Transportation Research Part C 46 (2014) 151–178

Contents lists available at ScienceDirect

Transportation Research Part C

journal homepage: www.elsevier.com/locate/trc

Local online kernel ridge regression for forecasting of urban travel times

James Haworth [a,*], John Shawe-Taylor [b], Tao Cheng [a], Jiaqiu Wang [c]

[a] SpaceTimeLab, Department of Civil, Environmental and Geomatic Engineering, University College London, Gower Street, London WC1E BT, United Kingdom
[b] Centre for Computational Statistics and Machine Learning, University College London, Gower Street, London WC1E BT, United Kingdom
[c] Centre for Advanced Spatial Analysis, University College London, 1st Floor, 90 Tottenham Court Road, London W1T 4TJ, United Kingdom

ARTICLE INFO

Article history:
Received 13 January 2014
Received in revised form 23 May 2014
Accepted 24 May 2014

Keywords:
Forecasting
Travel time
Prediction
Time series
Kernel method
Machine learning

ABSTRACT

Accurate and reliable forecasting of traffic variables is one of the primary functions of Intelligent Transportation Systems. Reliable systems that are able to forecast traffic conditions accurately, multiple time steps into the future, are required for advanced traveller information systems. However, traffic forecasting is a difficult task because of the nonlinear and nonstationary properties of traffic series. Traditional linear models are incapable of modelling such properties, and typically perform poorly, particularly when conditions differ from the norm. Machine learning approaches such as artificial neural networks, nonparametric regression and kernel methods (KMs) have often been shown to outperform linear models in the literature. A bottleneck of the latter approach is that the information pertaining to all previous traffic states must be contained within the kernel, but the computational complexity of KMs usually scales cubically with the number of data points in the kernel. In this paper, a novel kernel-based machine learning (ML) algorithm is developed, namely the local online kernel ridge regression (LOKRR) model. Exploiting the observation that traffic data exhibits strong cyclic patterns characterised by rush hour traffic, LOKRR makes use of local kernels with varying parameters that are defined around each time point. This approach has 3 advantages over the standard single kernel approach: (1) It allows parameters to vary by time of day, capturing the time varying distribution of traffic data; (2) It allows smaller kernels to be defined that contain only the relevant traffic patterns, and; (3) It is online, allowing new traffic data to be incorporated as it arrives. The model is applied to the forecasting of travel times on London's road network, and is found to outperform three benchmark models in forecasting up to 1 h ahead.

1. Introduction

The short term forecasting of traffic variables such as speeds, flows and densities is one of the primary goals of Intelligent Transportation Systems (ITSs), with applications in dynamic signal control, advanced traffic management systems (ATMSs) and advanced traveller information systems (ATISs) (Vlahogianni et al., 2004). To date, a wide range of methods have been

* Corresponding author. Tel.: +44 7816076958.
E-mail addresses: j.haworth@ucl.ac.uk (J. Haworth), j.shawe-taylor@ucl.ac.uk (J. Shawe-Taylor), tao.cheng@ucl.ac.uk (T. Cheng), w.jiaqiu@ucl.ac.uk (J. Wang).

http://dx.doi.org/10.1016/j.trc.2014.05.015
0968-090X/

used for short term traffic forecasting, which can be broadly separated into two categories: (1) parametric methods, and; (2) machine learning (ML) methods.[1] The former type includes statistical (space) time series methods such as the (space–time) auto-regressive integrated moving average (ST)ARIMA model family (Billings and Yang, 2006; Cheng et al., 2010; Kamarianakis and Prastacos, 2005; Williams and Hoel, 2003), state space models (Okutani and Stephanedes, 1984; Stathopoulos and Karlaftis, 2003), and Bayesian networks (Anacleto et al., 2013a; Fei et al., 2011; Sun et al., 2004, 2005, 2006; Zheng et al., 2006). Comprehensive reviews can be found in Vlahogianni et al. (2004) and, more recently Vlahogianni et al. (2014). These methods typically assume that the data being described are stationary. That is, they must have constant mean and variance. If this assumption is not satisfied, then the data must be transformed through differencing, or some other transformation (Kendall and Ord, 1990). It is often found that these assumptions are difficult to satisfy, leading standard parametric methods to perform poorly. This has led to the recent development of local parametric model specifications that attempt to model the local characteristics of traffic data in time and/or space (Ding et al., 2010; Kamarianakis et al., 2012; Min and Wynter, 2011; Min et al., 2009, 2010).

