Improving Generalisation in Multi-Output Gaussian Processes for Time Series Prediction

Felix L. Opolka[1,2], Wessel P. Bruinsma[1,2], Yuting Wu[2], Pietro Liò[1], and Eric Perim[2]

[1] University of Cambridge
[2] Invenia Labs

Abstract. Multi-output time series, and spatio-temporal graphs in particular, are powerful modelling tools for a variety of real-world scenarios. Gaussian processes linear mixing models are a common way to address these tasks, modelling both correlations in time and between time series by correlating a small number of latent processes using a mixing matrix. However, existing methods can suffer from overfitting and are difficult to combine with additional graph structure information in case of spatio-temporal graphs. In this short paper, we propose a novel, more parameter-efficient parameterisation of the mixing matrix by representing processes by states in a latent space and deriving correlations between processes from their relative positions. This formulation allows us to incorporate adjacency information between the time series by placing a prior on the process states and smoothing them using a low-pass graph filter. We show that our approach improves on existing GP mixing models on two traffic forecasting data sets by reducing overfitting and improving inductive bias.

1 Introduction

Gaussian process (GP) models are a common first-choice solution for time series modelling tasks, owing to a variety of strong characteristics. Their prior specification allows incorporating existing knowledge, such as periodicity or smoothness, in an elegant and interpretable fashion. Moreover, they are amenable to Bayesian inference, allowing accounting for uncertainty in time series prediction, which is crucial in a wide range of applications. Finally, their non-parametric nature provides partial protection against overfitting.

A particular challenge for GPs are multi-output time series tasks, as they arise in a variety of real-world scenarios including in geostatistics [8,13] or latent force models [16]. Here, the model is not only required to capture the relations between the different time points but also between each of the time series. Difficulties arise in overcoming the high computational complexity of inference in the multi-output GP (MOGP) model. Performing inference for p time series of n time steps each requires $\mathcal{O}(n^3p^3)$ time and $\mathcal{O}(n^2p^2)$ memory.

Work by Bruinsma et al. [2] introduced a formulation of a MOGP that achieves scaling linear in the number of time series p both for computation and memory by exploiting structure in the model setup. It employs free-form matrix optimisation to learn correlations between time series. Particularly in cases where a large number of time series are modelled, this can lead to overfitting. In this work, we introduce an alternative, more parameter-efficient parameterisation by representing time series with vectors (or states) in a latent space and deriving correlations between time series from the relative positions of their states.

Moreover, we study spatio-temporal graphs as special cases of multi-output time series where graph structure information is available. Each time series is uniquely associated with a node, and nodes are connected amongst each other to form a graph. Viewed differently, a spatio-temporal graph is a graph whose node attributes (*i.e.* values associated with nodes) change over time while the graph structure remains the same. We show that model fit improves when encoding graph structure in the latent space by choosing an appropriate prior distribution for the states.

We provide an introduction to multi-output Gaussian process models and graph signal priors in Section 2, before discussing the latent space parameterisation and spatio-temporal graph prior in Section 3. Section 4 provides experimental results on two traffic forecasting data sets.

2 Preliminaries

2.1 Multi-output Gaussian process model

Multi-output Gaussian Process models for time series tasks make the assumption that any set of data points across time series and time steps follow a multivariate Gaussian distribution. A popular subclass of MOGPs are Instantaneous Linear Mixing Models (ILMM), which follow a generative process as follows:

$$\begin{aligned} \mathbf{x}(t) &\sim \mathcal{GP}\left(\mathbf{m}(t), \mathbf{k}(t, t')\right) && \text{(latent processes)} \\ \mathbf{f}(t) \,|\, \mathbf{H}, \mathbf{x}(t) &= \mathbf{H}\mathbf{x}(t) && \text{(mixing mechanism)} \\ \mathbf{y}(t) \,|\, \mathbf{f}(t) &\sim \mathcal{GP}\left(\mathbf{f}(t), \delta[t-t']\mathbf{\Sigma}\right). && \text{(noise model)} \end{aligned}$$

