Short-term Forecasting of Urban Traffic using Spatio-Temporal Markov Field

Cyril Furtlehner* Jean-Marc Lasgouttes*

Alessandro Attanasi† Marco Pezzulla† Guido Gentile‡

July 2021

Abstract

The probabilistic forecasting method described in this study is devised to leverage spatial and temporal dependency of urban traffic networks, in order to provide predictions accurate at short term and meaningful for a horizon of up to several hours. By design, it can deal with missing data, both for training and running the model. It is able to forecast the state of the entire network in one pass, with an execution time that scales linearly with the size of the network. The method consists in learning a sparse Gaussian copula of traffic variables, compatible with the Gaussian belief propagation algorithm. The model is trained automatically from an historical dataset through an iterative proportional scaling procedure, that is well suited to compatibility constraints induced by Gaussian belief propagation. Results of tests performed on two urban datasets show a very good ability to predict flow variables and reasonably good performances on speed variables. Some understanding of the observed performances is given by a careful analysis of the model, making it to some degree possible to disentangle modeling bias from the intrinsic noise of the traffic phenomena and its measurement process.

Keywords

Traffic prediction, Gaussian belief propagation algorithm, statistical learning, Markov random field.

1 Introduction

The rapidly evolving sector of traffic management information systems has made the problem of traffic forecasting at the urban network level a central

*Inria, France. E-mail: *firstname.lastname*@inria.fr

†PTV-SISTeMA, Italy. E-mail: *firstname.lastname*@ptvgroup.com

‡DICEA, Sapienza University Rome, Italy. Email: guido.gentile@uniroma1.it

and key issue. Traffic management systems aim to monitor, allow real-time scenario evaluations, and find control actions that keep the traffic flow as fluid as possible. They generally operate on main axes or principal sub-networks of the transportation infrastructure, and typically rely on traffic flow models, which in principle are able to predict the onset and the propagation of congestion. To be accurate, these models must have a high level of detail, which requires calibrating an enormous amount of parameters. But this calibration is so challenging [1, 2] and time-consuming that the effectiveness of flow models can be greatly reduced. In this respect, data driven models replace the calibration effort with a simpler tuning of the hyperparameters.

Moreover, the massive increase in available traffic data over the last decade has triggered a surge of interest in traffic forecasting based on data-driven models. General references on these various approaches can be found in [3]. These can be divided into two main categories:

- *parametric*: autoregressive models, probabilistic models, Bayesian models, models based on Markov random fields (MRF);
- *non-parametric*: k nearest neighbors (k-NN), random forest, Gaussian process, support vector regression, neural networks.

Parametric models, based on ordinary statistical considerations, are more traditional. They are sometimes preferred to their non-parametric counterparts owing to their interpretability. Machine learning (ML) can potentially offer a very large variety of non-parametric models with a wide range of complexity and potential efficiency. Many methods are targeted toward independent modeling of road segments. Methods attempting to leverage spatial dependencies have been recently growing considerably in number [4]. If we focus more specifically on forecasting models which attempt to address the problem at the network scale, the requirements we find for such models to be deployed in online applications are the following:

- *accuracy*: predictions should be significantly better than a simple persistent predictor combined with an historical time-of-day-dependent average, for instance, used when data are incomplete;
- *missing data*: we cannot expect to have a complete information of the network state at any time, which means that both the training and the running of the model have to be able to take missing data into account;
- *scaling*: the model should scale up to high systems size, i.e. networks of the size of an urban area, where the number of road segments involved can be around one or two hundred thousand. If we think in terms of detectors, this requirement might be lower. Currently, the number of effective detectors covering a given urban area is smaller by one or two orders of magnitude than the number of road segments (see e.g. [5]).

Traditional methods based on autoregressive models [6–8] or hybrid ML methods (see e.g. [9]) have recently been adapted to this context, without being able to

scale up well to large network sizes. More recently, deep learning approaches have been proposed [10–12] showing convincing performance compared to traditional methods at the price of heavy computational costs [**?**, 13] and requiring complete data or to be combined with imputation methods [15, 16].

Beside that, it is shown in [17] that a simple Gaussian multivariate model with very few parameters can already perform better than simple interpolation methods at large scale with a cost that is linear w.r.t. system size, thanks to mean-field techniques of statistical physics [18], on which the present paper is also relying. More refined models, like that proposed in [19] based on a mixture of sparse Gaussian random variables, cannot cope with inference on large scale graphs.

2 Our Approach

This paper proposes a different direction that copes with missing data, based on Markov random field (MRF) modeling. A graphical model is used to couple observations on segments at different locations and different time steps, such that missing and future observations are treated alike. One difficulty is to train the model offline from historical data; another one is to be able to run it in real time, i.e. to perform the probabilistic inference of all the missing/future variables from the observed ones. In a previous study, we investigated [20] the possibility of building an MRF which could encode both spatial and temporal dependencies and come with a linear computational time when running the probabilistic inference task. We considered two types of models involving either latent binary variables [21] or Gaussian copula models. Both come with an associated learning algorithm to generate models adapted respectively to Generalized Belief Propagation for binary variables [22] and Gaussian Belief Propagation for real-valued variables [23].

Gathering all these elements leads us to propose an approach having the following features, which, to our knowledge, are not simultaneously provided by any other method:

- provide accurate predictions, by exploiting spatial and temporal dependency;
- deal with up to 80% of missing data without drastic degradation of performances;
- deal with online constraints for large network sizes, owing to the linear scaling of belief propagation;
- deal with inhomogeneous data, e.g. speed, flow, travel time, etc., thanks to the definition of a generic traffic index;
- incorporate in an economical way, through the aforementioned traffic index, contextual data like e.g. time of day and seasonal dependencies;
- provide consistent confidence intervals along with the predicted value.

The paper is organized as follows: Section 3 gives the necessary materials, respectively the Gaussian belief propagation algorithm used for inference, the $\star$-IPS algorithm used to construct the model and the copula encoding. The workflow of the method is described in Section 4. The results of experiments performed on two different traffic datasets with various features are then reported and analyzed in Section 5. Comparison will be made in particular with results from single detector times series [24] forecast models to measure the advantage gained from spatial dependencies. Section 6 presents a statistical analysis of the errors of the model in which we try to separate intrinsic uncertainty of the traffic phenomena from systematic errors based on modeling issues. Finally, Section 7 outlines future work.

3 Material and methods

3.1 Gaussian Belief propagation

We consider a set $\mathcal{V}$ of nodes associated to discrete random variables $\mathbf{x} = \{x_i, i \in \mathcal{V}\}$, obeying a joint probability distribution of the form

$$\mathcal{P}(\mathbf{x}) = \prod_{ij \in \mathcal{E}} \psi_{ij}(x_i, x_j) \prod_{i \in \mathcal{V}} \phi_i(x_i), \tag{1}$$

where $\mathcal{E} \subset \mathcal{V} \times \mathcal{V}$ is a set of edges and ϕ_i and ψ_{ij} are functions associated respectively to a single variable x_i and to an unordered pair of variables $ij \in \mathcal{E}$. The ψ_{ij} are called the "factors" while the ϕ_i are there by convenience and could be reabsorbed in the definition of the factors. $\mathcal{E}$ together with $\mathcal{V}$ define the graph $\mathcal{G}$, which will be assumed to be connected, and ∂i denotes the set of neighbors of node i in $\mathcal{G}$.