An alternative approach is to model the data directly in a nonlinear machine learning (ML) framework. ML methods typically make minimal explicit assumptions about the data generating process, and instead try to *learn* the characteristics of the data through exposure to examples (Mitchell, 1997). ML methods have often been shown to outperform parametric methods in the literature, although a recent comparison study by Chen et al. (2012) suggests that data preprocessing is just as important as model choice. The most widespread ML method in the short term traffic forecasting literature is the artificial neural network (ANN). ANNs have a long history of successful implementation in traffic forecasting, and readers are directed to Dougherty (1995) for a review of the early work. As the power of computers has increased, researchers have developed increasingly sophisticated ANNs for forecasting traffic variables both on highways (see, e.g., Chen and Grant-Muller, 2001; van Lint et al., 2005; van Hinsbergen et al., 2009; Huang and Sadek, 2009; Li and Rose, 2011) and on urban networks (see, e.g. Ledoux, 1997; Yin et al., 2002; Vlahogianni et al., 2005, 2007). The most successful implementations circumvent the traditional black box problem of neural networks by explicitly incorporating domain knowledge into the model structure. For example, the topological structure of the road network can be explicitly represented in the nodes of the hidden layer of an ANN, making the internal function of the model more transparent (van Lint et al., 2005; van Lint, 2006). In a similar vein, Vlahogianni et al. (2005) frame their genetically optimised modular ANN as a multivariate non-linear time series model, fed with spatially and temporally lagged data. The recent work of (Chan et al., 2013a,b; Chan et al., 2012a,b) has further enhanced the position of ANNs at the forefront of the traffic forecasting literature. Other ML methods that have been applied to traffic forecasting include fuzzy rule based systems (Dimitriou et al., 2008) and hybrid models (Van Der Voort et al., 1996; Hofleitner et al., 2012). Karlaftis and Vlahogianni (2011) summarise the main differences (and similarities) between ML methods and parametric methods.

Recent developments have seen the application of kernel methods (KMs) to traffic forecasting. The term *kernel method* is an umbrella term for a broad set of techniques that share a common characteristic. They comprise two components: (1) a *function* that maps the input data into a high (possibly infinite) dimensional space, known as a feature space, and; (2) a *learning algorithm* capable of discovering linear patterns in that space (Shawe-Taylor and Cristianini, 2004). Mapping to the feature space is accomplished efficiently using a *kernel function*, hence the term KM. Because linear relations are sought in the feature space, a broad range of theoretically well founded and efficient linear algorithms can be used. To date, many linear algorithms have been *kernelised* including ridge regression (Hoerl and Kennard, 1970; Saunders et al., 1998), the generalised portrait (Vapnik and Lerner, 1963; Boser et al., 1992), principal components analysis (Schölkopf et al., 1997) and canonical correlation analysis (Hotelling, 1936; Hardoon et al., 2004) amongst many others. KMs are modular in nature, meaning that any kernel algorithm can be applied using a particular kernel, and vice versa (Shawe-Taylor and Cristianini, 2004). This gives them great flexibility as a tool for solving a wide range of practical problems. KMs are an attractive approach for modelling nonlinear and nonstationary data because they combine the advantages of principled, linear learning algorithms such as ordinary least squares (OLS) with nonlinear solutions.

The most widely used KM in traffic forecasting is Support Vector Regression (SVR). Wu et al. (2004) first used SVR for the forecasting of travel times on Taipei's freeway system. Travel times over three distances for 28 consecutive days are used to train a model to forecast the following 7 days' travel times in a one-step-ahead scenario. The results are compared to a current–time predictor and a historical mean predictor and are found to be superior in all cases. The performance of SVR compares favourably with that of ANNs. Vanajakshi and Rilett (2007, 2004) compare the performance of SVR and ANNs in forecasting travel times on San Antonio's freeway system using a forecast horizon varying from 2 min to 1 h ahead. It is found that SVR performs better than ANNs when the size of the training set is small. Zhang and Xie (2008) used ν-SVR for highway traffic volume forecasting, and found the method to outperform a multi-layer feedforward neural network (MLFNN). Other KMs have been applied to traffic forecasting with similar success. Xie et al. (2010) apply Gaussian processes regression (GPR) to highway traffic volume forecasting, and the results are compared with the ν-SVR model of Zhang and Xie. The results are found to be similar, but the GPR model has the advantage of providing error bounds on the forecasts.