The model maps a set of m independent latent processes $\mathbf{x}(t)$ to the output (or observed) processes $\mathbf{y}(t)$ via a mixing matrix $\mathbf{H} \in \mathbb{R}^{p \times m}$. Bruinsma et al. [2] have shown that inference in this model is possible by conditioning on a sufficient statistic $\mathbf{Ty}(t) = (\mathbf{H}^\top \mathbf{\Sigma}^{-1}\mathbf{H})^{-1}\mathbf{H}^\top\mathbf{\Sigma}^{-1}\mathbf{y}(t)$ for $\mathbf{x}(t)$ rather than conditioning on the original outputs $\mathbf{y}(t)$, where $\mathbf{Ty}(t)$ is a projection of $\mathbf{y}(t)$ (see [2, Lemma 1] for details). The projected observations follow the prior

$$\mathbf{Ty}(t) \sim \mathcal{GP}\left(\mathbf{x}(t), \delta[t-t']\mathbf{\Sigma}_T\right), \tag{1}$$

with *projected noise* $\mathbf{\Sigma}_T = (\mathbf{H}^\top\mathbf{\Sigma}^{-1}\mathbf{H})^{-1}$. Performing inference by conditioning on the projected observations, *i.e.* computing $p(\mathbf{x}(t) \,|\, \mathbf{Ty}(t))$, is cheaper than conditioning on the original observations, *i.e.* computing $p(\mathbf{x}(t) \,|\, \mathbf{y}(t))$ when the number of latent processes is chosen to be smaller than the number of output processes, meaning $m < p$, reducing time and memory cost to $\mathcal{O}(n^3m^3)$ and

$\mathcal{O}(n^2m^2)$ respectively. However, this can still be prohibitive in cases where the data mandates a larger number of latent processes.

Inference cost can be further reduced to be linear in the number of latent processes m if the mixing matrix $\mathbf{H}$ in the generative model above is chosen to be an orthogonal matrix. The model is then called an Orthogonal Instantaneous Linear Mixing Model (OILMM) [2]. The reduction in inference time complexity, as well as for training and prediction, stem from the fact that if $\mathbf{H}$ is orthogonal, the projected noise $\mathbf{\Sigma}_T$ becomes diagonal. As a result, the posterior $p(\mathbf{x}(t) \,|\, \mathbf{Ty}(t))$ factorises over the m latent processes, allowing inference to be performed independently for each latent process $x_i(t)$ by conditioning on $(Ty)_i(t)$ for $i = 1, \ldots, m$.

Under the OILMM, the mixing matrix $\mathbf{H}$ can be learnt via maximum likelihood estimation (MLE) using a parameterisation that guarantees orthogonality. Depending on the number of observed processes p and latent processes m, the OILMM requires finding point estimates for pm parameters of the model, which can easily lead to overfitting. If we would like to model 200 observed time series with 20 latent processes, this amounts to $4,000$ parameters. In Section 3, we present a parameterisation that requires only $\mathcal{O}(m + p)$ parameters, reducing the number of parameters by an order of magnitude, and show that this improves model fit by reducing overfitting.

2.2 Priors over graph signals

An undirected, weighted graph of N nodes is a tuple $G = (V, E, w)$ consisting of a set of nodes $V = \{v_1, v_2, \ldots, v_N\}$, a set of edges $E \subseteq V \times V$ and a weight assignment $w : V \times V \to \mathbb{R}$, which assigns a weight to all edges in the graph and 0 to all other node pairs. For undirected graphs, $(v_i, v_j) \in E$ always implies $(v_j, v_i) \in E$. Graph G can be equivalently represented by its adjacency matrix $\mathbf{A} \in \mathbb{R}^{N \times N}$, where $A_{ij} = w(v_i, v_j)$ if $(v_i, v_j) \in E$ and 0 otherwise. The adjacency matrix establishes a canonical ordering of nodes for convenience.

