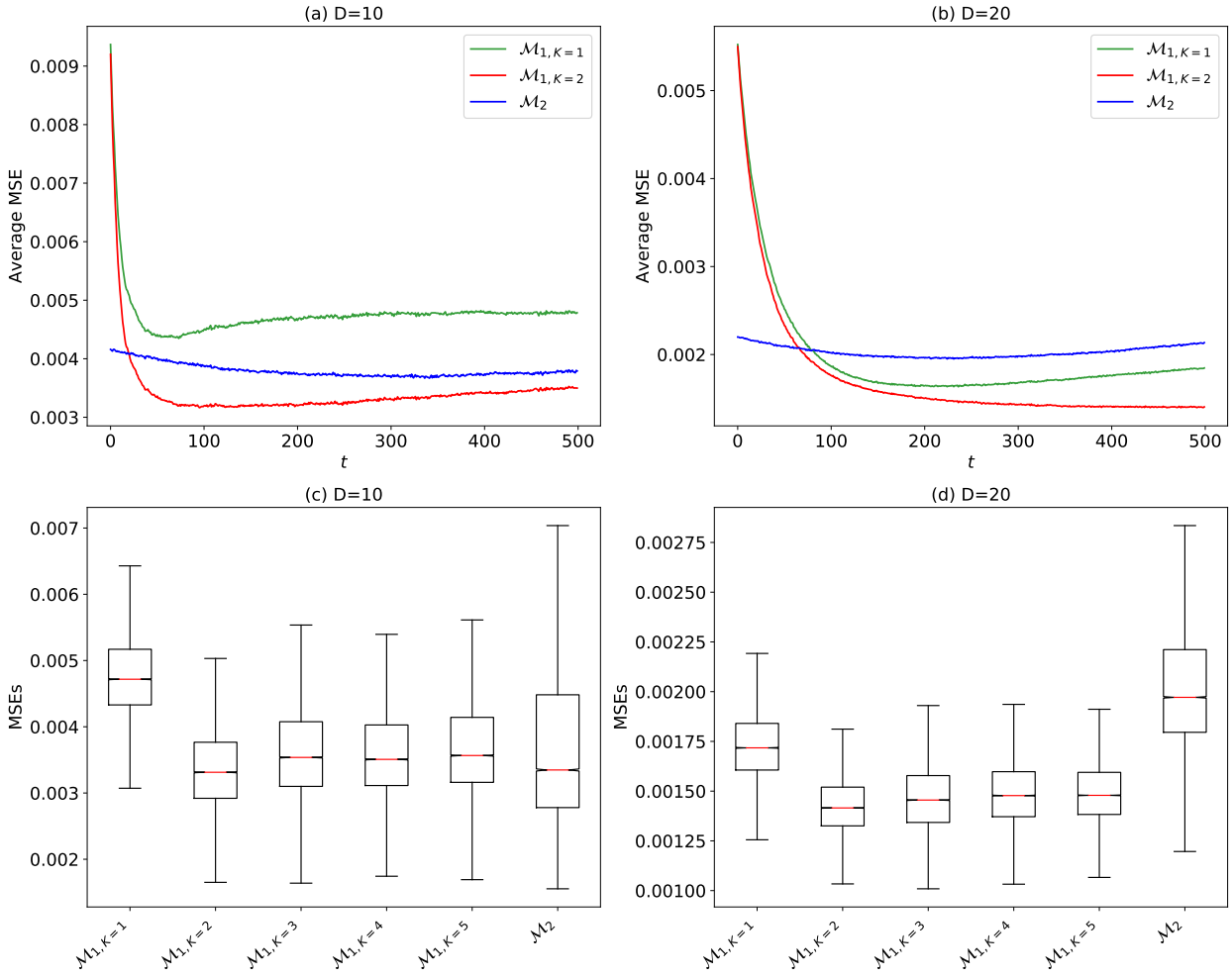

Figure 2: (a) and (b): the average of MSEs ($B = 100$ simulations) at each step. (c) and (d): Boxplots of MSEs. Each box summarises the quantiles of 30000 MSEs (100 simulations and 300 MSEs in each simulation). The measures in the warm-up period (i.e. 200 steps) are omitted.

3.2 Scenario 2: The stochastic W_t

In Scenario 2, the initial weight matrix W_0 is also generated by (8), but, in contrast to Scenario 1, W_t then evolves stochastically. The data generating process is described in Algorithm 2, where $T = 500$, $\sigma_\epsilon^2 = 1$ and $\sigma_\eta^2 = 1 \times 10^{-6}$. Note that, to guarantee that W_t evolves in a meaningful range, we set some constraints: at the beginning of each step, W_{t-1} is maximum-row normalised, and the coefficient 0.9 is applied for generating x_t. Thus, the applied ρ_t, which is the maximum row sum of W_t, is always around 0.9, and the resulting

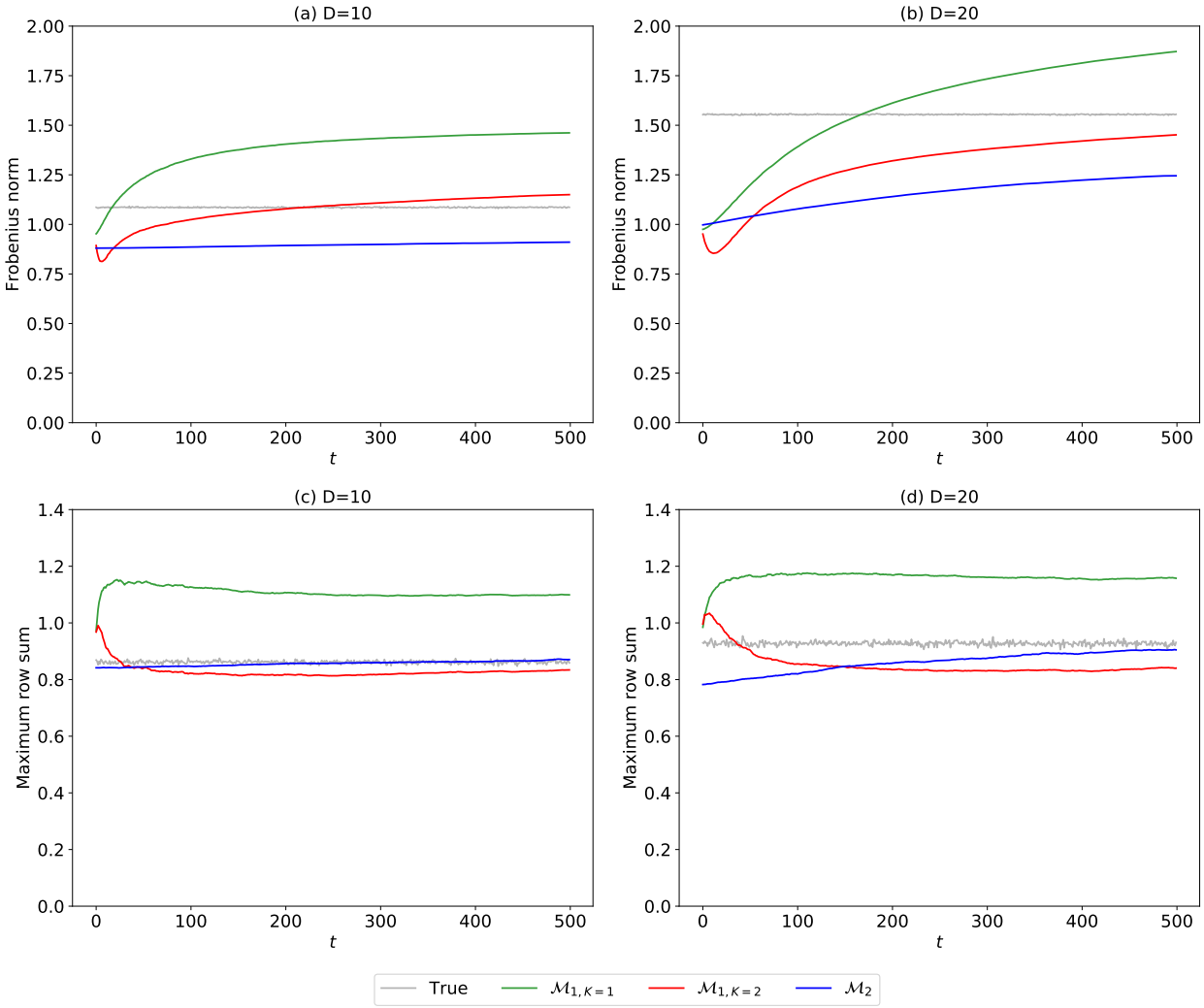

Figure 3: (a) and (b), the average Frobenius norm; (c) and (d), the average ρ_t.

$\tilde{W}_t$ matrices are persistent, coherently with the empirical evidence based on the data used in Section 5, and not leading to singularities in $I - \rho_t \tilde{W}_t$.

Algorithm 2 : The data-generating process for Scenario 2

Generate initial weight matrix W_0 with (8) and (9).
for $t = 1, ..., T$ **do**
$\tilde{W}_{t-1} = W_{t-1}/\rho_{t-1}$
$x_t = 0.9 * \tilde{x}_{t-1} + \eta_t, \eta_t \sim \mathcal{N}(0, \sigma_\eta^2 I_{D_x})$
$y_t = (I - W_t)^{-1}\epsilon_t, \epsilon_t \sim \mathcal{N}(0, \sigma_\epsilon^2 I_D)$
end for
Output: $\{x_t, y_t\}, t = 1, ..., T$

To evaluate the performance of $\mathcal{M}_1$ and $\mathcal{M}_2$, we follow the same procedure as that used in Scenario 1. As the results are consistent with the evidence shown in the previous section, we report the results in B for interested readers, while, here, we evaluate whether the filters perform differently in the two scenarios.

In Table 1, we summarise the statistics of MSEs for both scenarios. By comparing them, we could see that the average MSEs in Scenario 2 are slightly higher than those in Scenario 1. This is reasonable, as the hidden state W_t in Scenario 2 keeps evolving, which means that the filter has to catch up with the varying hidden state constantly, whereas, in Scenario 1, the filter's effort is concentrated in the warm-up period. Furthermore, we also report the average CPU times (in seconds) in Table 1, which show that the running time grows with the state dimension. In particular, when the state dimension increases by a factor of four, from 90 to 380, the CPU time consumed by $\mathcal{M}_{1,K=2}$ and $\mathcal{M}_2$ grows by a factor of three.