One of the main challenges in applying KMs lies in deciding what information to include in the kernel. In KMs a kernel induced feature space is constructed from a database of historical data patterns to store the relevant information about a

[1] ML methods are often referred to as nonparametric methods (Vlahogianni et al., 2004) or computational intelligence (CI) methods (Karlaftis and Vlahogianni, 2011).

particular problem. This feature space must contain sufficient richness of patterns in order to produce accurate forecasts, while not being so large as to sacrifice computational efficiency. Wang and Shi (2013) designed an SVR model in which the input space is defined using chaos theory and wavelet theory, namely the Chaos-Wavelet Analysis-Vector Machine (C-WSVM). It is demonstrated that selection of the correct input space to an SVR model can improve forecasting performance. However, the number of data points required in the kernel is not considered, and time varying input spaces are not investigated. In long traffic series, it is not feasible to use the entire historical dataset to forecast at each point in time because this would involve the construction of very large kernels. Therefore, typically one seeks to select a subset of the data that will be most informative. The most common approach taken in the literature is to use a small subset of the most recent observations, which makes the unrealistic assumption that a sufficiently diverse range of traffic states have been observed recently. For example, Wu et al. (2004) use just 5 weeks of data to train their SVR model, while (Hong, 2010, 2011, 2012; Hong et al., 2011) use just 1 month of data. While the ability to produce strong performance on smaller training data sets is one of the advantages of SVR (Vanajakshi and Rilett, 2004), the use of such a short period of training data means that there is unlikely to be sufficient richness of unusual traffic patterns. Furthermore, as time passes the training data will lose its relevance due to the yearly seasonal trends in traffic data.

One way to address the latter problem is to use online or sequential training in KMs. For example, Castro-Neto et al. (2009) implemented an online SVR algorithm for short term traffic flow forecasting. Instead of retraining the model every time new data become available, which is computationally expensive, the model is iteratively updated three samples at a time and the solution support vectors are changed accordingly. The advantage of this approach is that new information is incorporated into the existing structure of the solution. Empirical results show that the method outperforms Holt's exponential smoothing, and multi-layer perceptron (MLP) ANNs. Gaussian Maximum Likelihood (GML) produces better results under typical conditions but the OL-SVR model performs well under non-recurrent conditions due to the fact it adapts well to new data. Given that non-recurrent events are more difficult to predict this is a significant benefit. However, it does not address the issue that only very recent traffic patterns are included in the kernel, in this case 15 days.

It is well known that road traffic exhibits strong cyclic patterns, usually characterised by a peak period in the morning and evening, with intervening periods of lower traffic. This phenomenon is described statistically as seasonal temporal autocorrelation. Moreover, traffic conditions in the peak periods are more variable than those in the intervening periods, a phenomenon known as heteroskedasticity (Fosgerau and Fukuda, 2012). From the perspective of pattern analysis, this temporal locality implies that data pertaining to the period between peaks will be largely ineffective in forecasting the peak periods, and vice versa, motivating the development of models that produce forecasts based on similar traffic states. The simplest way to achieve this is to arbitrarily divide the data into time periods, such as AM peak and PM peak, and construct a separate model for each (e.g. Hong et al., 2011; Hong, 2012; Stathopoulos and Karlaftis, 2003). A more considered approach is to attempt to identify traffic states based on historical data and construct a model for each one (Kamarianakis et al., 2010, 2012; Min and Wynter, 2011). In operation, the models are switched according to the current traffic state. One can also make the assumption of smooth transition of traffic states, and construct a model in which the parameters vary smoothly by time of day (Anacleto et al., 2013a, 2013b). The latter model type is designed to model the seasonal autocorrelation and heteroskedasticity in traffic data.

In this study, a novel local kernel based algorithm is developed for the forecasting of traffic series, namely local online kernel ridge regression (LOKRR). LOKRR has its roots in OLS, making it relatively simple to interpret and implement compared with other KMs. The structure of the paper is as follows: In Section 2, the LOKRR algorithm is described, beginning with a motivation for the model. In Section 3, a case study is introduced, in which LOKRR is used for forecasting Unit Travel Times (UTT, seconds/metre) collected using automatic number plate recognition (ANPR) on London's road network. In Section 4, the results are presented and analysed. Finally, in Section 5, some conclusions and directions for future research are given.