A multi-dimensional graph signal $x : V \to \mathbb{R}^D$ is a function mapping nodes to vectors in $\mathbb{R}^D$. An equivalent representation is a matrix $\mathbf{X} \in \mathbb{R}^{N \times D}$, which stacks the vectors of each node following the same canonical ordering of nodes as the adjacency matrix.

A wide range of real-world graph signals fulfil the *homophily* assumption, which states that signal values of two nodes are more likely to be similar if those nodes are closer together in the graph [10] and, consequently, homophilous signals tend to vary smoothly across the graph. Closeness on the graph is measured by the geodesic distance (the smallest number of edges connecting two nodes) and adopted as the node closeness metric in the remainder of this work. Results from graph signal processing (see for example [11]) show that we can derive low-pass filters for graph signals, which can further filter for homophilous signals. A common choice for a low-pass filter is defined as $\mathbf{S} = \mathbf{D}^{-\frac{1}{2}}(\mathbf{I} + \mathbf{A})\mathbf{D}^{-\frac{1}{2}}$, where $\mathbf{D} = \mathrm{diag}(\sum_{j=1}^{N} A_{ij})$ is the diagonal degree matrix [4,14] and the filtered graph signal is then given by $\mathbf{SX}$.

3 Methodology

In the following, we introduce a variant of the OILMM, referred to as Latent Space OILMM (LS-OILMM), that parameterises the mixing matrix using states in a latent space, while retaining fast and memory efficient inference. We aim to make two improvements over the vanilla OILMM design. Firstly, LS-OILMM has fewer parameters per output time series, hence potentially reducing the risk of overfitting to the training data. Secondly, LS-OILMM introduces structure into the parameterisation of the mixing matrix, which is amenable to introducing prior assumptions. We show how graph structure information can be incorporated by choosing suitable prior distributions for the process states.

3.1 Latent space linear mixing model

At the core of the LS-OILMM is a specific parameterisation of the mixing matrix $\mathbf{H}$. We define a latent space M that will commonly be a real D-dimensional space, meaning $M \coloneqq \mathbb{R}^D$. We define two sets of process states living in this latent space, the output (process) states $\{\mathbf{z}_i^{\text{out}} \in M\}_{i=1,\dots,p}$ and the latent (process) states $\{\mathbf{z}_j^{\text{lat}} \in M\}_{j=1,\dots,m}$. Both sets of states can be stacked to form $\mathbf{Z}^{\text{out}} \in \mathbb{R}^{p \times D}$ and $\mathbf{Z}^{\text{lat}} \in \mathbb{R}^{m \times D}$. For clarity, note that, depending on context, latent (meaning unobserved) may describe the processes $\mathbf{x}(t)$, as in latent process states, or the space that both latent and output process states lie in. Moreover, we emphasise that the process states are learned parameters of the models rather than embeddings derived from the input time series.

The mixing weight H_{ij} between output process i and latent process j is now determined by the relationship between the two associated process states and is given by

$$H_{ij} = g(\mathbf{z}_i^{\text{out}}, \mathbf{z}_j^{\text{lat}}), \tag{2}$$

where $g(\cdot,\cdot)$ is referred to as the *latent similarity measure*. Intuitively, g outputs larger values for states closer together in the latent space. For the mixing matrix as a whole, we can now write $\mathbf{H} = g(\mathbf{Z}^{\text{out}}, \mathbf{Z}^{\text{lat}})$. Compared to the free-form matrix optimisation used by the OILMM with $\mathcal{O}(mp)$ MLE parameters, this parameterisation requires only $\mathcal{O}(m+p)$ MLE parameters, thus potentially reducing the risk of overfitting.