In the case where the graph does not have loops (i.e., it is a tree), the set of marginal distributions, called the belief, $b_i(x) = \mathcal{P}(x_i = x)$ associated to each variable i can be computed efficiently. The Belief Propagation algorithm (BP) [25] does this for all variables in one single procedure, by taking into account that the computation of each of these marginals involves intermediate quantities called the messages $m_{i \to j}(x_j)$ "sent" by node i to node j, and which are also necessary to compute other marginals. The idea of BP is to compute all these messages simultaneously, using the relation between them as a fixed point equation. The following message updates

$$m_{i \to j}(x_j) \leftarrow \sum_{x_i} \phi_i(x_i) \psi_{ij}(x_i, x_j) \prod_{k \in \partial i, k \neq j} m_{k \to i}(x_i), \tag{2}$$

are iterated until a fixed point is reached. Then the beliefs are given by

$$b_i(x_i) = \frac{1}{Z_i} \phi_i(x_i) \prod_{j \in \partial i} m_{j \to i}(x_i), \tag{3}$$

where Z_i is a normalization constant.

When the graph does have loops, it is still possible to apply the algorithm above. In this case, when convergence occurs, $b_i(x)$ is only an approximation of $\mathcal{P}(x_i = x)$.

In practice, these equations are only usable in two situations: when each x_i takes a finite number of values, and when $\mathbf{x}$ is a Gaussian vector. These are the only cases where the message update formulas can be parameterized easily. The case of binary values, which correspond to an Ising model, has been studied in [20, 21]. Our focus here is on Gaussian variables, for which the algorithm takes on a specific form, referred to as Gaussian Belief Propagation (GaBP) [26]. For such variables, the factors are naturally parameterized as

$$\psi_{ij}(x_i, x_j) = \exp\bigl(-A_{ij}x_i x_j\bigr), \tag{4}$$

$$\phi_i(x_i) = \exp\Bigl(-\frac{1}{2}A_{ii}x_i^2 + h_i x_i\Bigr), \tag{5}$$

where the precision matrix $A = (A_{ij})$ is the inverse of the correlation matrix of variable $\mathbf{x}$ and $\mathbf{h} = (h_i)$ is such that $A^{-1}\mathbf{h}$ is the expected value of the variable $\mathbf{x}$. Update rule (2) makes sense for these factors with messages of the form

$$m_{i\to j}(x_j) = \exp\Bigl(-\frac{(x_j - \mu_{i\to j})^2}{2\sigma_{i\to j}}\Bigr). \tag{6}$$

Information is then propagated via the 2-component real vector $(\mu_{i\to j}, \sigma_{i\to j})$ representing bias and variance with the following update rules:

$$\mu_{i\to j} \longleftarrow \frac{1}{A_{ij}}\Bigl[h_i + \sum_{k\in\partial i, k\neq j} \frac{\mu_{k\to i}}{\sigma_{k\to i}}\Bigr], \tag{7}$$

$$\sigma_{i\to j} \longleftarrow -\frac{1}{A_{ij}^2}\Bigl[A_{ii} + \sum_{k\in\partial i, k\neq j} \sigma_{k\to i}^{-1}\Bigr]. \tag{8}$$

At convergence, the beliefs take the form:

$$b_i(x) = \sqrt{\frac{\sigma_i}{2\pi}}\exp\Bigl(-\frac{(x-\mu_i)^2}{2\sigma_i}\Bigr), \tag{9}$$

with

$$\mu_i = \sigma_i\Bigl[h_i + \sum_{j\in\partial i}\frac{\mu_{j\to i}}{\sigma_{j\to i}}\Bigr], \tag{10}$$

$$\sigma_i^{-1} = A_{ii} + \sum_{j\in\partial i}\sigma_{j\to i}^{-1}. \tag{11}$$

There is at most one fixed point, even in the presence of loops. It might not necessarily be stable but, if convergence occurs, μ_i is equal to the expectation $\mathbb{E}(x_i)$ at each node i [27]. The computation time is $O(N\log(N))$ with the system size N.

3.2 ⋆-IPS algorithm for model selection

In order to use GaBP, it is first necessary to build a model in the form (4)–(5). Since GaBP may often encounter convergence issues, especially with non-sparse structures, it can be of practical interest to construct off-line a Gaussian MRF (GMRF) that is compatible with GaBP. Some features of the matrix A can be desirable to obtain convergence, albeit without much guarantee: sparsity, specific topological properties like the absence of short loops or spectral properties, e.g. walk-summability [28]. By combining various methods proposed in the context of sparse inverse covariance matrix estimation [29–31], one way to do that has been developed in [23] in the form of the ⋆-IPS algorithm[1]. ⋆-IPS incorporates the desired features explicitly, by combining an approach based on the iterative proportional scaling (IPS) procedure [32], with block updating techniques used in [29, 30].

The rationale is, instead of inverting the covariance matrix, to construct the matrix A link by link, by selecting the link that maximizes the log-likelihood of the corresponding distribution, and ensuring at each step that the constraints are satisfied. Several constraints can be considered:

- ℓ-LOOP forbids loops of size up to ℓ;
- ℓ-FLOOP is the same, but only for *frustrated* loops, i.e. loops with a negative product of partial covariances ($-A_{ij}$);
- WS (walk-summability) and WWS (weak walk-summability) are spectral constraints, which involve definite positiveness of some matrices.

Each modification is accepted only if it satisfies the constraints. The best candidate link can thus be discarded if the constraints are violated by this addition. In this paper, we will use the 5-FLOOP-IPS algorithm, which prohibits frustrated loop with size up to 5.

In practice, ⋆-IPS alternates many link additions, corresponding to a significant increase in mean connectivity, with block update procedures, which allow to fine-tune the coefficients. Sparse precision matrices of good likelihood are generated in $O(N^3)$ steps, with the advantage of having the complete optimization path available, by means of graphical models of intermediate connectivity. Note finally that we have discarded the standard way of generating sparse precision matrices based on the Lasso penalty [30, 33]. There are two reasons for that: firstly the L_1 norm penalty suffers from a modeling bias, due to excessive penalization of truly large magnitudes entries of A; secondly it is not flexible enough for the kind of constraints we are interested in, in order to produce graphical models compatible with GaBP of high likelihood [23].

3.3 Gaussian copula model of traffic indices

A key feature of our method is a mapping of raw data, that can correspond to flow or speed for instance, to a standard normal variable, a kind of properly normalized

[1] Available at `https://gitlab.inria.fr/bptraffic/star-ips`.

traffic index on which the prediction is performed. The joint probability measure of these traffic indices is approximated through a Gaussian copula. We define this index in the following way. Let t be a discretized time, measured in time steps δ_t of fixed length. N_t represents the number of such time steps contained in a single day. For a given t, the time of day $\tau \in \{0, \dots N_t - 1\}$ is given by $\tau = t$ modulo N_t. For instance, we have $N_t = 96$ if we consider $\delta_t = 15\,\text{min}$ time steps. At each time step, we assume the system to be represented by N_v variables X_i^t, $i \in \{1, \dots N_v\}$ corresponding to traffic detectors. Now, for each variable X_i and each time of day τ, we build from the historical data a running average $\bar{X}_i^\tau$ and a variance V_i^τ. If the dataset is clustered into a certain number of weekly or seasonal patterns representing contextual data, then these quantities will be estimated for each cluster labeled by some extra index ℓ. Then, for each variable index i, time t (and associated time of day τ) and cluster label ℓ, we map $X_i^{t,\ell}$ to the following variable

$$U_i^{t,\ell} \stackrel{\text{def}}{=} \frac{X_i^{t,\ell} - \bar{X}_i^{\tau,\ell}}{\sqrt{V_i^{\tau,\ell}}}, \tag{12}$$

which represents a centered and normalized variable for given τ and ℓ. From this transformed historical data, we build for each index i a single cumulative distribution function

$$F_i(x) = P(U_i < x), \tag{13}$$

which for a given realization $U_i^{t,\ell} = x$ represents the traffic index. The purpose of this index is to encapsulate all average time-dependent trends, week-day and seasonal dependencies, while the Gaussian copula will represent the fluctuations around these trends. In order to build a sparse Gaussian copula of all the indices, we first transform each variable $U_i^{t,\ell}$ into a normal centered variable via the following mapping:

$$Y_i^{t,\ell} = F_{\mathcal{N}(0,1)}^{-1} \circ F_i(U_i^{t,\ell}), \tag{14}$$

where $F_{\mathcal{N}(0,1)}$ is the cumulative distribution of a standard normal variable. The copula model corresponding to $n + h + 1$ time layers (n past layers, 1 present and h future layers) is then obtained by considering the vector

$$Z^t = (Y_i^{t+k,\ell}, i = 1 \dots N_v, k = -n, \dots, 0, \dots h) \tag{15}$$

and constructing its associated sparse multivariate approximate model with $\star$-IPS. This model will be used to generate predictions $\hat{Y}_i$, which in turn can be converted into predictions $\hat{X}_i$ of the original variables by inverting (14,12).