2. Local online kernel ridge regression

2.1. Motivation of the model

To motivate the model in the current context, the example of forecasting travel times on a single road section (link) is used. Assume a stream of observations of a traffic variable such as flow, density or travel time $[z_1, z_2, \ldots, z_T$ are observed at times $t = 1, 2, \ldots, T$, obtained at an interval of $\tau = 5$ min over a number of days. The task is to make use of observations of the process up to time t to forecast the value of z_{t+1}. Each day, $(60/5) * 24 = 288$ observations are made. Now assume that because of computational constraints the maximum kernel size of a kernel based model is set to $10{,}000 * 10{,}000$, which is a reasonable upper limit. The standard approach is to use the most recent observations of the process to construct a single kernel. Using this approach entails that a maximum of $10{,}000/288 = 34.7$ days of data could be included in the kernel. This approach has two clear drawbacks. Firstly, 34.7 days is a relatively short time in this context, and it is unlikely that sufficient variation in traffic patterns will have been observed over this period in order to produce reliable forecasts, particularly under abnormal conditions. Secondly, because of the strong cyclic pattern present in traffic data, historical traffic patterns recorded at temporal lags that are either close to zero or close to divisible by 288 are likely to be much more informative than those at other lags. For example, when forecasting the level of flow at 9 AM on a Tuesday morning, historical data pertaining to

midnight on a Saturday is unlikely to be useful. Furthermore, the data with which the model is trained is important: If the model is trained on data from August and applied for forecasting traffic in December, it is likely to encounter problems due to the differences in traffic between August and December.

Consider as an alternative, the extreme example where z_{t+1} is forecast as a function of only those travel times that were observed at the same time of day on previous days. With the same maximum kernel size, one could build a kernel containing the travel time patterns obtained on the previous 10,000 days. Of course, in practice, one would not have access to 10,000 days (27.4 years) of historical data, and the nonstationarity in long term travel patterns would render data collected beyond a certain time threshold irrelevant. However, this example serves to highlight the fact that if knowledge of the cyclic nature of traffic is directly incorporated into the model structure, then there is potential for having access to a much richer source of information about the time point to be forecast without increasing the kernel size. In fact, it is possible to include much more informative data in a far smaller kernel. Using the same example, one could capture the behaviour of the link at a single time point over the course of a year in a kernel of size $365 * 365$.

The advantage of the formulation outlined above is that it directly incorporates seasonality into the model structure. However, it has the drawback that it may exclude a significant amount of data that may be of interest. Returning to the example of forecasting some traffic variable on a road link, if one were to commute to work on the same road at approximately the same time each day, one may observe that the road tends to become congested at approximately the same time each day, and may be able to make statements such as "if I leave after 9 am there is always too much traffic", or "if I set off before 8 am my journey is usually pretty quick". However, there is usually significant variation around such trends. For instance, on some days a link may become congested earlier or later than usual; or the congestion may be slightly more or less severe, and one might find oneself making a statement such as "It's especially busy today, something must have happened", or "wow, it's really quiet, it's usually really busy by now". These intuitive observations summarise the variability inherent in traffic data, and a successful traffic forecasting model should be capable of modelling this variability.

Fig. 1 shows an illustration of the variability in Unit Travel Times (UTT, seconds/km) on a road link in London, UK. There are two recurrent peaks on this link, a small one in the morning and a larger one in the evening. Although they occur at roughly the same time each day, the time at which they begin and end, as well as their magnitude, differs within a time window. Within this window, the observed historical patterns are likely to be informative and should be included in the kernel. From the perspective of kernel based methods, this means that there is a temporal window extending in both directions around each point within which past information is informative and a kernel can be constructed. Based on this observation, a local kernel based model of traffic data is proposed, in which local kernels are constructed around each time point. This model is described in the following subsections.