Different choices for the latent similarity measure are possible. One such option is to choose a kernel (a positive definite function), such as the radial basis function (RBF) kernel, setting $g(\mathbf{z}, \mathbf{z}') = \exp\left(-\|\mathbf{z} - \mathbf{z}'\|^2/(2\sigma^2)\right)$. The RBF kernel maps the process states to an infinite-dimensional Hilbert space (see [7,12]) before computing their inner product, while retaining the low parameter count. The RBF kernel is particularly suitable for applications that give rise to data with a strictly positive empirical covariance matrix. In case where negative covariances do occur, we can choose alternative similarity measures, for example kernels that can map to negative values.

Choosing a similarity measure to parameterise the mixing matrix allows interpreting the relative positions of process states. The closer an output state i

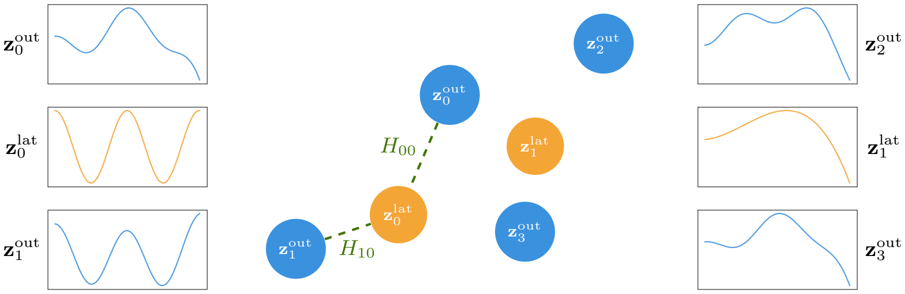
$\mathbf{z}_0^{\text{out}}$
$\mathbf{z}_0^{\text{lat}}$
$\mathbf{z}_1^{\text{out}}$
$\mathbf{z}_2^{\text{out}}$
$\mathbf{z}_0^{\text{out}}$
H_{00}
$\mathbf{z}_1^{\text{lat}}$
$\mathbf{z}_0^{\text{lat}}$
$\mathbf{z}_3^{\text{out}}$
$\mathbf{z}_1^{\text{out}}$
H_{10}
$\mathbf{z}_2^{\text{out}}$
$\mathbf{z}_1^{\text{lat}}$
$\mathbf{z}_3^{\text{out}}$

Fig. 1: Visualisation of the latent space M with the two groups of process states. The mixing weight between output process i and latent process j is governed by the kernel distance between output state $\mathbf{z}_i^{\text{out}}$ and latent state $\mathbf{z}_j^{\text{lat}}$. As a result, output processes whose states are closer together in M will be more similar.

is to a latent state j relative to another latent state k, the larger will be entry H_{ij} compared to H_{ik}. As the output processes are weighted linear combinations of the latent processes, output process i will be more similar to latent process j than latent process k. Following the same reasoning, if two output states are close together in latent space, their corresponding processes will be similar as they will both have a similar weighting of latent processes. This property of the latent space is visualised in Figure 1. Output state $\mathbf{z}_0^{\text{out}}$ is closer to $\mathbf{z}_0^{\text{lat}}$ than is $\mathbf{z}_3^{\text{out}}$ and therefore output process $y_0(t)$ bears slightly more resemblance to the periodic latent process $x_0(t)$ than does $y_3(t)$.

As for the OILMM, we can achieve time and memory complexity linear in m if the mixing matrix is constrained to be orthogonal. Importantly, the orthogonalisation needs to be a function continuous in the unconstrained matrix, as we do not want small changes in the process states and hence the unconstrained mixing matrix to lead to big changes in the orthogonal mixing matrix. This intuitively retains the interpretability of the latent space even after orthogonalising the mixing matrix. Such a continuous orthogonalisation can be achieved by computing the singular value decomposition of the unconstrained parameter $\mathbf{H} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top$ with $\mathbf{U}$ and $\mathbf{V}$ containing only the leading m singular vectors. We then redefine the mixing matrix, previously defined in Equation 2, to $\mathbf{H} = \text{Orthogonalise}(\mathbf{H}) \coloneqq \mathbf{U}\mathbf{V}^\top$. Hence, the final mixing matrix is computed as

$$\mathbf{H} = \text{Orthogonalise}\left(g(\mathbf{Z}^{\text{out}}, \mathbf{Z}^{\text{lat}})\right). \tag{3}$$