		Scenario 1			Scenario 2		
	model	avg	std	CPU time(s)	avg	std	CPU time(s)
$D = 10$	$\mathcal{M}_{1,K=1}$	0.00477	0.00055	39	0.00497	0.00069	35
	$\mathcal{M}_{1,K=2}$	**0.00336**	0.00055	77	**0.00371**	0.00069	78
	$\mathcal{M}_{1,K=3}$	0.00360	0.00061	93	0.00397	0.00072	92
	$\mathcal{M}_{1,K=4}$	0.00358	0.00058	119	0.00398	0.00071	112
	$\mathcal{M}_{1,K=5}$	0.00365	0.00061	152	0.00405	0.00075	142
	$\mathcal{M}_2$	0.00373	0.00125	499	0.00425	0.00154	473
$D = 20$	$\mathcal{M}_{1,K=1}$	0.00182	0.00015	169	0.00184	0.00021	176
	$\mathcal{M}_{1,K=2}$	**0.00143**	0.00014	199	**0.00150**	0.00023	224
	$\mathcal{M}_{1,K=3}$	0.00146	0.00015	242	0.00153	0.00025	235
	$\mathcal{M}_{1,K=4}$	0.00149	0.00015	274	0.00154	0.00025	288
	$\mathcal{M}_{1,K=5}$	0.00149	0.00015	315	0.00156	0.00027	325
	$\mathcal{M}_2$	0.00202	0.00029	1562	0.00208	0.00038	1665

Table 1: The mean and the standard deviation of the MSEs ($B = 100$ simulations); Columns 5 and 8 reports the average computing time on an Intel Core i5-7200U Processor.

4 Sensitivity analysis

The performance of the filters employed to recover the latent W_t might be influenced by the various settings that we adopt in the simulation exercise. To check robustness of the proposed methods as well to provide some guidelines on how to specify these settings in real-world analyses, we provide a set of additional results that focus on the sensitivity of the filter performances to the different parameter values.

4.1 Parameter settings for $\mathcal{M}_1$

In the previous section, we assumed $R = \sigma_\epsilon^2 I_D$ for both simulations designs. In fact, we assume that W_t, together with its higher orders, represents the only source of contemporaneous interdependence, i.e. the spatial effects driven by the weight matrix in the SAR process (1). We now expand the representation (5) by assuming $\epsilon_t \sim \mathcal{N}(0, R)$, i.e.

$$\begin{cases} y_t = \sum_{k=1}^{K} W_t^k y_t + \epsilon_t, & \epsilon_t \sim \mathcal{N}(0, R) \\ x_t = x_{t-1} + \eta_t, & \eta_t \sim \mathcal{N}(0, \sigma_\eta^2 I_{D_x}) \end{cases}, \tag{10}$$

where the off-diagonal entries of R can be non-zero. The aim of this simulation is to examine whether the performance of $\mathcal{M}_1$ is impacted by introducing correlated observation innovations, i.e. to verify if W_t still captures the spatial dependence or is distorted by the presence of correlation; in the latter case, we expect that W_t would also proxy for R. We mainly focus on the representations $\mathcal{M}_{1,K=1}$ and $\mathcal{M}_{1,K=2}$.

The data-generating process is reported in Algorithm 3. Different from Algorithm 1, Algorithm 3 generates ϵ_t with covariance matrix R, which is assumed to have unit diagonal entries and equal off-diagonal elements, i.e., $\rho_{i,j} = \tilde{c}$, $i \neq j$. We consider two settings for $\tilde{c}$: $\tilde{c} = 0$ and $\tilde{c} = 0.05$. When $\tilde{c} = 0$, the data is essentially generated from a pure SAR model. When $\tilde{c} = 0.05$, the disturbances are mildly correlated. The latter case is more challenging,

as the contemporaneous correlation is jointly determined by both W and R. Furthermore, we assume that the underlying W is populated with a constant value 0.05 outside the main diagonal, i.e.

$$\begin{cases} W_{ij} = 0.05, & i \neq j \\ W_{ij} = 0, & i = j \end{cases}. \tag{11}$$

Finally, for completeness, we set $\sigma_\eta = 0.01$, and $T = 5000$. We note that the simulation settings are particularly challenging when $\tilde{c} > 0$ as W_t and R do have exactly the same structure.

Algorithm 3 : The data-generating process for sensitivity analysis

Generate initial weight matrix W_0 with (11).
for $t = 1, ..., T$ **do**
  $x_t = x_0 + \eta_t, \eta_t \sim \mathcal{N}(0, \sigma_\eta^2 I_{D_x})$
  $y_t = (I - W_t)^{-1}\epsilon_t, \epsilon_t \sim \mathcal{N}(0, R)$
end for
Output: $\{x_t, y_t\}, t = 1, ..., T$

4.1.1 The observation and state variance

To obtain a better understanding of the behaviour of the state-space representations, we examine their performance across the different parameter values. The full parameter set for the model is $\theta = \{\mu_0, \sigma_0\, \sigma_\eta, c\}$; μ_0 and σ_0 are the starting values for the state mean and standard deviation, respectively. We assume that $\mu_0 = 0.2$, which is far from the true state value ($W_{i,j} = 0.05$). In addition, the different combinations of σ_0 and σ_η are reported in the first column of Table 2. As for the observation covariance matrix, we assume the structure of R is known, i.e. with the diagonal entries being equal to 1 and the off-diagonal entries being equal to c. Further, c varies from 0 to 0.3. All the results are based on $B = 100$ simulations.

In each simulation, the data is generated according to Algorithm 3, and the two filters, $\mathcal{M}_{1,K=1}$ and $\mathcal{M}_{1,K=2}$, are employed to estimate the latent weight matrix W_t. The mean and

	$M_{1,K=1}$						
	$c = 0$	$c = 0.05$	$c = 0.1$	$c = 0.15$	$c = 0.2$	$c = 0.25$	$c = 0.3$
$\sigma_0 = 0.1, \sigma_\eta = 0.001$	0.00138 (0.00012)	0.00075 (0.00010)	**0.00066** (0.00009)	0.00127 (0.00014)	0.00272 (0.00026)	0.00525 (0.00042)	0.00911 (0.00066)
$\sigma_0 = 0.1, \sigma_\eta = 0.0005$	0.00119 (0.00007)	0.00054 (0.00006)	**0.00043** (0.00005)	0.00097 (0.00010)	0.00231 (0.00019)	0.00468 (0.00030)	0.00829 (0.00048)
$\sigma_0 = 0.01, \sigma_\eta = 0.001$	0.00139 (0.00012)	0.00076 (0.00010)	**0.00065** (0.00008)	0.00116 (0.00013)	0.00242 (0.00023)	0.00459 (0.00037)	0.00788 (0.00060)
$\sigma_0 = 0.01, \sigma_\eta = 0.0005$	0.00143 (0.00007)	0.00084 (0.00006)	0.00052 (0.00005)	**0.00046** (0.00005)	0.00069 (0.00008)	0.00128 (0.00012)	0.00230 (0.00021)
	$M_{1,K=2}$						
$\sigma_0 = 0.1, \sigma_\eta = 0.001$	**0.00065** (0.00010)	**0.00065** (0.00009)	0.00088 (0.00009)	0.00147 (0.00013)	0.00274 (0.00025)	0.00549 (0.00048)	0.01067 (0.00067)
$\sigma_0 = 0.1, \sigma_\eta = 0.0005$	**0.00045** (0.00006)	**0.00045** (0.00006)	0.00066 (0.00006)	0.00121 (0.00010)	0.00233 (0.00018)	0.00445 (0.00029)	0.00795 (0.00049)
$\sigma_0 = 0.01, \sigma_\eta = 0.001$	**0.00064** (0.00010)	0.00065 (0.00009)	0.00087 (0.00009)	0.00146 (0.00013)	0.00267 (0.00024)	0.00504 (0.00042)	0.00895 (0.00059)
$\sigma_0 = 0.01, \sigma_\eta = 0.0005$	**0.00042** (0.00005)	0.00043 (0.00006)	0.00061 (0.00006)	0.000103 (0.00009)	0.000175 (0.00013)	0.000284 (0.00017)	0.000433 (0.00023)

Table 2: ϵ_t is generated independently, i.e $\tilde{c} = 0$. The mean and the standard deviation (in brackets) of MSEs ($B = 100$ simulations) are presented. The statistics are based on the last 1000 steps. For each combination of σ_0 and σ_η, the filter with the best $\hat{c}$ is the one that achieves the lowest MSE and is marked in bold. In the case of two equal lowest average MSEs (up to 10^{-5}), both have been marked in bold.

the standard deviation of the MSEs for the last 1000 steps for each setting are reported in Table 2 for the case where $\tilde{c} = 0$. We remind that the MSEs monitors the distance between the estimated W_t sequence and the corresponding true sequence.

As discussed before, when $\tilde{c} = 0$, the data is generated from a spatial model. We see from Table 2 that $\mathcal{M}_{1,K=1}$ performs best, in terms of MSE, when $c = 0.1/0.15$, while $\mathcal{M}_{1,K=2}$ achieves the lowest MSE when $c = 0/0.05$. Further, both representations achieve similar MSEs at the optimal $\hat{c}$. The evidence shows that $\mathcal{M}_{1,K=1}$ is able to capture the spatial effects, but it has to leverage both W_t and R, with some mild distortions (expectations were in favor of the case $c = 0$ given the absence of correlation among residuals). Meanwhile, $\mathcal{M}_{1,K=2}$, with $c = 0$, which is equivalent to its original specification (5), can capture all the spatial effects generated from a pure SAR process.

In the case of $\tilde{c} = 0.05$, the MSEs for the last 1000 steps are reported in Table 3. We see that, compared to the case where $\tilde{c} = 0$, the optimal $\hat{c}$, in the case where $\tilde{c} = 0.05$, increases for both $\mathcal{M}_{1,K=1}$ and $\mathcal{M}_{1,K=2}$. In particular, the optimal $\hat{c}$ for all variance settings shifts approximately by 0.05. For example, given the first variance setting, the optimal $\hat{c}$ shifts from 0.1 to 0.15 for $\mathcal{M}_{1,K=1}$ and shifts from 0/0.05 to 0.1 for $\mathcal{M}_{1,K=2}$. Notice that, both of them are able to deliver the same level of accuracy as they do in the case where $\tilde{c} = 0$, which indicates that, by adopting the specification in (10), the correlations in the innovations introduced in the data-generating process can be well captured, but they will be mixed-up with the true dynamic in the W_t matrices. Summing up, the results from Tables 2 and 3 suggest that, when the correlations in the innovations are introduced in the data-generating process, they will be captured together with the spatial effect, thus leading to distortions associated with the impossibility of properly disentangling the presence of dynamic spatial dependence and correlations among the observation equation innovations.