2.2. Ridge regression

The LOKRR model is based on a simple form of regularised linear regression, called ridge regression (RR) (Hoerl and Kennard, 1970). Beginning with the well-known case of multiple linear regression, one seeks to solve a system of equations of the following form:

$$\mathbf{y} = \mathbf{X}\mathbf{w} + \boldsymbol{\varepsilon} \tag{1}$$

where $\mathbf{X}$ is $(n * p)$ and of rank p, n is the number of observations, p is the number of variables, $\mathbf{w}$ is $(p * 1)$ and unknown , $E[\boldsymbol{\varepsilon}] = 0$, and $E[\boldsymbol{\varepsilon}\boldsymbol{\varepsilon}'] = \sigma^2\mathbf{I}_n$. The solution of this system for $\mathbf{w}$ is:

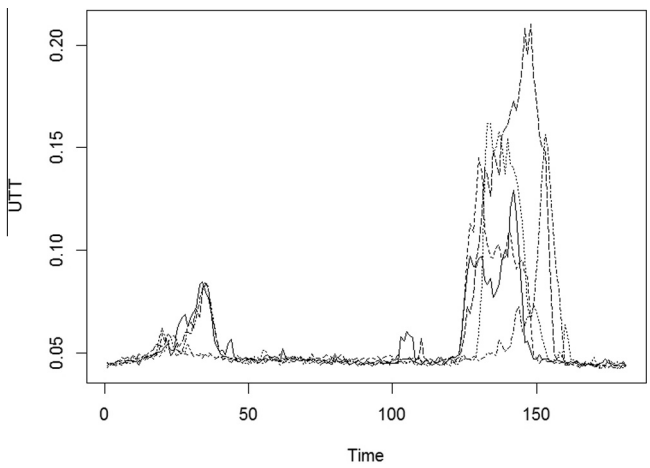

Fig. 1. Illustration of variability in traffic data: each line in the plot is the Unit Travel Time (UTT) profile recorded on a single link (link 1815) on a different day (5 days total).

$$\mathbf{w} = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{y} \tag{2}$$

This solution is viable when $\mathbf{X}'\mathbf{X}$ is nearly a unit matrix. However, if this is not the case then this solution is sensitive to the number of errors in the data. To overcome this problem, a regularisation constant λ can be introduced:

$$\mathbf{w} = (\mathbf{X}'\mathbf{X} + \lambda\mathbf{I}_\mathbf{p})^{-1}\mathbf{X}'\mathbf{y}; \quad \lambda \geq \mathbf{0} \tag{3}$$

λ is called the ridge parameter and the resulting algorithm is known as ridge regression. The matrix $(\mathbf{X}'\mathbf{X} + \lambda\mathbf{I}_\mathbf{p})$ is always invertible if $\lambda > 0$, allowing the solution of ill-posed regression problems. If $\lambda = 0$, Eq. (3) is equivalent to Eq. (2).

2.3. Kernel ridge regression

Linear RR can be converted to a nonlinear algorithm. Eq. (3) can be rearranged in terms of $\mathbf{w}$ as follows (Shawe-Taylor and Cristianini, 2004):

$$\mathbf{w} = \lambda^{-1}\mathbf{X}'(\mathbf{y} - \mathbf{X}\mathbf{w}) = \mathbf{X}'\boldsymbol{\alpha} \tag{4}$$

$\mathbf{w}$ can be written as a linear combination of the training data points, $\mathbf{w} = \sum_{i=1}^{n} a_i\mathbf{x}_i$, with $\boldsymbol{\alpha} = \lambda^{-1}(\mathbf{y} - \mathbf{X}\mathbf{w})$. Therefore, $\boldsymbol{\alpha}$ can be computed as:

$$\begin{aligned} &\boldsymbol{\alpha} = \lambda^{-1}(\mathbf{y} - \mathbf{X}\mathbf{w}) \\ &\Rightarrow \lambda\boldsymbol{\alpha} = (\mathbf{y} - \mathbf{X}\mathbf{X}'\boldsymbol{\alpha}) \\ &\Rightarrow (\mathbf{X}\mathbf{X}' + \lambda\mathbf{I}_n)\boldsymbol{\alpha} = \mathbf{y} \\ &\Rightarrow \boldsymbol{\alpha} = (\mathbf{G} + \lambda\mathbf{I}_n)^{-1}\mathbf{y} \end{aligned} \tag{5}$$

where $\mathbf{G} = \mathbf{X}\mathbf{X}'$, and is known as the Gram matrix. It can be seen from Eq. (5) that the solution is now written in terms of the data points and the weight vector $\mathbf{w}$ need not be solved explicitly. This formulation is computationally more efficient than Eq. (3) when $p > n$, which is often the case in machine learning applications. However, the main benefit of expressing the algorithm in this way is that, given a valid kernel function $\mathbf{K}$, $\mathbf{G}$ can be replaced with a kernel matrix $\mathbf{K}$ as follows:

$$\boldsymbol{\alpha} = (\mathbf{K} + \lambda\mathbf{I}_n)^{-1}\mathbf{y} \tag{6}$$