Its continuity stems from the fact that it maps the unconstrained matrix to the closest orthogonal matrix [15, Theorem 4], which naturally does not change much for small deviations to the input matrix.

3.2 Incorporating graph structure information

When the time series live on a graph, *i.e.* for spatio-temporal graphs, we wish to incorporate the adjacency information into the prior of our model to improve its inductive bias. For the original OILMM model, a possible approach is to assign a prior to the mixing matrix $\mathbf{H}$ and compute its maximum *a posteriori* (MAP) estimate instead of the MLE. In practice, however, it is not obvious how to construct a prior for the mixing matrix. In contrast, for the LS-OILMM, we can straightforwardly build on the latent space interpretation described in the previous section.

In a first step, we assign isotropic normal priors to all components of the output states, meaning $\mathbf{Z}^{\text{out}}_{:i} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_p)$ for $i = 1, \ldots, D$. To incorporate graph structure, we smoothen the output states using a low-pass graph signal filter $\mathbf{S}$ as described in Section 2.2. The smoothed output states follow the prior

$$\mathbf{S}\mathbf{Z}^{\text{out}}_{:i} \sim \mathcal{N}\left(\mathbf{0}, \mathbf{S}\mathbf{S}^T\right). \tag{4}$$

Intuitively, this prior correlates output states that correspond to neighbouring nodes in the graph. When used as part of the loss function when computing MAP estimates, it acts as "rubber bands" between the output states, pulling closer together those states that are closer in the graph. Consequently, this choice of prior encodes a homopholy assumption for spatio-temporal graphs: Two time series that belong to two nodes closer together in the graph will be more strongly correlated than time series of two nodes with many hops between them.

4 Experiments

We conduct experiments on two traffic forecasting data sets to examine the properties of the proposed models. The METR-LA data set [3,5] consists of 207 nodes corresponding to traffic measurement stations in Los Angeles County. Each node is associated with a time series of car speed measurements in 5-minute intervals. The PEMS-BAY [5] data set follows the same format but consists of 325 measurement stations in the Bay Area. Both data sets come with an underlying graph constructed by thresholding the road distance between nodes.

For both data sets, we choose the first 1000 time steps for our experiments. Time series are divided into a training interval, consisting of the first 70% of time steps, and a test interval, consisting of the remaining time steps. Moreover, a subset of 21 and 33 test nodes are selected for the two data sets respectively; the remaining nodes form the training set. For the training nodes, the whole time series, both training and test interval, is available as training data. For the test nodes, only the training interval is available as training data. The model is tasked with reconstructing the test interval for the test nodes given the training data. Note that the test interval comes strictly after the training interval, hence the model is required to extrapolate on the test nodes.

We use the L-BFGS optimiser [6] to tune the hyperparameters, including the process states in the case of the LS-OILMM and the unconstrained mixing

	METR-LA		**PEMS-BAY**	
	PPLP	**MAE**	**PPLP**	**MAE**
OILMM	−2.00 ±0.07	6.56 ±0.22	−46.22 ±3.73	8.92 ±0.48
LS-OILMM	−1.69 ±0.07	6.55 ±0.34	−13.62 ±1.56	5.46 ±0.45
LS-OILMM-adj	**−1.61** ±0.06	**6.16** ±0.19	**−10.39** ±0.94	**4.78** ±0.16

Table 1: Experimental results comparing the baseline OILMM model with the LS-OILMM that ignores graph structure and the LS-OILMM-adj, which incorporates graph structure information via its prior. Results given with one standard deviation. Results significantly better (using Student t-test) than other results in the same column are highlighted in boldface.