Next, we examine how the filter performs with different state variance levels. For the case where $\tilde{c} = 0$, we plot the average MSE at each step in Figure 4. Panels (a) and (b)

	$c=0$	$c=0.05$	$c=0.1$	$c=0.15$	$c=0.2$	$c=0.25$	$c=0.3$
				$M_{1,K=1}$			
$\sigma_0=0.1, \sigma_\eta=0.001$	0.00213 (0.00011)	0.00140 (0.00011)	0.00088 (0.00010)	**0.00065** (0.00009)	0.00080 (0.00011)	0.00142 (0.00016)	0.00239 (0.00028)
$\sigma_0=0.1, \sigma_\eta=0.0005$	0.00194 (0.00007)	0.00120 (0.00008)	0.00068 (0.00007)	**0.00043** (0.00006)	0.00054 (0.00007)	0.00111 (0.00012)	0.00230 (0.00022)
$\sigma_0=0.01, \sigma_\eta=0.001$	0.00213 (0.00011)	0.00141 (0.00011)	0.00089 (0.00010)	**0.00066** (0.00008)	0.00076 (0.00010)	0.00131 (0.00015)	0.00244 (0.00026)
$\sigma_0=0.01, \sigma_\eta=0.0005$	0.00205 (0.00007)	0.00143 (0.00007)	0.00097 (0.00007)	0.00064 (0.00006)	**0.00048** (0.00005)	0.00050 (0.00006)	0.00078 (0.00010)
				$M_{1,K=2}$			
$\sigma_0=0.1, \sigma_\eta=0.001$	0.00074 (0.00010)	0.00066 (0.00010)	**0.00065** (0.00009)	0.00075 (0.00009)	0.00103 (0.00012)	0.00160 (0.00016)	0.00271 (0.00027)
$\sigma_0=0.1, \sigma_\eta=0.0005$	0.00054 (0.00007)	0.00045 (0.00007)	**0.00044** (0.00006)	0.00053 (0.00006)	0.00079 (0.00008)	0.00133 (0.00012)	0.00232 (0.00020)
$\sigma_0=0.01, \sigma_\eta=0.001$	0.00072 (0.00010)	0.00065 (0.00009)	**0.00064** (0.00009)	0.00074 (0.00009)	0.00102 (0.00011)	0.00158 (0.00016)	0.00265 (0.00026)
$\sigma_0=0.01, \sigma_\eta=0.0005$	0.00052 (0.00005)	0.00044 (0.00005)	**0.00042** (0.00005)	0.00051 (0.00005)	0.00072 (0.00007)	0.00113 (0.00010)	0.00179 (0.00014)

Table 3: ϵ_t is generated with correlation $\tilde{c}=0.05$. The mean and the standard deviation (in brackets) of MSEs ($B=100$ simulations) are presented. The statistics are based on the last 1000 steps. For each combination of σ_0 and σ_η, the filter with the optimal $\hat{c}$ achieves the lowest MSE and is marked in bold.

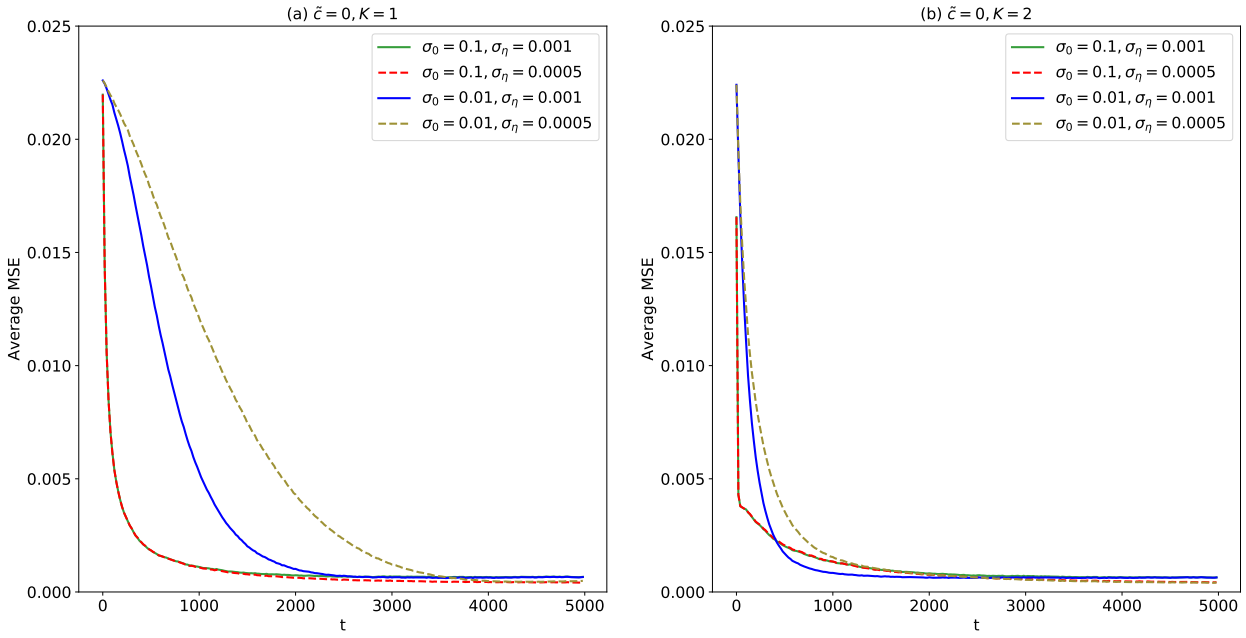

Figure 4: The average of the MSEs ($B = 100$ simulations) at each step. Two parameters were examined: the initial state variance σ_0^2 and the state variance σ_η^2.

show the results for $K = 1$ and $K = 2$, respectively. The four curves correspond to different combinations of σ_0 and σ_η; for each combination, an optimal $\hat{c}$ is chosen. In all settings, the initial state mean μ_0 is far from the true value. Note that, a large μ_0 is only used to demonstrate the convergence of the filter. We could see that, although the initial guess is far from being reasonable, the filter is still able to converge. LEAVE OUT: σ_0^2, the variance of the initial condition x_0 of x, is a measure of how we are confident about the initial value x_0 itself. Here μ_0, the mean of x_0, is set to 0.2 which, as previously mentioned, is a large value compared to the true state value $W_{ij} = 0.05$ used for simulations. This might explains why in this particular simulation study the algorithm converges faster if we increase the uncertainty σ_0^2 in having proposed $\mu_0 = 0.2$ as the average value of x_0. Notice also that the convergence slows down as the initial state variance σ_0^2 decreases. For example, in Figure 4 (a), when $\sigma_0 = 0.1$, the curve stabilises within 2000 steps. However, as we reduce the variance to $\sigma_0 = 0.01$, the converging speed slows down significantly, and it takes more than

2000 steps to arrive at a stable value. σ_η seems to have similar effects, and the results are more obvious in the case where $\sigma_0 = 0.01$: in both Panels (a) and (b) in Figure 4, the yellow curve ($\sigma_\eta = 0.0005$) moves downward at a slightly slower pace than the blue curve ($\sigma_\eta = 0.001$). Moreover, the advantage of a small σ_η is also significant, as, from Tables 2 and 3, we see that, after convergence, the filter with the smaller σ_η leads to a much lower error. For example, in Table 2, given that $\sigma_0 = 0.1$, when σ_η is reduced from 0.001 to 0.0005, the average MSE of $\mathcal{M}_{1,K=2}$ is reduced from 0.00065 to 0.00045. We may conclude that, if the state variance decreases, the converging process slows down, but the estimation error also declines. In practice, when choosing a proper σ_η, one may need to evaluate the balance between the converging speed and the expected error level.

4.1.2 The ensemble size

The ensemble size N plays a crucial role from a computational point of view, but it could also have an impact on accuracy. Its appropriate selection is thus of paramount importance. There are several rules that may help make this decision. First, the SEKF can only converge when the ensemble size N is larger than the rank of the observation covariance matrix, and, in our case, $N > D$ (Anderson, 2009). Second, as discussed in Gillijns et al. (2006), usually 50 to 100 samples can be enough for a state dimension up to thousands, and the error can be reduced as N increases. Third, the necessary ensemble size has a linear relationship with the state dimension (Katzfuss et al., 2020).

To obtain a concrete idea about the necessary amount of samples, we measure the performance of $\mathcal{M}_{1,K=1}$ and $\mathcal{M}_{1,K=2}$ by varying N from 100 to 1000. The data-generating process follows Algorithm (3), where $\tilde{c} = 0$. The parameters for the state-space representation are fixed with $\mu_0 = 0.1$, $\sigma_0 = 0.1$ and $\sigma_\eta = 0.001$. The optimal $\hat{c}$ is chosen, i.e. $c = 0.1$ for $\mathcal{M}_{1,K=1}$, and $c = 0$ for $\mathcal{M}_{1,K=2}$. We run $B = 100$ simulations. Figure 5 (a) reports the average MSEs with different ensemble sizes. The curves show that both $\mathcal{M}_{1,K=1}$ and $\mathcal{M}_{1,K=2}$