This algorithm is known as kernel ridge regression (KRR). A forecast for a new data point can be computed as:

$$g(\mathbf{x}) = \mathbf{y}'(\mathbf{K} + \lambda\mathbf{I}_n)^{-1}\mathbf{k} \tag{7}$$

where $\mathbf{K}$ is a kernel matrix of inner products between training vectors and $k = k(\mathbf{x}_i, \mathbf{x})$ is a vector of inner products between the test vector $\mathbf{x}_i$ and the training vectors $\mathbf{x}$. KRR is called a regularization network in the ANN literature (Poggio and Girosi, 1990). The method is also strongly related to Kriging (Krige, 1951), which is termed GPR in the machine learning community (Rasmussen and Williams, 2006). There are many kernels that one could use, but the most widely used and generally applicable kernel is the Gaussian radial basis function (RBF) kernel (Keerthi and Lin, 2003):

$$\mathbf{K}(\mathbf{x}, \mathbf{z}) = \exp\left(-\frac{\|\mathbf{x} - \mathbf{z}\|^2}{2\sigma^2}\right) \tag{8}$$

where σ is the kernel bandwidth, and controls the smoothness of the function. In the following description of the methodology, use of the Gaussian RBF kernel is assumed, and the parameter selection described in Section 2.6 is specific to this kernel type.

2.4. Online kernel ridge regression

To take into account the long term seasonality and trend in traffic data, the LOKRR model described here is online. Online model training involves processing the data one example at a time, making a forecast, and then updating the model based on the predictive error (Shawe-Taylor and Cristianini, 2004). Online learning allows new data to be incorporated into the model as they arrive, and old data to be removed, enabling the model to adapt to changes in the distribution of the data.

LOKRR is based on the sliding window kernel recursive least squares (KRLS) algorithm of Van Vaerenbergh et al. (2006), which is equivalent to an online KRR (OKRR) algorithm. Given a stream of training data points $\{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots\}$, the training data at time t is constructed as $\mathbf{y}_t = [y_t, y_{t-1}, \ldots, y_{t-N+1}]'$ and $\mathbf{X}_t = [\mathbf{x}_t, \mathbf{x}_{t-1}, \ldots, \mathbf{x}_{t-N+1}]'$, where y_t and $\mathbf{x}_t$ are the values of the dependent and independent variables at time t, respectively, N is the size of the window and $t = 1, 2, \ldots, N$. In the context of traffic forecasting, an autoregressive model form is usually specified: Given a series of observations of a traffic variable $z_1, z_2, \ldots, z_T$, $y_t = z_{t+1}$ and $\mathbf{x}_t = [z_t, z_{t-1}, \ldots, z_{t-m+1}]$, where m is the embedding dimension of the series. However, it is possible to include other variables such as a time varying mean, observations of another traffic variable, or other data such as precipitation or incidents

At time t, a regularised kernel matrix $\mathbf{K}_t^\lambda = (\mathbf{K}_t + \lambda\mathbf{I}_t)$ is constructed from $\mathbf{X}_t$. At the following time step, the data window slides along by one point to time $t + 1$ and the most recent observation is added to the training data and the oldest training

point is removed. A new matrix $\mathbf{K}_{t+1}^{\lambda}$ is constructed. Thus, at each time point the kernel matrix is constructed from only the previous N training examples. The main computational burden of standard KRR lies in calculating the inverse of the regularised kernel matrix in Eq. (7). OKRR requires the inverse to be recalculated at each time step, which is infeasible in an online setting. Fortunately, methods exist for updating $\mathbf{K}_t^{\lambda-1}$ without computing the inverse from scratch. To remove a row and column, a kernel matrix $\mathbf{K}$ and its inverse $\mathbf{K}^{-1}$ can be partitioned as follows (Van Vaerenbergh et al., 2006, p. 8):

$$\mathbf{K} = \begin{bmatrix} a & \mathbf{b}' \\ \mathbf{b} & \mathbf{D} \end{bmatrix}, \quad \mathbf{K}^{-1} = \begin{bmatrix} e & \mathbf{f}' \\ \mathbf{f} & \mathbf{G} \end{bmatrix} \tag{9}$$