4 Workflow of the method

The complete workflow of our method (see Figure 1) comprises two separate tasks: creating the model, and using it to produce detailed forecasts. We give the details of our implementation below.

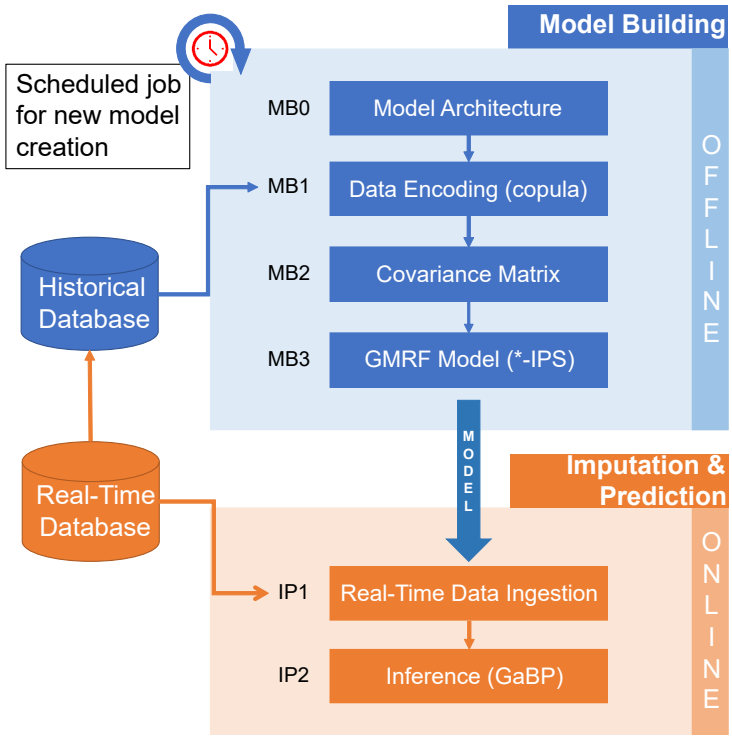

Figure 1: Overall workflow of the proposed methodology. The "Model Building" top part of the picture (in blue) represents the offline job for creating our model starting from an historical database. This offline part can be scheduled with a frequency of days/weeks. The "Imputation & Prediction" bottom part of the picture (in orange) is the continuous online part where a forecast is produced, by running the model on the latest real-time data.

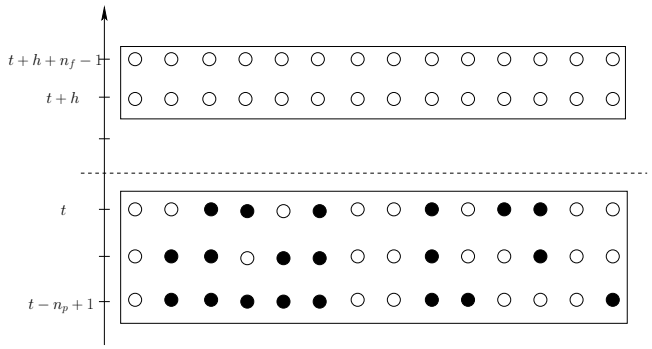

Figure 2: Time layers setting of one basic space time configuration. Each circle represents one single variable at a given time, filled circles correspond to observations.

4.1 Model building (MB, offline)

This task is supposed to be run daily, weekly or even monthly, in order to refresh the model with recent data. It typically takes several hours to complete.

(MB0) Model architecture Choose an architecture for the model, that is:

- n_p: number of past and present layers of variables for which observations may be available; if t is the present time, this covers times $t-n_p+1,\dots,t$.
- n_f: number of future layers we want to predict, $n_f \geq 1$;
- h: horizon of prediction of the first future layer.

From this, we construct by concatenation of $(n_p+n_f)N_v$ variables the dynamical configurations (see Figure 2):

$$\{X_i^{t-n_p+1},\dots,X_i^t,X_i^{t+h},\dots,X_i^{t+h+n_f-1}, i=1,\dots,N_v\}. \tag{16}$$

This subtask is only supposed to be done once, when setting up the original model. There is no need to tweak the values later when updating historical data.

(MB1) Data encoding (copula) Compute the required statistics of the model from the training set: the mean and variance $\bar{X}_i^{\tau,\ell}$ and $V_i^{\tau,\ell}$ for each variable i, time of day τ and label ℓ for the day (when a clustering is in use). Then, for each i, compute the c.d.f. F_i of $U_i^{t,\ell}$ aggregated over all t and approximate it by an invertible monotonous linear piecewise function.

(MB2) Covariance matrix Generate using (14) a training set of vectors

$$\{Y_i^{t-n_p+1},\dots,Y_i^t,Y_i^{t+h},\dots,Y_i^{t+h+n_f-1}, i=1,\dots N_v\} \tag{17}$$

and compute the corresponding $(n_p + n_f)N_v \times (n_p + n_f)N_v$ covariance matrix. Because the data is incomplete, some regularization can be required if some modes are associated to large negative eigenvalues. In such a case the eigenvalue is replaced by its absolute value.

(MB3) GMRF model ($\star$-IPS) The model is built using the $\star$-IPS algorithm [23], which takes the regularized covariance matrix as input and generates an almost continuous set of models with increasing mean connectivity $\bar{d}$. The variant that we use (5-FLOOP-IPS) avoids frustrated short loops of sizes up to 5 (see Section 3.2).

4.2 Imputation and prediction (IP, online)

At this point, the model is ready to use for prediction tasks. It is defined by a multivariate Gaussian distribution $P(Y)$ in which the variables corresponding to observations are clamped, while the other (future and non-observed past variables) are inferred using GaBP, as described in Section 3.1. We insist also on the fact that the same model is used independently of day-time or any other contextual data (see Section 3.3).

(IP1) Real-Time data ingestion Initialize the GMRF with observations, using strong beliefs: if no observation exists for node i, then $h_i = 0$ by construction (see Section 3.3); otherwise, x_i is fixed to its observed value $\hat{x}_i$ and detached from the factor graph and neighboring variables see their local field modified:

$$h_j \leftarrow h_j - A_{ij}\hat{x}_i, \text{ for all } j \in \partial i \tag{18}$$

Another possibility not used here is to use soft beliefs: the observable $\hat{x}_i$ comes with an uncertainty either given by some variance $\hat{\sigma}_i$.

(IP2) Inference (GaBP) Run GaBP according to equations (7)–(8) until convergence is reached. The predictions are then obtained from (10) for each variable to be predicted, along with an estimation of the uncertainty provided by the variance (11). Note that, for observed variables inserted with soft constraints, the value of the variance (11) provided by the fixed point may give some indication of whether the input observation is trustworthy or not.

4.3 Practical considerations

There are a couple of practical issues to be aware of when using this workflow.