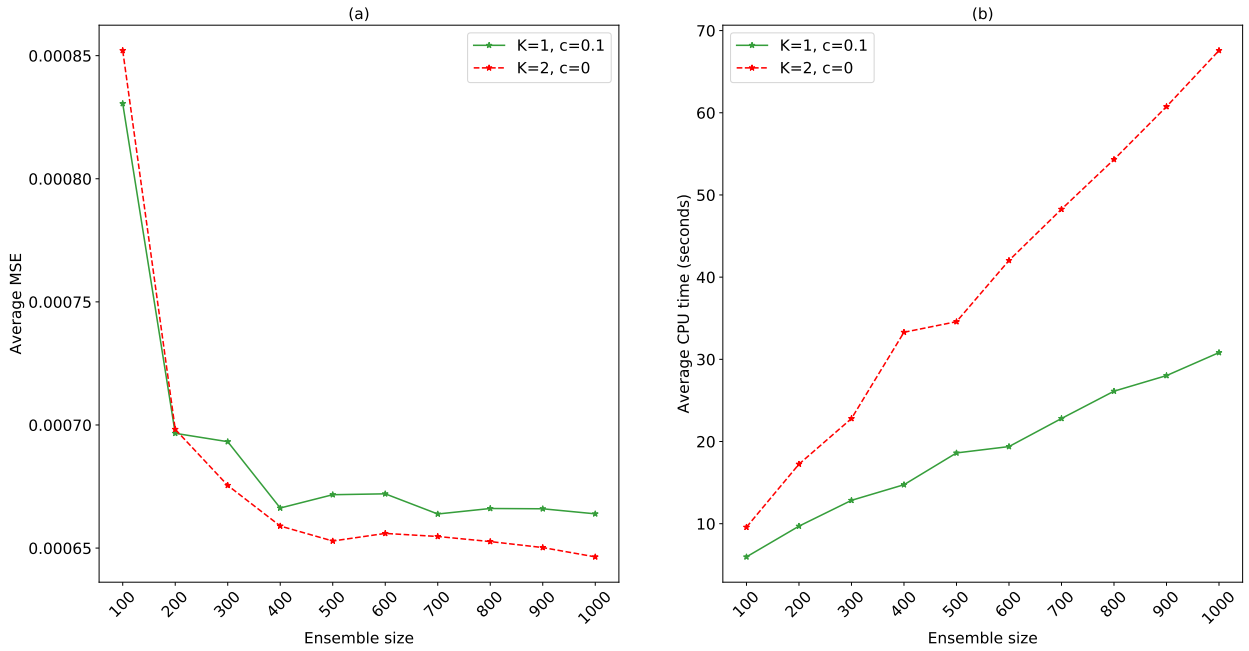

Figure 5: (a) The average of MSEs ($B = 100$ simulations) with increasing ensemble size N. (b) The average computing time refers to an Intel Core i5-7200U Processor.

need 400-500 samples to achieve reasonable accuracy and the level of errors could hardly be further reduced as N continues increasing. In our case, the ensemble size is larger than the values of 50-100 reported in Gillijns et al. (2006). A possible reason for this might be the large dimensional gap between the state and the observation; given the same state dimension, it is more challenging for the filter to retrieve the information if the observation is in a much lower dimensional space. Finally, we also report the average CPU times in Figure 5 (b), which shows a linear relationship between the running time and the ensemble size.

4.2 Parameter settings for $\mathcal{M}_2$

We follow the same procedure to study the parameters in $\mathcal{M}_2$. In particular, Algorithm 3 is employed for generating the data when the correlations among the innovations are considered, i.e. $\epsilon_t \sim \mathcal{N}(0, R)$. To capture the possible contemporaneous correlation, we also

adopt a relaxed specification for $\mathcal{M}_2$, i.e.

$$\begin{cases} y_t = (I_D - W_t)^{-1}\epsilon_t, & \epsilon_t \sim \mathcal{N}(0, R) \\ x_t = x_{t-1} + \eta_t, & \eta_t \sim \mathcal{N}(0, \sigma_\eta^2 I_{D_x}) \end{cases}, \tag{12}$$

where R is assumed to have unit diagonal entries and an equal off-diagonal value c. Accordingly, the log-likelihood function changes from (7) to

$$l(W_t|y_t) = -\frac{D}{2}\log(2\pi) - \frac{1}{2}\log|R| + \log|A_t| - \frac{1}{2}(A_t y_t)^T R^{-1}(A_t y_t), \tag{13}$$

where $A_t = I_D - W_t$. It is easy to see that the correlated innovations only impact the last part of the log-likelihood function, i.e. $-\frac{1}{2}(A_t y_t)^T R^{-1}(A_t y_t)$.

In the following subsections, we first investigate two parameters, σ_η and c, for specification (12). The initial settings μ_0 and σ_0 are not examined here, as it is the proposal parameter $\alpha_{t=0}$ that traces the initial state. They are set with fixed values: $\mu_0 = 0.1$ and $\sigma_0 = 0.1$. Further, we examine two hyper-parameters, the number of particles N and the time span T. As discussed before, VSMC typically optimises the model on the whole dataset. The performance of $\mathcal{M}_2$ may be compromised if T is too small or too large. Thus, T is also included in the sensitivity analysis for $\mathcal{M}_2$. With respect to the optimisation process for VSMC, two parameters can be relevant: the number of optimisation steps M and the step size s_m. Considering these are usual parameters in many optimization algorithms, here we only report the values adopted: α_m is optimised with 200 steps, i.e. $M = 200$; for the first 100 steps, the step size is fixed to 0.001; for the second 100 steps, the step size is fixed to 0.0001.

4.2.1 The observation and state variance

The data is generated using Algorithm 3 with two scenarios: $\tilde{c} = 0$ and $\tilde{c} = 0.05$. All the parameters used in the data-generating process are the same as their $\mathcal{M}_1$ counterparts except for a shorter time span, which is $T = 500$. For the state-space representation, we let c vary from 0 to 0.2 and σ_η vary from 0.001 and 0.0005.

We run $B = 100$ simulations. The averages and standard deviations of the MSEs estimated with different combinations of the parameter settings are summarised in Table 4. The lowest average MSE (in bold) in each row corresponds to the optimal $\hat{c}$. Table 4 shows that, when the data is generated from a pure SAR process, i.e. $\tilde{c} = 0$, the optimal $\hat{c}$ is 0 or 0.05. Further, when the extra correlations are introduced for the disturbance terms, i.e. $\tilde{c} = 0.05$, the optimal $\hat{c}$ also increases accordingly. However, with the optimal $\hat{c}$, $\mathcal{M}_2$ achieves similar accuracy in both cases. The evidence implies that $\mathcal{M}_2$, in its relaxed form (12), is capable of capturing both the spatial effects driven by the weight matrix and the correlations in the innovations, but again they are going to be mixed-up, due to identification issue.

In terms of the state variance, Table 4 shows that, in both cases, as σ_η decreases, the average MSE also decreases. This has also been observed for $\mathcal{M}_1$ (Tables 2 and 3). In general, smaller σ_η provides higher precision for the filter, but it also limits the converging speed. Thus, to set a proper σ_η in practice, one also needs to consider the possible changing speed of the true state.

4.2.2 The number of particles N and the time span T

In this section, we study the role of the two hyper-parameters N and T. In VSMC, the particle filter is applied in each optimisation step, so a proper setting for N can benefit both the computational cost and the performance in general. On the other hand, if T is too large, the performance of $\mathcal{M}_2$ may also be deteriorated, as a larger T implies a larger parameter space for VSMC. To obtain a better understanding on how the two parameters might impact

Mean Squared Errors, $\tilde{c} = 0$					
	$c = 0$	$c = 0.05$	$c = 0.1$	$c = 0.15$	$c = 0.2$
$\sigma_\eta = 0.001$	**0.00064** (0.00008)	**0.00064** (0.00008)	0.00072 (0.00013)	0.00079 (0.00019)	0.00090 (0.00027)
$\sigma_\eta = 0.0005$	**0.00045** (0.00008)	**0.00045** (0.00008)	0.00053 (0.00011)	0.00059 (0.00017)	0.00066 (0.00022)
Mean Squared Errors, $\tilde{c} = 0.05$					
	$c = 0$	$c = 0.05$	$c = 0.1$	$c = 0.15$	$c = 0.2$
$\sigma_\eta = 0.001$	0.00078 (0.00010)	0.00065 (0.00008)	**0.00063** (0.00008)	0.00067 (0.00010)	0.00073 (0.00014)
$\sigma_\eta = 0.0005$	0.00063 (0.00011)	0.00047 (0.00010)	**0.00045** (0.00008)	0.00048 (0.00012)	0.00052 (0.00013)

Table 4: The mean and the standard deviation (in brackets) of MSEs ($B = 100$ simulations). For each setting for σ_η, the filter with the optimal $\hat{c}$ achieves the lowest MSE (in bold). In the case of two equal lowest average MSEs (up to 10^{-5}), both have been marked in bold.

the empirical results, different settings are tested, i.e. $N = \{10, 100\}$ and $T = \{100, 500, 1000\}$.

Mean Squared Errors			
	$T = 100$	$T = 500$	$T = 1000$
$N = 10$	0.00073 (0.00018)	**0.00066** (0.00010)	0.00086 (0.00011)
$N = 100$	0.00075 (0.00016)	**0.00065** (0.00009)	0.00089 (0.00011)
CPU Time (seconds)			
	$T = 100$	$T = 500$	$T = 1000$
$N = 10$	101	841	2559
$N = 100$	162	1160	3170

Table 5: The upper part of the table reports the mean and the standard deviation (in brackets) of MSEs ($B = 100$ simulations). The filter achieves the lowest MSE (in bold) when $T = 500$. The lower part of the table reports the average computing time on a 4-core Intel Xeon processor.

We run $B = 100$ simulations. The data is generated using Algorithm 3, where $\tilde{c} = 0$.

The parameters for $\mathcal{M}_2$ are fixed, i.e., $\mu_0 = 0.1$, $\sigma_0 = 0.1$, $\sigma_\eta = 0.001$ and $c = 0$. The means and standard deviations of MSEs are summarised in Table 5. We can see that the filter performs similarly in cases where $N = 10$ and $N = 100$. In fact, as discussed in Le et al. (2017), a larger N can even be harmful for proposal learning, as a larger number of particles can reduce the magnitude of the gradient. In addition, a small N should be preferred from a computational-cost perspective (Naesseth et al., 2018). As for the time span, Table 5 shows that $\mathcal{M}_2$ performs the best when $T = 500$. When $T = 100$, despite the small number of observations, the performance of $\mathcal{M}_2$ is quite reasonable. It is worth noticing that, in this case, T is only slightly larger than the state dimension $D(D-1) = 90$. When estimating an invariant weight matrix in a SAR model, $T > D(D-1)$ is required for providing enough degrees of freedom. In our case, for the state-space representation, $T > D(D-1)$ also provides a good starting point for determining a minimum T. When $T = 1000$, the increasing number of parameters starts to deteriorate the performance of $\mathcal{M}_2$. Naesseth et al. (2018) suggested that the accuracy of VSMC can be kept the same by setting $N \propto T$. In practice, if the full dataset is longer than the optimal T, an efficient way of estimation would be to split the dataset in such a way that each set of data is of optimal length T. Further, the need of sufficiently long time series might represent a limit for the application of our methodology to fields were data availability is not a great concern as, for instance, in the case of data recovered from financial markets.