matrix in the case of OILMM. The latent space dimensionality is set to $D = 3$ and we use $m = 10$ latent processes. The RBF kernel is chosen as the latent similarity measure. For the latent processes we use a composition of periodic and smooth kernels that can be summarised as

$$\begin{aligned}&\mathrm{Matern12}()\\&\quad+\mathrm{RBF}()\cdot\mathrm{Periodic}(\mathrm{Matern12}(),\mathrm{period}=60/5\cdot 24)\\&\quad+\mathrm{RBF}()\cdot\mathrm{Periodic}(\mathrm{Matern12}(),\mathrm{period}=60/5\cdot 24\cdot 7).\end{aligned}\tag{5}$$

All experiments are implemented using Python 3.9 and the GPflow library [9] with TensorFlow [1].

The goal of our experiments is twofold: Firstly, we examine the benefit of the latent space parameterisation, proposed in this paper, over the free-form matrix parameterisation, proposed in the original OILMM paper, by comparing the LS-OILMM without the graph structure prior (hence using MLE) to the baseline OILMM. Secondly, we assess the benefit of incorporating graph structure by comparing the vanilla LS-OILMM with the LS-OILMM which employs the graph structure prior and computes MAP estimates, referred to as LS-OILMM-adj. Experimental results for all three models averaged over 10 parameter initialisations are shown in Table 1. We report both the normalised posterior predictive log probability (PPLP) and mean absolute error (MAE). The MAE is a metric taking into account only the predictive mean, whereas the PPLP is also a function of the predictive covariance.

We find that the LS-OILMM performs better than the OILMM both in terms of PPLP and MAE on both data sets, indicating that the LS-OILMM more accurately reconstructs the time series in the domain of the test set. The improvement is particularly pronounced on the PEMS-BAY data set, which has a larger number of output processes (325 vs. 207) and might therefore be more susceptible to overfitting when using free-form matrix optimisation, which requires about ten times as many parameters as the latent space parameterisation. By design, the LS-OILMM aims to prevent overfitting to the training data by reducing the number of point-estimated parameters. We verify that this hypothesis holds by plotting the PPLP of the *test* set against the number of steps executed

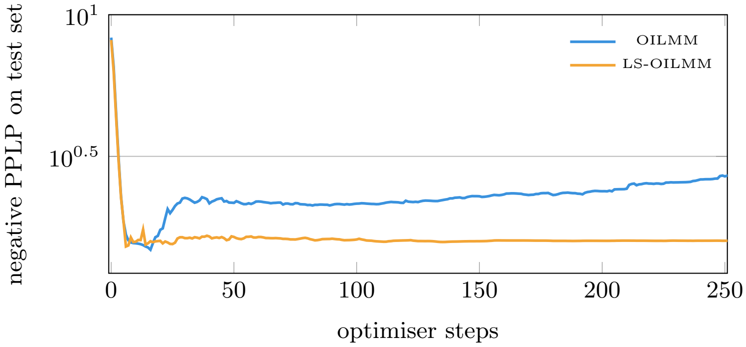

Fig. 2: Plot of the test set PPLP during training. The OILMM starts to overfit after about 20 optimiser steps whereas the LS-OILMM continues to improve.

by the optimiser in Figure 2. We find that the PPLP reaches a similar level for both the OILMM and LS-OILMM after at most 20 optimiser steps. However, as the marginal likelihood is further maximised, the OILMM starts to overfit whereas the LS-OILMM continues to improve in terms of test set predictions.

The LS-OILMM-adj again improves on the LS-OILMM both in terms of PPLP and MAE on both data sets. Once more, the improvement is more marked on the PEMS-BAY data set compared to the METR-LA data set. The results suggest that incorporating graph structure via a prior that raises correlation between nearby nodes increases the inductive bias of the model and improves its predictive capabilities.