Model tuning There are basically two hyper-parameters of the model to tune, namely the number of past layers n_p and the mean connectivity d^*. Indeed, h is imposed by the task and n_f is an optional choice, defaulting to 1. d^* can be loosely selected somewhere in the region where the log likelihood monitored by $\star$-IPS as a function of d becomes flat. A more systematic procedure, leading

to similar parameters, relies on a grid search, based on accuracy of predictions measured on the training set. At low connectivity, we observe similar pattern in all datasets: a sharp rise in accuracy, followed by an almost flat plateau while increasing both d and n_p. Their values are therefore chosen at the beginning of the flat domain in order to get a good trade-off between speed and precision with a loss in accuracy below 1% w.r.t. the would-be optimal setting while computational time is reduced by a significant factor.

Complexity and affordable systems sizes Since $\star$-IPS has a cubic complexity, the model building task (MB) has a running time that scales like $\big((n_p + n_f)N_v\big)^3$. In practice, the total number of variables $(n_p + n_f)N_v$ should not exceed the order of 10^4 so as to be able to train the model in a reasonable time, which means that, if we consider $n_p = 2$ and $n_f = 1$, we can deal with systems of roughly 3000 detectors.

Concerning the imputation and prediction task (IP), GaBP can converge on networks of size 10^5 within a few tenth of a second on an ordinary CPU, while typically, the online constraint for traffic information is to generate a forecast every few minutes, according to the refresh rate of the data.

5 Experiments

In order to check the robustness of the method, we have tested our system on two different datasets corresponding to urban traffic, both for imputation and prediction. Horizon of prediction range from 15 minutes to 5 hours, with input observation covering up to one past hour. The datasets considered here are the cities of Vienna and Turin. More extensive results, including figures for the Turin dataset and an additional dataset about highways in the San Francisco Bay Area, are contained in the research report [34].

5.1 Error metrics

We consider a set of metrics commonly used in traffic analysis. General purpose metrics are the root mean square error (RMSE), the mean absolute error (MAE), and the mean absolute percentage error (MAPE). Letting X be the variable to predict and $\hat{X}$ its predictor, the metrics are defined as follows.

$$\mathrm{RMSE} \stackrel{\text{def}}{=} \sqrt{\mathbb{E}[(\hat{X} - X)^2]} \tag{19}$$

$$\mathrm{MAE} \stackrel{\text{def}}{=} \mathbb{E}[|\hat{X} - X|] \tag{20}$$

$$\mathrm{MAPE} \stackrel{\text{def}}{=} 100 \times \mathbb{E}\left[\frac{|\hat{X} - X|}{X}\right] \tag{21}$$

Besides, for traffic flows, we use the GEH statistic, introduced in the 1970s by Geoffrey E. Havers, and used today as a standard indicator by Transport for

London, one of the leading transport authorities worldwide [35]. When X is an hourly traffic flow, GEH is defined as

$$\mathrm{GEH} \stackrel{\text{def}}{=} \sqrt{\frac{2(X-\hat{X})^2}{X+\hat{X}}}. \tag{22}$$

This number has been empirically designed to yield comparable numbers for a broad range of arterial road capacities. Typically, the prediction is considered good if GEH is below 5, and the statistic of interest is thus the percentage of $\mathrm{GEH} < 5$.

The expectations above are taken uniformly over all detectors and all time steps for which there is an observation to compare against the prediction. Note that MAPE is not defined when $X = 0$, which can happen in particular for flow measurements. To regularize this, we simply threshold the denominator in MAPE to $X_{\min} = 10$ in practice.

5.2 Baseline predictors

In order to provide some reference points of comparison for our results we use some simple predictors and when possible, i.e. for the Vienna dataset, other results from the literature. Our simple baseline predictors are the following:

- Mean(t) is the historical average $\bar{X}_i^{\tau,\ell}$ for detector i at time of day τ and label ℓ, as defined in Section 3.3.
- t_0 is the persistent predictor, sometimes called the random walk predictor. It is equal to the last observed value within the past time window of the detector, and to Mean(t) if no such value is found.
- k-NN is a k nearest neighbors predictor. Given the past and present observations, which can comprise many time layers of incomplete data, we look in the training set for the k closest states to the current ones by means of an $\mathbb{L}_2$ discrepancy. The hyper-parameter k needs to be optimized.

For the Vienna dataset, other methods, based respectively on Bayesian networks (BN), neural networks (NN) and clustering of day profiles, have been previously tested [24] on only 36 count locations. A direct comparison (see Table 2) shows that GaBP improves significantly on these techniques, especially for longer forecast horizons.

5.3 Vienna dataset

Our first dataset concerns urban traffic in the Vienna (Austria) urban area, as depicted in Figure 3. It consists in flow data collected over 4 years (2011–2014) from 292 count locations (CLOC) measuring the number of vehicles every 15 minutes. Since these detectors are quite often down, we arbitrarily select those which are up during at least 10% of the time. This leads to retaining 266 CLOCs

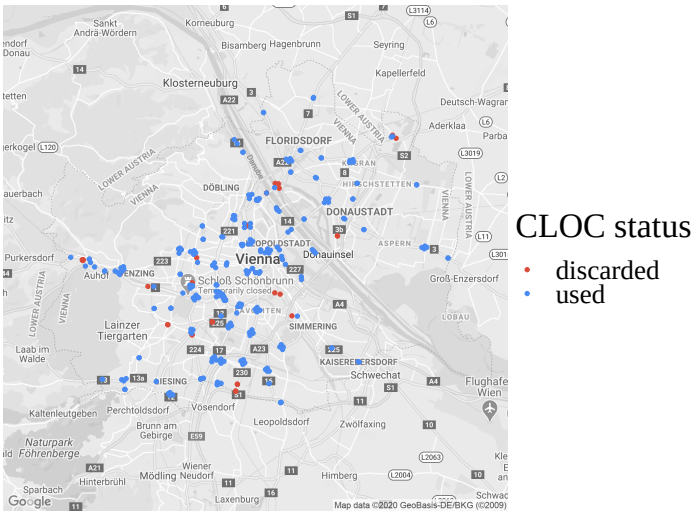

Figure 3: Location of the CLOCs in Vienna. Only the blue ones have enough data to be usable.

out of 292. With this setting, at any given time, we have typically around 65% of the detectors which are up. This means that, in addition to forecasting a certain number of layers of variables ahead in time, our model performs an imputation of at least 35% of missing observations (more when we randomly hide observations). The first 3 years (2011–2013) are used as a training set, while 2014 is used for testing. As a preprocessing step, we have performed a clustering analysis of the daily traffic conditions using the Affinity Propagation algorithm [36]. This analysis yields 10 meaningful clusters, presented in Table 1. This clustering will be used in two ways: first it will serve to build a baseline predictor $\bar{X}_i^{(\tau,\ell)}$ for each CLOC i, time-step $\tau = 1, \ldots, 96$ and cluster index $\ell = 1, \ldots, 10$; secondly our model, as explained before, will be built on the residues $U_i^{(t,\ell)}$ defined in (12) from this baseline or some related ones, obtained in the same way after merging some of the smallest clusters into larger ones, in order to have a sufficient number of observations in each cluster to avoid statistical problems when building the model.