The average computing times for this simulation are reported in the lower part of Table 5. We can see that the CPU time has an approximately linear relationship with the time span T. However, as N increases exponentially, the elapsed time only increases slightly. This is reasonable, as N is only related to the efficiency of the particle filter, rather than the whole optimisation process.

5 Real-world analysis

As the last step, we test $\mathcal{M}_1$ and $\mathcal{M}_2$ on real-world data and examine whether the features of the estimated weight matrix provide some economic information as well as if they align with our expectations. To that end, we apply two models, $\mathcal{M}_{1,K=2}$ and $\mathcal{M}_2$, to a weekly CDS dataset for 16 large banks from four selected European countries. The bank names are reported in C. The time horizon is ten years (i.e., 521 weekly observations), covering the period from 4th Jan 2009 to 30th Dec 2018. The input for the modelling is the logarithm change of CDS levels, which is normalised such that the standard deviations of the observations are all equal to one. The expected output is a series of 16×16 weight matrices, which represents the dynamic spillover effects among banks. As banks from different countries may have exposure to different types of risks, country-driven communities are expected to be identified from the estimated weight matrices (see Torri et al. (2018)).

To take into consideration the presence of limited serial dependence, typical of financial times series, we also add a time lag, an autoregressive component, into the state-space models. Specifically, the lagged observation y_{t-1} is introduced as a factor into the observation equations, and the applied state-space models are as follows:

$$\begin{cases} y_t = \mathbf{A}_t y_{t-1} + (W_t + W_t^2) y_t + \epsilon_t, & \epsilon_t \sim \mathcal{N}(0, \sigma_\epsilon^2 I_D) \\ x_t = x_{t-1} + \eta_t, & \eta_t \sim \mathcal{N}(0, \sigma_\eta^2 I_{D_x}) \end{cases}, \tag{14}$$

$$\begin{cases} y_t = \mathbf{A}_t y_{t-1} + (I - W_t)^{-1} \epsilon_t, & \epsilon_t \sim \mathcal{N}(0, \sigma_\epsilon^2 I_D) \\ x_t = x_{t-1} + \eta_t, & \eta_t \sim \mathcal{N}(0, \sigma_\eta^2 I_{D_x}) \end{cases}, \tag{15}$$

where y_t denotes the CDS observations for 16 countries at week t. $\mathbf{A}_t$ is the 16×16 diagonal matrix of the autoregressive coefficient, i.e. $\mathbf{A}_t = diag(a_1, ..., a_{16})$. The state x_t is composed of the diagonal entries of A_t and the off-diagonal entries of W_t. The variance of the observations and the states are assumed to be known: $\sigma_\epsilon^2 = 1$ and $\sigma_\eta^2 = 1 \times 10^{-6}$. Note

that the representation (15) is just a variant of $\mathcal{M}_2$. The only difference is that, now, the weight matrix measures the spillover effects among the residual $y_t - \mathbf{A}_t y_{t-1}$, i.e. once the serial dependence is filtered out from the data.

Figure 6 shows the estimated weight matrices at certain cut-offs, $t = 100, 200, 300, 400$ and 500. For each matrix, the observable community blocks are outlined using dashed red lines. The first row illustrates the matrices estimated with representation (14). We could see that, when $t = 100$, the community blocks are still at the state of emerging (the warm-up period); but as t increases, the partitions become clearer. As for the estimation using representation (15) (the second row), the communities are well identified across all periods.

Further, we compute the partitions based on each estimated weight matrix using the Louvain algorithm (Blondel et al., 2008). The results are plotted in Figures 7 and 8, where the underlying weight matrices are estimated using (14) and (15), respectively. The graph can be understood as a table with 16 columns, which represent the banks, and 521 rows, which correspond to the 521 weeks. In each row, banks belonging to the same group are displayed by the same color. For example, the Italian banks, Intesa Sanpaolo and Monte Paschi, are coloured green each time, which indicates that they belong to the same group over the entire study period. We see from Figure 7 that the partitions are not quite clear-cut in the first few years, which is in line with the evidence from Figure 6. However, afterwards, banks belonging to the same countries are all classified into their respective groups: green for Italy, black for France and yellow for the UK. Figure 8 is almost coherent with the content of Figure 7. There are, in fact, two exceptions: first, after 2011, the two German banks belong to an independent group in Figure 7, but they merge with Italian banks in Figure 8; second, related to the features of different filters, in contrast to Figure 7, Figure 8 exhibits country-based partitions from the very beginning and maintains consistency afterwards. This could be considered as an advantage for offline methods, especially when the estimates for the early steps are important.

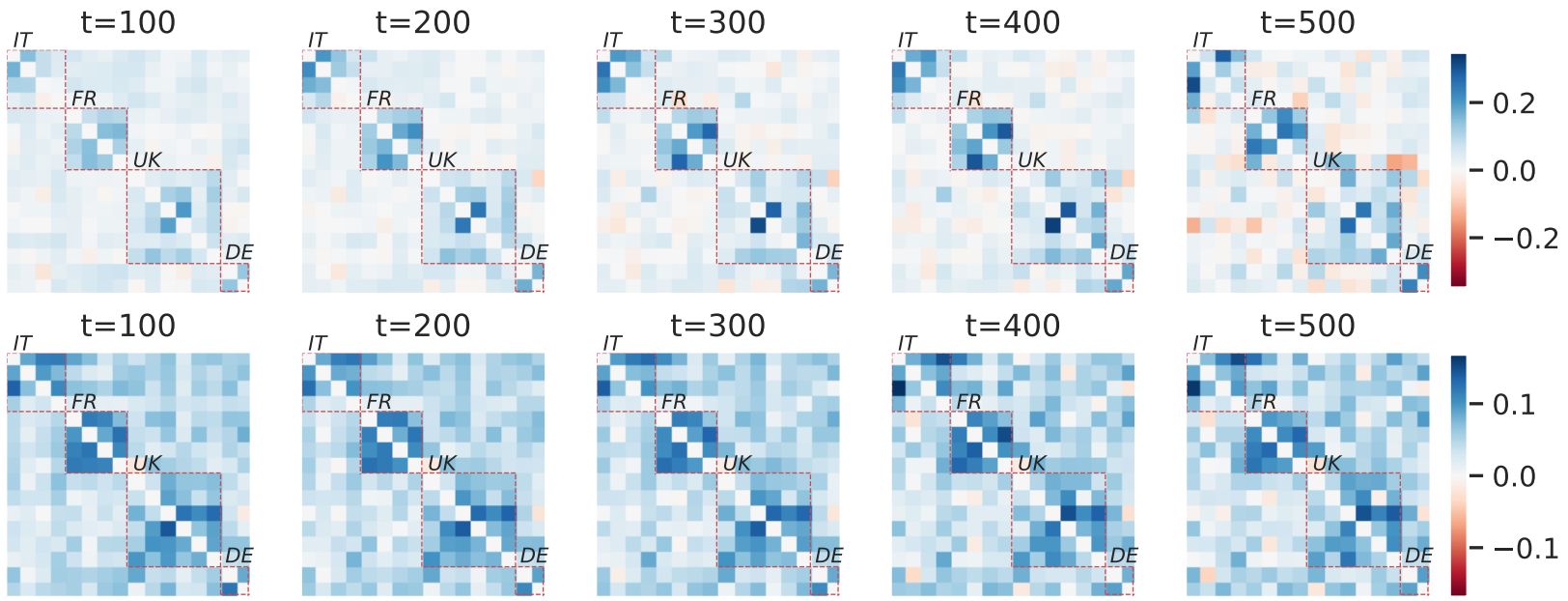

Figure 6: Estimated weight matrix at $t = 100, 200, 300, 400, 500$. The matrices in the first row are estimated with representation (14), and the matrices in the second row are estimated with representation (15). Each community is outlined using dashed red lines and labeled with the corresponding country name abbreviations, i.e. IT for Italy, FR for France, UK for the United Kingdom and DE for Germany.

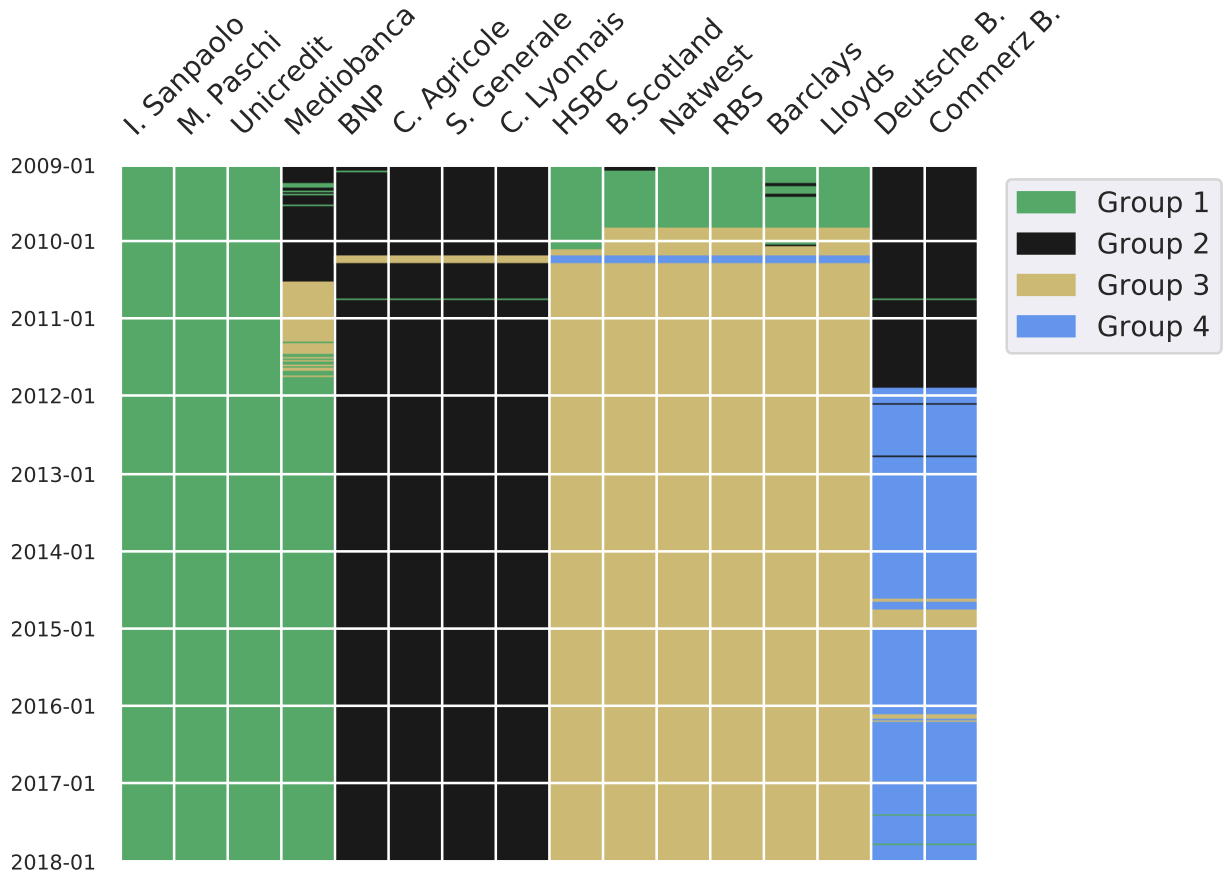

Figure 7: Community detection. The underlying weight matrix is estimated with representation (14). Green for Group 1; black for Group 2; yellow for Group 3; blue for Group 4.