The inverse of the submatrix $\mathbf{D}$ is required, which can be calculated from the submatrices of $\mathbf{K}^{-1}$ as follows:

$$\mathbf{D}^{-1} = \mathbf{G} - \mathbf{f}\mathbf{f}'/e \tag{10}$$

Therefore, the updated inverse is calculated from the known elements of $\mathbf{K}^{-1}$. To add a row and column to the inverse, $\mathbf{K}^{-1}$ and $\mathbf{K}$ are partitioned as follows:

$$\mathbf{K} = \begin{bmatrix} \mathbf{A} & \mathbf{b} \\ \mathbf{b}' & d \end{bmatrix}, \quad \mathbf{K}^{-1} = \begin{bmatrix} \mathbf{E} & \mathbf{f} \\ \mathbf{f}' & g \end{bmatrix} \tag{11}$$

Then one can take advantage of the following formula:

$$\mathbf{K}^{-1} = \begin{bmatrix} \mathbf{A}^{-1}(\mathbf{I} + \mathbf{b}\mathbf{b}'\mathbf{A}^{-1H}g) & -\mathbf{A}^{-1}\mathbf{b}g \\ -(\mathbf{A}^{-1}\mathbf{b})'g & g \end{bmatrix} \tag{12}$$

where

$$g = (d - \mathbf{b}'\mathbf{A}^{-1}\mathbf{b})^{-1} \tag{13}$$

Here, the superscript H of $\mathbf{A}^H$ denotes the conjugate transpose of $\mathbf{A}$. $\mathbf{A}^{-1}$ is the inverse kernel matrix obtained at the previous time step, demonstrating that the inverse kernel matrix does not need to be recalculated from scratch. These steps reduce the computational complexity of inverting the matrix from $O(N^3)$ to $O(N^2)$, where N is the number of data samples in the kernel.

2.5. Local online kernel ridge regression

OKRR has the advantage over standard KRR that it can incorporate new information in the model structure without a significant increase in computation time. However, it has two limitations that limit its application to traffic series. Firstly, a single kernel is constructed using only the data of the N time points immediately preceding time t. This is reasonable in cases where time series exhibit sudden regime changes such as the Wiener system reported in Van Vaerenbergh et al. (2006). However, in seasonal data, this is likely to be insufficient since the most informative data comes not just from the immediately preceding time points, but from the same times on previous days, weeks or months. The second drawback is that it uses a single set of parameters (ridge parameter λ and kernel parameter(s)) to describe the behaviour of the system across all times. This limits the ability of the model to account for heteroskedasticity. To address these two limitations, a local version of OKRR is proposed, namely Local (L)OKRR, in which a separate kernel is defined for each time of day, each with its own set of parameters.

2.5.1. Local temporal kernels

Consider, again, a stream of training data points $\{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots\}$. The goal is to forecast $\mathbf{y}_t$ using a subset of the observed patterns. Assume the dataset has a regular seasonal component with order S. The training dataset is constructed as $\mathbf{y}_{st} = [y_{1,t}, y_{2,t}, \ldots, y_{n,t}]'$ and $\mathbf{X}_{st} = [\mathbf{x}_{1,t}, \mathbf{x}_{2,t}, \ldots, \mathbf{x}_{n,t}]'$, where $s = 1, 2, \ldots, N$ is the index of the seasonal component, for example the day index, and $t = 1, 2, \ldots, S$ is the time of day index. This formulation is shown diagrammatically in Fig. 2.