For this setting, several architectures lead to similar performances, as long as at least 3 layers in the past are taken into account, corresponding to a look back interval of at least 30 minutes. Specializing the model to one single horizon instead of directly performing the forecast for many features ahead does not seem to help much. The results presented in Figure 4 were obtained with a model with 4 past layers and one single specialized future layer. The results summarized in Table 2 were obtained with a multistep-ahead model having 4 past and 4 future layers. Specialized or multistep-ahead models yield identical performance within error bars. The multistep-ahead model appears to be more advantageous in practice, as it can deliver all horizon predictions in one pass. The only drawback

	Jan.	Feb.	Mar.	Apr.	May	Jun.	Jul.	Aug.	Sep.	Oct.	Nov.	Dec.
Sun.[1]	9	9	9	5	5	5	5	5	5	9	9	9
Mon.	1	1	1	4	4	4	7	7	8	8	10	10
Tue.	1	1	1	4	4	4	7	7	8	8	10	10
Wed.	1	1	1	4	4	4	7	7	8	8	10	10
Thu.	1	1	1	4	4	4	7	7	8	8	10	10
Fri.	3	3	3	3	3	3	3	3	3	3	3	3
Sat.	2	2	2	2	2	2	6	6	6	2	2	2

[1] Public holidays are classified as Sundays; the ranges for Saturday and Sunday are by season, so that Cluster 5, for example, starts on March 21$^{\text{st}}$.

Table 1: Description of the 10 clusters for Vienna dataset according to month and day of the week.

comes from the cubic increase in computational cost necessary to build it, since it contains more variables.

As our model is able to perform both completion of missing data and forecasting at the same time within a single GaBP fixed point convergence, we first study the dependency of the forecasting performance on the fraction of observed variables. To this end, we randomly ignore some of the available observations in the past and present layers, in order to have a given density ρ of observed variables per layer, as illustrated in Figure 2. As shown in Figure 4 (top), a lot of precision is gained by observing only a small fraction of variables, say 5–10%, and after that the forecasting and imputation errors continue to decrease more slowly but steadily. Conversely, the prediction error of the k-NN model saturates after $\rho = 0.2$. The forecasting performance of the model as a function of the horizon is shown in Figure 4 (bottom). We see that the GaBP error is 33% smaller than that of Mean(t) on the short-term prediction. In addition, our model provides meaningful information up to 5 hours in advance w.r.t. this baseline, which means that it is able to identify additional traffic patterns not captured by the clustering.

Table 2 provides the numerical results for the 266 CLOCs, as well as comparison with the predictors proposed in [24] tested on 36 of these CLOCs. These methods treat each detector independently, so the comparison gives an indication of the added value of taking spatial dependencies into account. In all cases, we see some gain at all horizons regarding all metrics of evaluation, rather marginal for $h = 15'$ but rapidly sizable when the horizon increases. Visual inspection of individual CLOC errors shows that those having less than 70% of GEH < 5 are either badly behaving sensors clearly sending corrupted data, or sensors for which a systematic bias is present in our Mean(t) baseline, possibly reflecting a long lasting modification in the traffic conditions for the corresponding segment.

	Flow (266 CLOCs)				Flow (same 36 CLOCs as [24])			
	%GEH < 5	RMSE	MAE	MAPE	%GEH < 5	RMSE	MAE	MAPE
$h = 15'$								
BN	-	-	-	-	-	33.86	23.06	16
NN	-	-	-	-	-	32.38	22.13	16
Cluster	-	-	-	-	-	35.51	23.84	16
Mean(t)	78.88	40.97	21.30	36.66	73.51	49.65	29.59	23.56
t_0	80.32	31.71	17.57	20.71	70.49	47.54	30.19	18.94
k-NN	83.46	31.89	17.03	27.27	81.33	36.99	23.07	17.87
GaBP	**89.11**	**25.97**	**13.62**	**18.85**	**87.19**	**31.89**	**19.45**	**15.31**
$h = 30'$								
BN	-	-	-	-	-	38.98	25.81	17
NN	-	-	-	-	-	37.10	24.34	17
Cluster	-	-	-	-	-	39.65	25.88	18
Mean(t)	78.88	40.97	21.30	36.66	73.51	49.65	29.59	23.56
t_0	74.74	35.17	20.02	24.10	70.33	45.68	29.39	20.44
k-NN	82.77	32.60	17.43	28.39	80.50	37.90	23.60	18.31
GaBP	**87.82**	**27.74**	**14.41**	**20.22**	**85.68**	**34.07**	**20.55**	**16.19**
$h = 60'$								
BN	-	-	-	-	-	47.20	30.92	20
NN	-	-	-	-	-	46.68	28.46	20
Cluster	-	-	-	-	-	46.68	29.81	20
Mean(t)	78.88	40.97	21.30	36.66	73.51	49.65	29.59	23.56
t_0	62.42	47.84	27.56	32.61	58.47	63.00	40.24	27.57
k-NN	81.44	33.32	17.98	29.31	78.90	39.32	24.57	19.16
GaBP	**85.90**	**29.70**	**15.51**	**21.66**	**83.70**	**36.58**	**22.01**	**17.27**

Table 2: Vienna dataset: results for different horizons of forecasting and comparison on a subset of CLOCs with single time-series forecasting based methods (BN and NN from [24]).

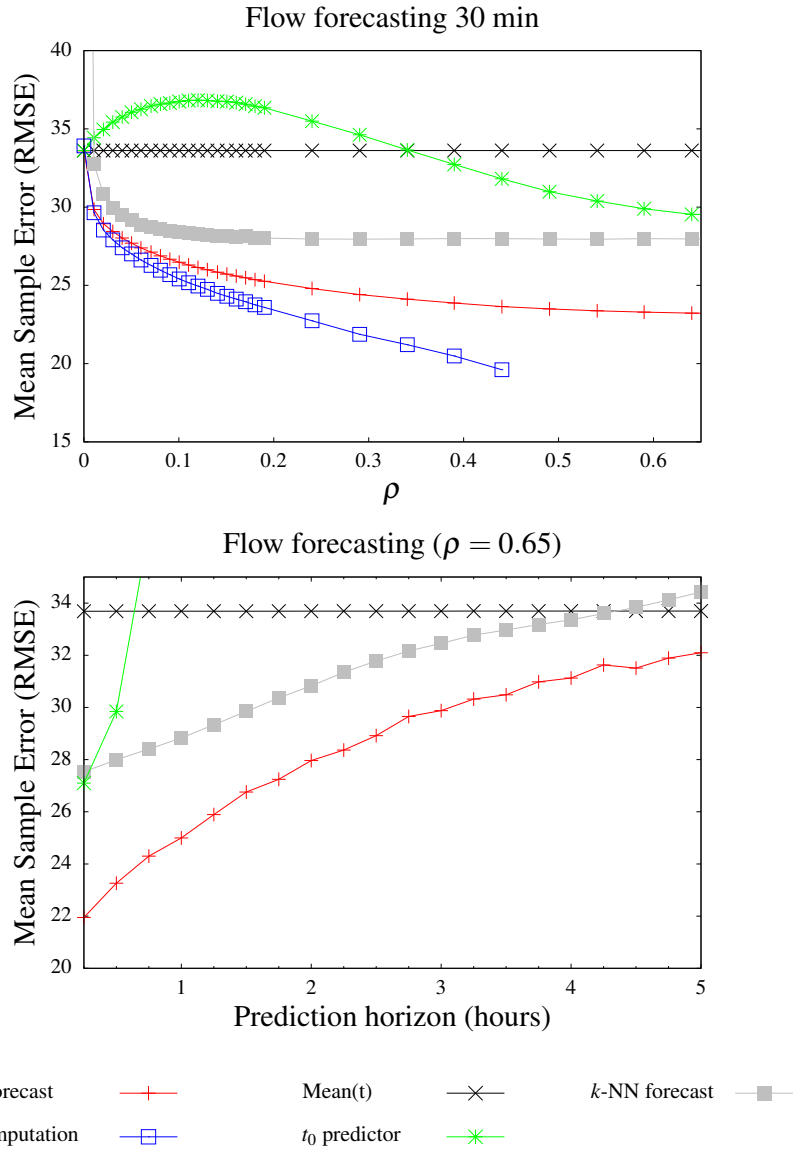

Figure 4: Vienna dataset. Top: average flow forecasting error for a fixed horizon of 30 minutes, as a function of the fraction ρ of observed variables in the past time layers, averaged over 5000 test samples. The imputation error for missing data is also shown (in blue). A point of comparison is given by the k-NN predictor, optimized with $k = 50$. Bottom: Average flow forecasting error as a function of forecasting horizon at maximum possible observation rate $\rho \sim 0.65$ (average over the full test set).