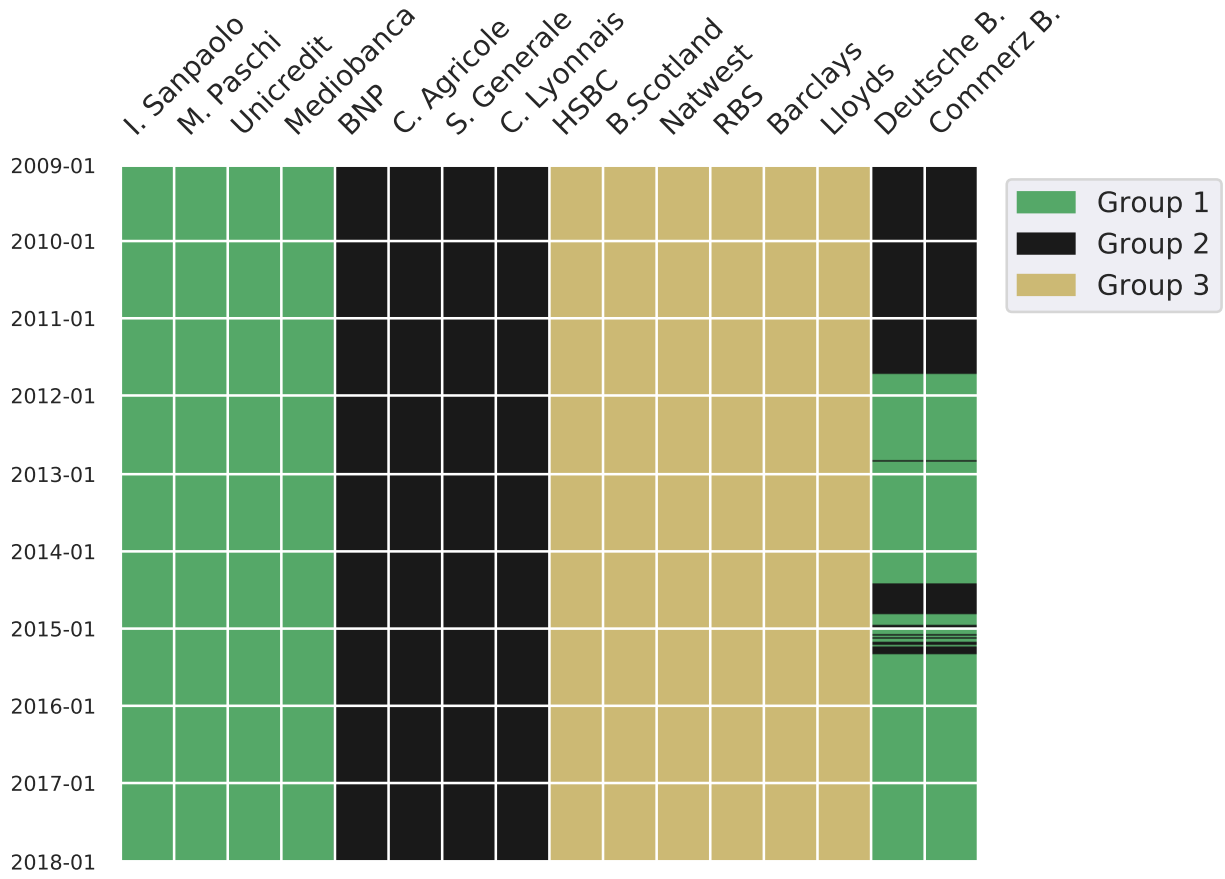

Figure 8: Community detection. The underlying weight matrix is estimated with representation (15). Green for Group 1; black for Group 2; yellow for Group 3.

In the end, we investigate the performance of representations (14) and (15) from a forecasting perspective. A VAR model is used as the benchmark. To perform the dynamic estimation, we adopt a moving-window approach for VAR, where three window sizes are applied: 52 weeks, 104 weeks and 156 weeks. The in-sample errors are measured for each moving window, and the out-of-sample errors are calculated using one-step forecasting. As for the two state-space representations, at each step, the estimated $\mathbf{A}_t$ and W_t are applied for calculating in-sample errors at t and out-of-sample errors at $t+1$. Table 6 summarises

the MSEs for all five models and shows that, in terms of both the in-sample and the out-of-sample measures, the state-space model could outperform the moving-window VAR model. To complement this point value comparison with a statistical test, we report in the last column the p-value of the Model Confidence Set (Hansen et al., 2011). We perform the test using the MSE loss function and R test statistic, with 1000 block-bootstrap replications and an average block length of four observations (we refer the reader to Hansen et al. (2011) for details on the test implementation); the test confirms that the preferred specification is the first state-space representation (small p-values identify the exclusion of the model from the set of preferred specifications). The result is promising, as it suggests that, in some cases, the dynamic weight matrix may also be useful for model fitting and forecasting.

Model	MSE (in sample)	MSE (out of sample)	MCS
$\mathcal{M}_{1,K=2}$ (eq. (14))	0.00270	0.01305	1.000
$\mathcal{M}_2$ (eq. (15))	0.00279	0.01458	0.014
VAR (52 weeks)	0.00614	0.03760	0.014
VAR (104 weeks)	0.00738	0.03557	0.014
VAR (156 weeks)	0.00786	0.02545	0.014

Table 6: The average MSE of model fitting and forecasting (Column 2 and 3) and the Model Confidence Set p-values (Column 4).

6 Conclusion

In this paper, we propose two types of state-space representations for estimating dynamic networks. For $\mathcal{M}_1$, we introduce the higher order of W_t such that both direct and indirect spillover effects can be addressed, while, for $\mathcal{M}_2$, W_t is updated with a conventional spatial

model. The simulation results show that when the indirect spillover effect is taken into account, i.e. $K > 1$, the performance of $\mathcal{M}_1$ improves significantly. Compared to $\mathcal{M}_2$, the advantage of $\mathcal{M}_1$ is that can be identified efficiently and it can better deal with the higher-dimensional cases, as the representation is estimated with SEKF, which requires only a linear relationship between the sample size and the state dimension (Katzfuss et al., 2020). Meanwhile, the evidence shows that the performance of $\mathcal{M}_2$ cannot reach the performances of $\mathcal{M}_{1,K=2}$ as D increases to 20, i.e. the state dimension increases to 380. Essentially, this is because $\mathcal{M}_2$ could only be identified by a particle filter, which is known to suffer from the curse of dimensionality problem. However, $\mathcal{M}_2$ provides an authentic way to identify the weight matrix, as it directly utilises the log-likelihood function of the SAR model. Further, by employing VSMC, we show that W_t can be estimated with a small T, which can be quite useful in practice when the length of the dataset is limited. In the real-world application, we test two representations on the CDS levels for banks from four countries and show the effectiveness of the representations in identifying country-driven partitions.

The method we proposed can be further studied along several directions. First, in this work, we only apply the state-space representations for estimating small networks, i.e. $D = 10$ and $D = 20$, and with relatively long time series. Further research is needed for identifying larger networks, and also for covering cases where the ratio between the temporal dimension and the cross-sectional dimension decreases, a situation that would create relevant computational challenges. Essentially, we want to associate the state-space representations with filtering algorithms that are efficient in high-dimensional space, also controlling for the sample size over time. Second, if any underlying dependency structure is known, W_t can be estimated under certain constraints, such as a symmetric W_t, or estimated based on some known matrices, such as $W_t = \mathcal{G}_t(\mathcal{W}_1, \mathcal{W}_2, ...)$, where $\mathcal{G}_t$ is the function of some known matrices $\mathcal{W}_1$ and $\mathcal{W}_2$ at step t. In either case, the state dimension is expected to be reduced significantly. Third, the method may be further generalised, in the sense that the spatial

model can be replaced by other models where the dependency structure is of interest. One such example is the BEKK model (Engle and Kroner, 1995), where the parameter matrix measures the intertemporal spillover effects among volatilities and co-volatilities. Finally, as our contribution is purely empirical and simulation-based, theoretical properties of the estimators and a deeper understanding of the identification issue could also represent interesting lines of research.

Acknowledgements

An earlier draft of this work (i.e. the state-space model $\mathcal{M}_{1,K=1}$ and the performance of the filters) was presented at the 13th International Conference on Computational and Financial Econometrics (CFE 2019, London). We thank the participants of the conference for valuable feedbacks. The second author acknowledges financial support from Ministero dell'Universitá e della Ricerca project PRIN2017 HiDEA: Advanced Econometrics for High-frequency Data, 2017RSMPZZ.

References

Ahrens, A. and Bhattacharjee, A. (2015). Two-step lasso estimation of the spatial weights matrix. *Econometrics*, 3(1):128–155.

Alter, A. and Beyer, A. (2014). The dynamics of spillover effects during the european sovereign debt turmoil. *Journal of Banking & Finance*, 42:134–153.

Anderson, J. L. (2009). Ensemble kalman filters for large geophysical applications. *IEEE Control Systems Magazine*, 29(3):66–82.

Andrieu, C., Doucet, A., and Holenstein, R. (2010). Particle Markov chain Monte Carlo

methods. *Journal of the Royal Statistical Society. Series B: Statistical Methodology*, 72(3):269–342.

Anselin, L. (2010). Thirty years of spatial econometrics. *Papers in Regional Science*, 89(1):3–25.

Anselin, L. (2013). *Spatial econometrics: methods and models*, volume 4. Springer Science & Business Media.