The results indicate that the more parameter-efficient mixing matrix parameterisation of the LS-OILMM prevents overfitting and still achieves an equally good fit for the two data sets studied here. Furthermore, the graph-structured prior on the process states made possible by the latent space formulation further improves model fit for time series associated with graphs.

5 Conclusions

We have shown that representing time series by states in a low-dimensional latent space yields a suitable parameterisation for covariances in a multi-output Gaussian process model. This more parameter-efficient parameterisation compared to a free-form matrix paramterisation leads to improved model fit on the test set by reducing overfitting of the model on two traffic forecasting data sets. We have further demonstrated how we can derive a prior over spatio-temporal graphs by placing a normal prior that correlates neighbouring nodes on the output process states and applying a low-pass graph filter, and computing MAP estimates. Experimental results confirm that the graph-informed prior improves model fit for the two traffic forecasting tasks. Future work could investigate how to relax the setting to fully dynamic graphs for which not only the node attributes change over time but also the graph structure itself.

References

1. Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G.S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., Zheng, X.: TensorFlow: Large-scale machine learning on heterogeneous systems (2015), `https://www.tensorflow.org/`, software available from tensorflow.org
2. Bruinsma, W., Perim, E., Tebbutt, W., Hosking, S., Solin, A., Turner, R.: Scalable exact inference in multi-output gaussian processes. In: Proceedings of the 3th International Conference on Machine Learning. vol. 119 (2020)
3. Jagadish, H.V., Gehrke, J., Labrinidis, A., Papakonstantinou, Y., Patel, J.M., Ramakrishnan, R., Shahabi, C.: Big data and its technical challenges **57**(7) (2014)
4. Kipf, T.N., Welling, M.: Semi-Supervised Classification with Graph Convolutional Networks. In: Proceedings of the 5th International Conference on Learning Representations (2017)
5. Li, Y., Yu, R., Shahabi, C., Liu, Y.: Diffusion convolutional recurrent neural network: Data-driven traffic forecasting. In: International Conference on Learning Representations (2018)
6. Liu, D.C., Nocedal, J.: On the limited memory bfgs method for large scale optimization. Math. Program. **45**(1–3) (1989)
7. Mackay, D.J.C.: Introduction to gaussian processes (1998)
8. Matheron, G.: Le krigeage universel, vol. 1. Cahiers du Centre de Morphologie Mathématique, École des Mines de Paris (1969)
9. Matthews, A.G.d.G., van der Wilk, M., Nickson, T., Fujii, K., Boukouvalas, A., León-Villagrá, P., Ghahramani, Z., Hensman, J.: GPflow: A Gaussian process library using TensorFlow. Journal of Machine Learning Research **18**(40) (2017)
10. McPherson, M., Smith-Lovin, L., Cook, J.M.: Birds of a feather: Homophily in social networks. Annual Review of Sociology **27**(1) (2001)
11. Ortega, A.: Introduction to Graph Signal Processing. Cambridge University Press (2022)
12. Rasmussen, C.E., Williams, C.K.I.: Gaussian processes for machine learning. Adaptive computation and machine learning, MIT Press (2006)
13. Wackernagel, H.: Multivariate Geostatistics: An Introduction with Applications. Springer Berlin Heidelberg (2003)
14. Wu, F., Souza, A., Zhang, T., Fifty, C., Yu, T., Weinberger, K.: Simplifying graph convolutional networks. In: Proceedings of the 36th International Conference on Machine Learning. vol. 97 (2019)
15. Zou, H., Hastie, T., Tibshirani, R.: Sparse principal component analysis. Journal of Computational and Graphical Statistics **15**(2) (2006)
16. Álvarez, M., Luengo, D., Lawrence, N.D.: Latent force models. In: Proceedings of the Twelfth International Conference on Artificial Intelligence and Statistics. vol. 5 (2009)