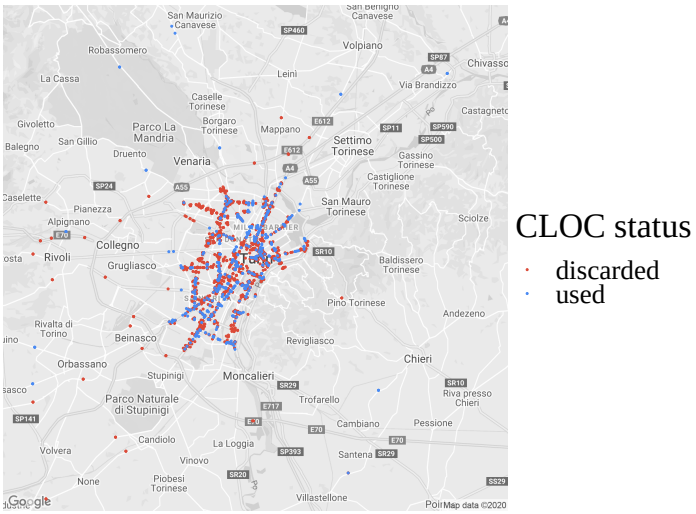

Figure 5: Location of the CLOCs in the center of Turin. Only the blue ones have enough data to be usable. Note that there are 148 additional sensors outside of this area that are not displayed here for the sake of clarity.

5.4 Turin dataset

The second dataset again concerns urban traffic, in the Turin (Italy) urban area. There are 1489 count locations, giving average speed measurements in km/h in addition to the flow measurements as number of vehicles every 15 minutes. The map of these CLOCs in the center of Turin is shown in Figure 5. Two years of data (2015–2016) have been collected. As for the previous dataset, we select detectors based on an activity threshold of 10%. We end up with 685 detectors, of which 566 correspond to flow and 119 to speed measurements. With this setting, the ratio of available data is around 60% in the past layers. Year 2015 is used for training and year 2016 for testing. A straightforward clustering is done here with 4 clusters: one for Monday–Thursday and the other ones respectively for Fridays, Saturdays and Sundays. This leads to a slightly noisier Mean(t) baseline (around 45 in RMSE instead of 40 for Vienna). Table 3 compares different methods to a multistep-ahead GaBP with $(n_p, n_f) = (4, 4)$ with 5480 variables. The selected model is sparser than for Vienna, with mean connectivity 4. Yet the behavior and performance are comparable, with e.g. more than 85% of GEH < 5 at a 1 hour horizon. All indicators for the flow are very similar to those of the Vienna dataset, with slightly higher RMSE and lower MAPE and MAE, because of an higher overall capacity bias of the segments in the case of Turin. As far as speed is concerned, we observe for instance at a 30-minute horizon an RMSE improvement of 10% w.r.t. the Mean(t) baseline and 20% w.r.t. the t_0 predictor. The results for flow are clearly more impressive than those for speed predictions. This hides important disparities among various days. Moreover,

	Flow (566 CLOCs)				Speed (119 CLOCs)		
	%GEH < 5	RMSE	MAE	MAPE	RMSE	MAE	MAPE
$h = 15'$							
Mean(t)	75.46	45.09	21.65	31.02	8.40	5.21	13.73
t_0	81.41	27.58	15.43	20.32	9.24	5.00	11.74
k-NN	79.78	37.71	18.47	27.19	8.20	4.99	13.28
GaBP	**88.10**	**27.56**	**13.32**	**19.03**	**7.65**	**4.24**	**10.44**
$h = 30'$							
Mean(t)	75.46	45.09	21.65	31.02	8.40	5.21	13.73
t_0	73.67	35.02	19.22	24.50	9.68	5.35	12.70
k-NN	77.65	39.70	19.55	29.42	8.34	5.11	13.78
GaBP	**87.10**	**30.73**	**14.20**	**20.68**	**7.56**	**4.25**	**10.70**
$h = 60'$							
Mean(t)	75.46	45.09	21.65	31.02	8.40	5.21	13.73
t_0	62.00	47.92	26.67	32.22	10.36	5.88	14.12
k-NN	73.99	42.28	21.32	31.51	8.43	5.15	13.32
GaBP	**85.24**	**33.90**	**15.45**	**22.58**	**7.67**	**4.34**	**11.09**

Table 3: Turin dataset: results for different horizons of forecasting, for both flow and speed measurements.

the aggregated error for the speed is dominated by nighttime prediction errors, where the small number of speed measurements leads to a very noisy signal.

Finally, we examine a subset of the results from a transportation engineering point of view. We consider the set of 22 CLOCs from the dataset for which both flow and speed are available. For each of these CLOCs, the object of interest is the empirical fundamental diagram, i.e. the flow versus the density (obtained as a simple division between the flow and the speed) of all the data of the year 2016. Then, each data point is labeled as describing either a fluid (hypocritical) or a congested (hypercritical) traffic condition. The manual labeling of the traffic conditions has been performed by three different experts, all of them trained to recognize visually the traffic conditions. These three classifications are not 100% identical, since some boundary situations are difficult to classify. In those cases the majority label is used, which is always possible with an odd number of outcomes. The main use for this manual labeling is to assess the expectation that congested traffic conditions are harder to predict than fluid ones. To this end, we measure the performance of the flow forecast against actual data through the GEH indicator, evaluated either on the congested states, the fluid ones or all of them, as shown in Table 4. As expected, the congested states are more difficult to predict for all methods. Plotting the cumulative distributions of GEH for different forecast horizons shows that GaBP achieves consistently a GEH score that is smaller in distribution than the other methods in all traffic state

%GEH < 5	All Data	Fluid	Congested
$h = 15'$			
Mean(t)	70.93	71.11	63.12
t_0	79.18	79.37	72.03
GaBP	**88.28**	**88.36**	**85.32**
$h = 30'$			
Mean(t)	70.80	70.96	62.93
t_0	66.71	66.83	61.08
GaBP	**85.71**	**85.79**	**81.43**
$h = 45'$			
Mean(t)	71.09	71.26	62.99
t_0	57.47	57.50	55.73
GaBP	**84.81**	**84.95**	**79.90**
$h = 60'$			
Mean(t)	70.94	71.13	62.61
t_0	50.34	50.28	53.08
GaBP	**83.14**	**83.28**	**77.30**

Table 4: Turin dataset: results with manually labeled traffic state conditions.

conditions.

6 Error analysis and limits of the model

In this section, in order to estimate the margin and possible directions for improvement, we first discuss the main limit of accuracy of our model and then perform a data analysis of the errors made by the model. In particular, we would like to be able to separate components in the errors due to modeling bias from the intrinsic noise of the traffic phenomenon, which cannot be predicted.

Up to now, we have left aside the fact that our prediction μ_i based on GaBP in (10) comes with an uncertainty estimate through the variance σ_i (11). While μ_i is guaranteed to be the exact marginal of the model once GaBP has converged, σ_i on the other hand is only an approximate value of the true variance. These values can be exploited to deliver levels of confidence on our predictions. Applying the inverse map of (14) to the corresponding confidence interval, defined by say ± 1 standard deviation in copula space, we get straightforwardly a confidence interval of our prediction in the prediction space, e.g. flow, speed or occupancy. This confidence interval proves to be very reliable: about 67% of the predictions lay inside it, indifferently for speed or flow in Vienna or Turin, which is very close to the theoretical value of 68.27%, corresponding to one standard deviation in copula space. As shown, for example, on Figure 6, these confidence intervals are quite consistent and meaningful and may help to identify detector errors.