Bhattacharjee, A. and Jensen-Butler, C. (2013). Estimation of the spatial weights matrix under structural constraints. *Regional Science and Urban Economics*, 43(4):617–634.

Billio, M., Caporin, M., Panzica, R., and Pelizzon, L. (2016). The impact of network connectivity on factor exposures, asset pricing and portfolio diversification.

Billio, M., Getmansky, M., Lo, A. W., and Pelizzon, L. (2012). Econometric measures of connectedness and systemic risk in the finance and insurance sectors. *Journal of Financial Economics*, 104(3):535–559.

Blondel, V. D., Guillaume, J.-L., Lambiotte, R., and Lefebvre, E. (2008). Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2008(10):P10008.

Burgers, G., Jan van Leeuwen, P., and Evensen, G. (1998). Analysis scheme in the ensemble kalman filter. *Monthly Weather Review*, 126(6):1719–1724.

Caporin, M. and Paruolo, P. (2015). Proximity-structured multivariate volatility models. *Econometric Reviews*, 34(5):559–593.

Debreu, G. and Herstein, I. N. (1953). Nonnegative square matrices. *Econometrica: Journal of the Econometric Society*, pages 597–607.

Diebold, F. X. and Yılmaz, K. (2014). On the network topology of variance decompositions: Measuring the connectedness of financial firms. *Journal of Econometrics*, 182(1):119–134.

Elliott, M., Golub, B., and Jackson, M. O. (2014). Financial networks and contagion. *American Economic Review*, 104(10):3115–53.

Engle, R. F. and Kroner, K. F. (1995). Multivariate simultaneous generalized arch. *Econometric Theory*, pages 122–150.

Getis, A. and Aldstadt, J. (2004). Constructing the spatial weights matrix using a local statistic. *Geographical Analysis*, 36(2):90–104.

Gillijns, S., Mendoza, O. B., Chandrasekar, J., De Moor, B., Bernstein, D., and Ridley, A. (2006). What is the ensemble kalman filter and how well does it work? In *2006 American Control Conference*, pages 6–pp. IEEE.

Hansen, P. R., Lunde, A., and Nason, J. M. (2011). The model confidence set. *Econometrica*, 79(2):453–497.

Haslbeck, J. M., Bringmann, L. F., and Waldorp, L. J. (2020). A tutorial on estimating time-varying vector autoregressive models. *Multivariate Behavioral Research*, pages 1–30.

Horn, R. A. and Johnson, C. R. (2012). *Matrix analysis*, page 351. Cambridge university press.

Houtekamer, P. L. and Mitchell, H. L. (1998). Data assimilation using an ensemble kalman filter technique. *Monthly Weather Review*, 126(3):796–811.

Katzfuss, M., Stroud, J. R., and Wikle, C. K. (2020). Ensemble kalman methods for high-dimensional hierarchical dynamic space-time models. *Journal of the American Statistical Association*, 115(530):866–885.

Kelejian, H. H. and Prucha, I. R. (1999). A generalized moments estimator for the autoregressive parameter in a spatial model. *International Economic Review*, 40(2):509–533.

Kelejian, H. H. and Prucha, I. R. (2002). 2sls and ols in a spatial autoregressive model with equal spatial weights. *Regional Science and Urban Economics*, 32(6):691–707.

Khan, Z., Balch, T., and Dellaert, F. (2005). Mcmc-based particle filtering for tracking a variable number of interacting targets. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(11):1805–1819.

Kimura, T., Kobayashi, H., Muranaga, J., Ugai, H., et al. (2003). The effect of the increase in the monetary base on Japans economy at zero interest rates: An empirical analysis. In *Monetary Policy in a Changing Environment, Bank for International Settlements Conference Series*, volume 19, pages 276–312.

Lam, C. and Souza, P. C. (2019). Estimation and selection of spatial weight matrix in a spatial lag model. *Journal of Business & Economic Statistics*, pages 1–41.

Le, T. A., Igl, M., Rainforth, T., Jin, T., and Wood, F. (2017). Auto-encoding sequential monte carlo. *arXiv preprint arXiv:1705.10306*.

Lee, L.-F. (2002). Consistency and efficiency of least squares estimation for mixed regressive, spatial autoregressive models. *Econometric Theory*, 18(2):252–277.

LeSage, J. and Pace, R. (2009). *Introduction to Spatial Econometrics*. Statistics: A Series of Textbooks and Monographs. CRC Press.

LeSage, J. P. (2008). An introduction to spatial econometrics. *Revue D'économie Industrielle*, (123):19–44.

LeSage, J. P. and Fischer, M. M. (2008). Spatial growth regressions: model specification, estimation and interpretation. *Spatial Economic Analysis*, 3(3):275–304.

LeSage, J. P. and Pace, R. K. (2007). A matrix exponential spatial specification. *Journal of Econometrics*, 140(1):190–214.

LeSage, J. P. and Pace, R. K. (2014). The biggest myth in spatial econometrics. *Econometrics*, 2(4):217–249.

Meen, G. (1996). Spatial aggregation, spatial dependence and predictability in the uk housing market. *Housing Studies*, 11(3):345–372.

Naesseth, C. A., Linderman, S. W., Ranganath, R., and Blei, D. M. (2018). Variational sequential monte carlo. *Proceedings of the 21st International Conference on Artificial Intelligence and Statistics 2018*.

Ord, K. (1975). Estimation methods for models of spatial interaction. *Journal of the American Statistical Association*, 70(349):120–126.

Snyder, C., Bengtsson, T., Bickel, P., and Anderson, J. (2008). Obstacles to high-dimensional particle filtering. *Monthly Weather Review*, 136(12):4629–4640.

Torri, G., Giacometti, R., and Paterlini, S. (2018). Robust and sparse banking network estimation. *European Journal of Operational Research*, 270(1):51–65.

A Filter algorithms

We provide pseudo-algorithms for the three filters used in this paper, i.e., KF, SEKF and VSMC. The general form of the state-space model is defined as follows:

$$\begin{cases} y_t = \mathcal{H}(x_t) + \sigma_t, & \sigma_t \sim \mathcal{N}(0, R) \\ x_t = \mathcal{F}(x_{t-1}) + \eta_t, & \eta_t \sim \mathcal{N}(0, Q) \end{cases}.$$

In case of linear SSM, $\mathcal{H}(x_t) = Hx_t$ and $\mathcal{F}(x_{t-1}) = Fx_{t-1}$, where H and F are the observation and transition matrices, respectively.

Notations:

$N-$ the sample/particle size

x_t- the estimated state mean at time t

$x_t^{(i)}-$ the i-th sample state in the analysis step, $i = \{1, ..., N\}$

$\tilde{x}_t-$ the predicted state mean in the forecast step, at time t

$\tilde{x}_t^{(i)}-$ the i-th sample state in the forecast step, $i = \{1, ..., N\}$

y_t- the observation at time t

$\tilde{y}_t-$ the predicted observation at time t

P_t- the estimated state variance at time t

$\tilde{P}_t-$ the predicted state variance in the forecast step, at time t

K_t- the Kalman gain at step t

$R-$ the observation covariance matrix

$Q-$ the state covariance matrix

A.1 Kalman Filter (KF)

Algorithm 4 : Kalman Filter

Initial guess for x_0, P_0

for $t = 1, ..., T$ **do**

$\tilde{x}_t = Fx_{t-1}$ ▷ Forecast step

$\tilde{P}_t = FP_{t-1}F^T + Q$

$K_t = \tilde{P}_t H^T (H\tilde{P}_t H^T + R)^{-1}$ ▷ Analysis step

$x_t = \tilde{x}_t + K(y_t - H\tilde{x}_t)$

$P_t = (I - KH)\tilde{P}_t$

end for

A.2 Stochastic Ensemble Kalman Filter (SEKF)

Algorithm 5 : Stochastic Ensemble Kalman Filter

Generate samples $x_0^{(i)} (i = 1, ..., N)$ randomly.

for $t = 1, ..., T$ **do**

for $i = 1, ..., N$ **do**

$\tilde{x}_t^{(i)} = \mathcal{F}(x_{t-1}^{(i)}) + \eta_t$ ▷ Propagate samples

$\tilde{y}_t^{(i)} = \mathcal{H}(\tilde{x}_t^{(i)})$

end for

$K_t = Cov(\tilde{x}_t, \tilde{y}_t)(Var(\tilde{y}_t) + R)^{-1}$ ▷ Calculate Kalman gain

for $i = 1, ..., N$ **do**

$x_t^{(i)} = \tilde{x}_t^{(i)} + K(y_t - \tilde{y}_t^{(i)} + \epsilon_t)$

end for

$x_t = \frac{1}{N} \sum_{i=1}^{\mathcal{N}} (x_t^{(i)})$ ▷ Sample mean as state estimation

end for

$Cov(\tilde{x}_t, \tilde{y}_t)$ denotes the covariance of $\tilde{x}_t$ and $\tilde{y}_t$; $Var(\tilde{y}_t)$ is the variance of $\tilde{y}_t$.

A.3 Variational Sequential Monte Carlo (VSMC)

VSMC is proposed in Naesseth et al. (2018). An important fact used in the paper is that the ELBO is equivalent to the average of importance weights, which converges to the marginal likelihood as the sample size increases (Le et al., 2017), i.e.

$$ELBO = \sum_{t=1}^{T} \log \left(\frac{1}{N} \sum_{i=1}^{N} w_t^{(j)} \right) \xrightarrow{N \to \infty} \log p(y_{1:T}), \tag{16}$$

where N is the total number of particles, and $w_t^{(j)}$ denotes the weight of the j-th particle at step t. Thus, in each optimisation step, the particle filter (Algorithm 7) can be used for calculating the empirical ELBO.

Algorithm 6 : Variational Sequential Monte Carlo

Initial guess for proposal parameter α_1

for $m = 1, ..., M$ **do**

Run Algorithm 7 with proposal distribution $q(x_t|x_{t-1}, \alpha_m)$

Calculate surrogate ELBO: $\tilde{\mathcal{L}} \approx \sum_{t=1}^{T} \log \left(\frac{1}{N} \sum_{i=1}^{N} w_t^{(j)} \right)$ ▷ Maximise $\tilde{\mathcal{L}}$

Estimate the gradient $\nabla(\tilde{\mathcal{L}})$

Compute step size s_m

$\alpha_{m+1} = \alpha_m + s_m \nabla(\tilde{\mathcal{L}})$ ▷ Update proposal parameter

end for

Algorithm 7 : Particle filter at step m for optimisation

for $t = 1, ..., T$ **do**

for $j = 1, ..., N$ **do**

Sample $x_t^{(j)}$ from $q_t(x_t|x_{t-1}, \alpha_m)$ ▷ Propose particles

$w_t^{(j)} \propto \frac{p(y_t|x_t^{(j)})p(x_t^{(j)}|x_{t-1}^{(j)})}{q_t(x_t^{(j)}|x_{t-1}^{(j)})}$ ▷ Calculate weights

end for

$\tilde{w}_t^{(j)} = \frac{w_t^{(j)}}{\sum_{i=1}^{N} w_t^{(j)}}$

Resampling $\{x_t^{(j)}\}$ according to $\{\tilde{w}_t^{(j)}\}$

end for

B Results for the second scenario from Section 3

We report the detailed simulation results for the second scenario, in which the hidden W_t evolves stochastically. Similar to the steps taken in Scenario 1, we first generate 100 sets of data according to the introduced process; then, we apply the state-space models $\mathcal{M}_1$ and $\mathcal{M}_2$ to each set of data and collect different measures. In this scenario, we set the same observation variance and state variance for all filters: $\sigma_\epsilon^2 = 1$ and $\sigma_\eta^2 = 1 \times 10^{-6}$.

Figure 9 compares the MSEs for different representations and shows that $\mathcal{M}_{1,K=2}$ delivers the lowest MSEs on average. Figure 10 reports the average Frobenius norm and ρ_t of the estimated matrices, and we could see that the estimates from $\mathcal{M}_{1,K=1}$ diverge the most from the true value (the grey lines).

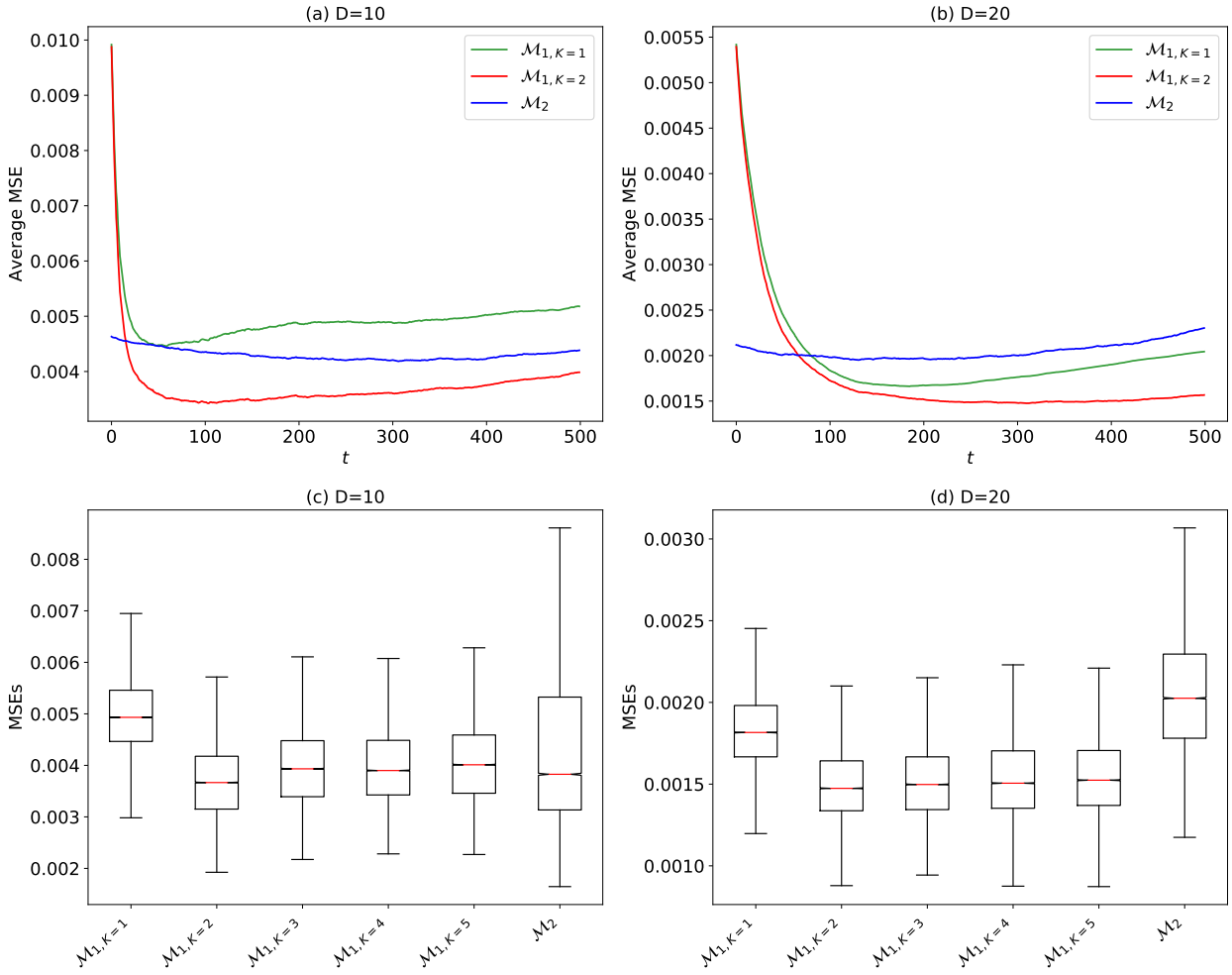

Figure 9: (a) and (b): the average of the MSEs ($B = 100$ simulations) at each step. (c) and (d): Boxplots of MSEs. Each box summarises the quantiles of 30000 MSEs (100 simulations and 300 MSEs in each simulation). Results referring to the warm-up period (i.e., 200 steps) are omitted.

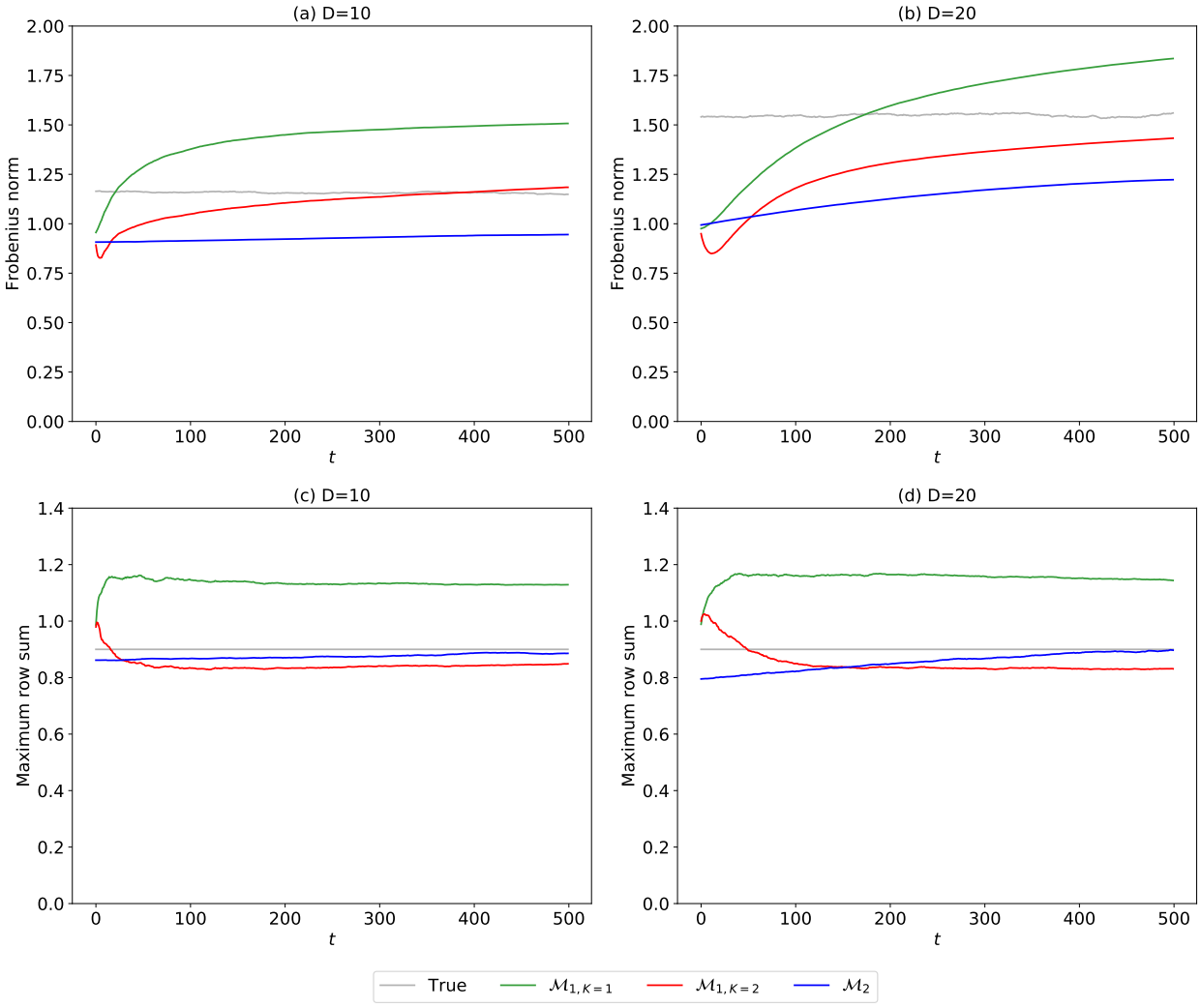

Figure 10: (a) and (b) the average Frobenius norm; (c) and (d) the average ρ_t.

C Real-world Dataset Constituents

Number	Bank Name	Abbreviation	Country
1	Intesa Sanpaolo	I. Sanpaolo	Italy
2	Monte Paschi	M. Paschi	Italy
3	Unicredit	Unicredit	Italy
4	Mediobanca	Mediobanca	Italy
5	BNP PARIBAS	BNP	France
6	Credit Agricole	C. Agricole	France
7	SOCIETE GENERALE	S. Generale	France
8	Credit Lyonnais	C. Lyonnais	France
9	HSBC	HSBC	UK
10	Bank of Scotland	B. Scotland	UK
11	Natwest	Batwest	UK
12	RBS	RBS	UK
13	Barclays	Barclays	UK
14	Lloyds	LLoyds	UK
15	Deutsche Bank	Deutsche B.	Germany
16	Commerzbank	Commerz B.	Germany