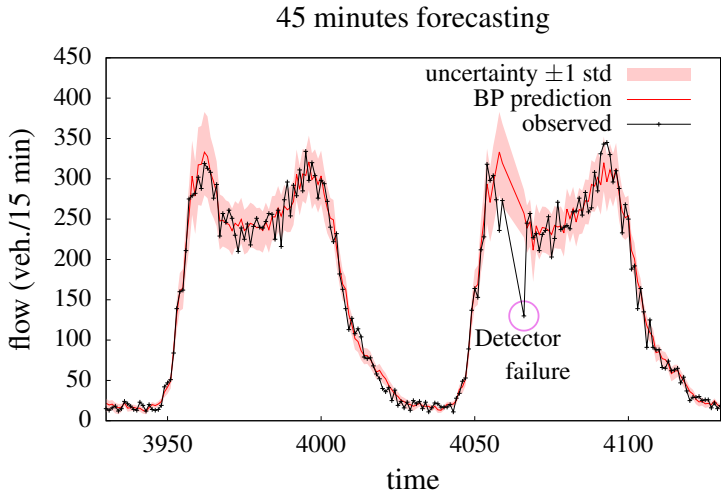

Figure 6: Vienna dataset: flow for CLOC "MA_33_23039_R3", together with uncertainty estimate delivered by GaBP. The confidence interval is 1 standard deviation in copula space, corresponding to 68.27% of confidence that the true value is within this interval.

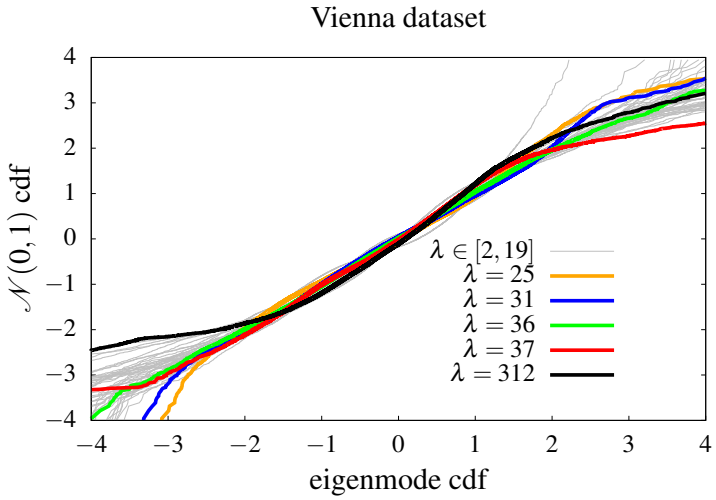

Figure 7: Vienna dataset: empirical cumulative distribution of the normalized projection of the (copula transformed) data along the 50 first principal modes against the cumulative distribution of a standard normal variable. Modes are ordered by decreasing eigenvalues λ_α.

Indeed, we see that the detector wakes up most probably in the middle of a time interval after being silent during 6 time steps. Then it delivers a clearly underestimated observation of traffic flow, 2 standard deviations away from our confidence interval. These confidence intervals reflect an estimation of some intrinsic noise of the traffic signal, which might not be possibly predicted. In any case, it is comparable to the mean error that we finally end up measuring on our predictions.

That said, there are systematic errors that our model makes which we would like to identify and cure. The main source of error comes from the Gaussian copula hypothesis. Recall first that, after the transformation (14) is performed, each $Y_i^{t,\ell}$ taken individually is by construction and up to numerical precision, a standard normal variable. However, the joint distribution has no specific reason in general to be multivariate Gaussian. In order to estimate how far from a multivariate Gaussian our model is, we consider the main directions of fluctuations of Y by extracting the dominant eigenmodes of the covariance matrix. Then for each of these modes $e_{\cdot,\alpha}$, we compute the corresponding cumulative distribution $P(z_\alpha^{t,\ell}/\sigma_\alpha < x)$ of the normalized components defined as

$$z_\alpha^{t,\ell} = \sum_{i=1}^{N} e_{i,\alpha} Y_i^{t,\ell}, \qquad \sigma_\alpha \stackrel{\text{def}}{=} \sqrt{\mathbb{E}\big[(z_\alpha^{t,\ell})^2\big]}. \tag{23}$$

Figure 7 shows for the Vienna dataset how these distributions compare to the expected cumulative distribution of a standard normal variable. As we can see the alignment is pretty good for most of the dominant modes over more or less 2 standard deviations. Beyond that, the model shows inadequacy in the distributions tails which are clearly non-Gaussian. Tests done without the clustering preprocessing step show as expected degraded performance, with a much more pronounced deviation from the normal distribution and a multi-modal behavior. This underlines the importance of the clustering of the data beforehand.

Another way to look at the results is to consider the Fourier spectrum of the error signal, and in particular how the error is distributed along the different frequencies. Figure 8 shows this for the different datasets and types of variables, compared to a white noise predictor, that is a with random decorrelated errors. First, we notice that the departure from the white noise predictor is more pronounced for flow than for speed. What is also noticeable is the finite contribution of very low frequencies. When looking more precisely at the data, it turns out that this corresponds to detectors which, for a long period, send values completely at odds with the ones observed during training. These badly behaving detectors may either correspond to corrupted ones, or to drastic changes of the traffic conditions on the corresponding segment, because of road work for instance. Then there remains the domain corresponding to time scales ranging between roughly 30 minutes and one day, represented by the right part of the plot. Overall, as seen in Figure 8, this corresponds to more than 80% of the total error. It is in this region that we can mostly pinpoint modeling biases.

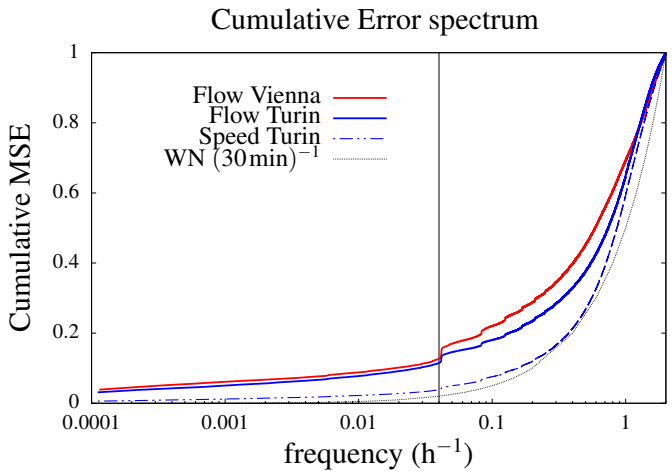

Figure 8: Cumulative power spectrum of the mean errors signal of the GaBP predictor. The vertical line indicates the 1-day time scale. The black dotted line represents the white noise (WN) reference.

7 Conclusions

The method presented in this paper is intended both to provide valuable predictors of traffic variables on large traffic networks, in order to feed traffic management systems for instance, and to provide some understanding of the statistical properties of traffic. With the advent of big data in the domain of traffic management, it is becoming possible to analyze these properties in detail at the macroscopic level. The structure of the generated network could potentially help us to visualize the information flow among the variables. Various sources of improvement can also be expected, like e.g. working with variables corresponding to positions in the fundamental diagrams at the level of each segment. The question of corrupted data was merely touched upon in this paper. Our model offers the possibility, via the message passing structure, of estimating the reliability of an input variable, depending on the information coming from other variables. Integrating this information in the belief propagation schema with the help of the soft belief setting mentioned in Section 4.2 might help us to improve the accuracy of the forecast by discarding suspicious observations.

Acknowledgments

The authors would like to thank the municipality of Vienna (Austria) and the Italian company 5T, which manages the mobility in Piemonte region (including Turin), for sharing their data with them.

References

[1] M. Jha, G. Gopalan, A. Garms, B. Mahanti, T. Toledo, and M. Ben-Akiva, "Development and calibration of a large-scale microscopic traffic simulation model," *Transportation Research Record*, vol. 1876, no. 1, pp. 121–131, 2004.

[2] C. Osorio, "High-dimensional offline origin-destination (OD) demand calibration for stochastic traffic simulators of large-scale road networks," *Transportation Research Part B: Methodological*, vol. 124, no. C, pp. 18–43, 2019.

[3] E. Vlahogianni, M. Karlaftis, and J. Golias, "Short-term traffic forecasting: Where we are and where we're going," *Transportation Research Part C: Emerging Technologies*, vol. 43, Part 1, pp. 3–19, 2014, special Issue on Short-term Traffic Flow Forecasting.

[4] A. Ermagun and D. Levinson, "Spatiotemporal traffic forecasting: review and proposed directions," *Transport Reviews*, vol. 38, no. 6, pp. 786–814, 2018.

[5] A. Loder, L. Ambühl, M. Menendez, and K. W. Axhausen, "Understanding traffic capacity of urban networks," *Sci Rep*, vol. 9, p. 16283, 2019.

[6] M. Levin and Y. Tsao, "On forecasting freeway occupancies and volumes (abridgment)," *Transportation Research Record*, no. 773, 1980.

[7] M. Lippi, M. Bertini, and P. Frasconi, "Short-term traffic flow forecasting: An experimental comparison of time-series analysis and supervised learning," *IEEE Transactions on Intelligent Transportation Systems*, vol. 14, pp. 871–882, 06 2013.

[8] G. Fusco, C. Colombaroni, and N. Isaenko, "Short-term speed predictions exploiting big data on large urban road networks," *Transportation Research Part C: Emerging Technologies*, vol. 73, pp. 183–201, 2016.

[9] S. Sun, R. Huang, and Y. Gao, "Network-scale traffic modeling and forecasting with graphical lasso and neural networks," *Journal of Transportation Engineering*, vol. 138, no. 11, pp. 1358–1367, 2012.

[10] Y. Lv, Y. Duan, W. Kang, Z. Li, and F. Y. Wang, "Traffic flow prediction with big data: A deep learning approach," *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 2, pp. 865–873, 2015.

[11] H. Yu, Z. Wu, S. Wang, Y. Wang, and X. Ma, "Spatiotemporal recurrent convolutional networks for traffic prediction in transportation networks," *Sensors*, vol. 17, no. 7, 2017.

[12] Y. Li, R. Yu, C. Shahabi, and Y. Liu, "Diffusion convolutional recurrent neural network: Data-driven traffic forecasting," in *International Conference on Learning Representations*, 2018.

[13] N. Polson and V. Sokolov, "Deep learning for short-term traffic flow prediction," *Transportation Research Part C: Emerging Technologies*, vol. 79, pp. 1–17, 2017.

[14] E. Manibardo, I. Laña, and J. Del Ser, "Deep learning for road traffic forecasting: Does it make a difference?" 2020, arXiv 2012.02260.

[15] Y. Duan, Y. Lv, Y. Liu, and F. Wang, "An efficient realization of deep learning for traffic data imputation," *Transportation Research Part C: Emerging Technologies*, vol. 72, pp. 168–181, 2016.

[16] Z. Zhang, M. Li, X. Lin, Y. Wang, and F. He, "Multistep speed prediction on traffic networks: A deep learning approach considering spatio-temporal dependencies," *Transportation Research Part C: Emerging Technologies*, vol. 105, pp. 297–322, 2019.

[17] S. Kataoka, M. Yasuda, C. Furtlehner, and K. Tanaka, "Traffic data reconstruction based on Markov random field modeling," *Inverse Problems*, vol. 30, no. 2, p. 025003, 2014.

[18] J. S. Yedidia, W. T. Freeman, and Y. Weiss, "Generalized belief propagation," *Advances in Neural Information Processing Systems*, pp. 689–695, 2001.

[19] Y. Hara, J. Suzuki, and M. Kuwahara, "Network-wide traffic state estimation using a mixture Gaussian graphical model and graphical lasso," *Transportation Research Part C: Emerging Technologies*, vol. 86, pp. 622–638, 2018.

[20] C. Furtlehner, J. Lasgouttes, and A. de La Fortelle, "A belief propagation approach to traffic prediction using probe vehicles," in *Intelligent Transportation Systems Conference*. IEEE, 2007, pp. 1022–1027.

[21] V. Martin, J. Lasgouttes, and C. Furtlehner, "Latent binary MRF for online reconstruction of large scale systems," *Annals of Mathematics and Artificial Intelligence*, pp. 1–32, 2015.

[22] C. Furtlehner and A. Decelle, "Cycle-based cluster variational method for direct and inverse inference," *Journal of Statistical Physics*, vol. 164, no. 3, pp. 531–574, 2016.

[23] V. Martin, C. Furtlehner, Y. Han, and J. Lasgouttes, "GMRF estimation under topological and spectral constraints," in *ECML PKDD Proceedings,Part II*, 2014, pp. 370–385.

[24] A. Attanasi, L. Meschini, M. Pezzulla, G. Fusco, G. Gentile, and N. Isaenko, "A hybrid method for real-time short-term predictions of traffic flows in urban areas," *2017 5th IEEE International Conference on Models and Technologies for Intelligent Transportation Systems (MT-ITS)*, pp. 878–883, 2017.

[25] J. Pearl, *Probabilistic Reasoning in Intelligent Systems: Network of Plausible Inference*. Morgan Kaufmann, 1988.

[26] D. Bickson, "Gaussian Belief Propagation: Theory and application," Ph.D. dissertation, Hebrew University of Jerusalem, 2008.

[27] Y. Weiss and W. Freeman, "Correctness of belief propagation in Gaussian graphical models of arbitrary topology," *Neural Comput.*, vol. 13, no. 10, pp. 2173–2200, 2001.

[28] D. Malioutov, J. Johnson, and A. Willsky, "Walk-sums and Belief Propagation in Gaussian graphical models," *JMLR*, vol. 7, pp. 2031–2064, 2006.

[29] O. Banerjee, L. El Ghaoui, and A. d'Aspremont, "Model selection through sparse maximum likelihood estimation for multivariate Gaussian or binary data," *JMLR*, vol. 9, pp. 485–516, 2008.

[30] J. Friedman, T. Hastie, and R. Tibshirani, "Sparse inverse covariance estimation with the graphical lasso," *Biostatistics*, vol. 9, no. 3, pp. 432–441, 2008.

[31] K. Scheinberg and I. Rish, "Learning sparse Gaussian Markov networks using a greedy coordinate ascent approach," in *ECML-PKDD*, 2010.

[32] T. Speed and H. Kiiveri, "Gaussian Markov distributions over finite graphs," *The Annals of Statistics*, vol. 14, no. 1, pp. 138–150, 1986.

[33] C. Hsieh, M. A. Sustik, I. S. Dhillon, and K. Ravikumar, "Sparse inverse covariance matrix estimation using quadratic approximation," in *NIPS*, 2011.

[34] C. Furtlehner, J.-M. Lasgouttes, A. Attanasi, L. Meschini, and M. Pezzulla, "Spatio-temporal probabilistic short-term forecasting on urban networks," INRIA, Research Report RR-9236, 2018.

[35] J. Smith and R. Blewitt, *Traffic Modeling Guidelines Version 3.0*. Transport for London, 2010.

[36] B. J. Frey and D. Dueck, "Clustering by passing messages between data points," *Science*, vol. 315, no. 5814, pp. 972–976, 2007.