The rows of the table indicate the successive days in the training dataset. The columns represent the successive time intervals in a day. N is the total number of days in the training dataset. In the example outlined above of forecasting travel times, $S = 288$. Under this formulation, S kernels $\mathbf{K}_1, \mathbf{K}_2, \ldots, \mathbf{K}_S$ are constructed from the columns of Fig. 2. For example, to forecast the travel time at t_2, the first column of training examples is used. Compared with a single kernel model, this would

	t_1	t_2	...	t_s
s_1	$(\mathbf{x}_{11}, y_{11})$	$(\mathbf{x}_{12}, y_{12})$	...	$(\mathbf{x}_{1S}, y_{1S})$
s_2	$(\mathbf{x}_{21}, y_{21})$	$(\mathbf{x}_{22}, y_{22})$	...	$(\mathbf{x}_{2S}, y_{2S})$
⋮	⋮	⋮	⋱	⋮
s_N	$(\mathbf{x}_{N1}, y_{N1})$	$(\mathbf{x}_{N2}, y_{N2})$	...	$(\mathbf{x}_{NS}, y_{NS})$

Fig. 2. Diagram of the training data construction.

appear to be a large number of kernels, however, each of them can have a much smaller dimension than a single kernel model. Furthermore, a smaller kernel needs to be updated at each time step, which is computationally more efficient than updating a single large kernel. Eq. (14) shows a local temporal kernel constructed in this way.

$$\mathbf{K}_{st} = \begin{pmatrix} k(\mathbf{x}_{1,t},\mathbf{x}_{1,t}) & k(\mathbf{x}_{1,t},\mathbf{x}_{2,t}) & \cdots & k(\mathbf{x}_{1,t},\mathbf{x}_{N,t}) \\ k(\mathbf{x}_{2,t},\mathbf{x}_{1,t}) & k(\mathbf{x}_{2,t},\mathbf{x}_{2,t}) & \cdots & k(\mathbf{x}_{2,t},\mathbf{x}_{N,t}) \\ \vdots & \vdots & \ddots & \vdots \\ k(\mathbf{x}_{N,t},\mathbf{x}_{1,t}) & k(\mathbf{x}_{N,t},\mathbf{x}_{2,t}) & \cdots & k(\mathbf{x}_{N,t},\mathbf{x}_{N,t}) \end{pmatrix} \tag{14}$$

2.5.2. Local temporal windows

The model form shown in Fig. 2 restricts the model to include only those patterns that were observed at exactly the same time of day. However, it was argued in Section 2.2 that this is overly restrictive. To address this, a local temporal window is formed around each time t, which is denoted as w and the training data is constructed as:

$$\mathbf{y}_{stw} = [\{y_{1,t-w},\ldots,y_{1,t-1},y_{1,t},y_{1,t+1},\ldots,y_{1,t+w}\},\ldots,\{y_{n,t-w},\ldots,y_{n,t-1},y_{n,t},y_{n,t+1},\ldots,y_{n,t+w}\}]' \tag{15}$$

$$\mathbf{x}_{stw} = [\{\mathbf{x}_{1,t-w},\ldots,\mathbf{x}_{1,t-1},\mathbf{x}_{1,t},\mathbf{x}_{1,t+1},\ldots,\mathbf{x}_{1,t+w}\},\ldots,\{\mathbf{x}_{n,t-w},\ldots,\mathbf{x}_{n,t-1},\mathbf{x}_{n,t},\mathbf{x}_{n,t+1},\ldots,\mathbf{x}_{n,t+w}\}]' \tag{16}$$

Based on this, kernels $\mathbf{K}_1, \mathbf{K}_2, \ldots, \mathbf{K}_S$ are defined according to:

$$\mathbf{K}_t = \begin{pmatrix} \mathbf{K}_{1,1} & \mathbf{K}_{1,2} & \cdots & \mathbf{K}_{1,n} \\ \mathbf{K}_{2,1} & \mathbf{K}_{2,2} & \cdots & \mathbf{K}_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{K}_{n,1} & \mathbf{K}_{n,2} & \cdots & \mathbf{K}_{n,n} \end{pmatrix} \tag{17}$$

where

$$\mathbf{K}_{1,1} = \begin{pmatrix} k(\mathbf{x}_{1,t-w},\mathbf{x}_{1,t-w}) & \cdots & k(\mathbf{x}_{1,t-w},\mathbf{x}_{1,t-1}) & k(\mathbf{x}_{1,t-w},\mathbf{x}_{1,t}) & k(\mathbf{x}_{1,t-w},\mathbf{x}_{1,t+1}) & \cdots & k(\mathbf{x}_{1,t-w},\mathbf{x}_{1,t+w}) \\ \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ k(\mathbf{x}_{1,t-1},\mathbf{x}_{1,t-w}) & \cdots & k(\mathbf{x}_{1,t-1},\mathbf{x}_{1,t-1}) & k(\mathbf{x}_{1,t-1},\mathbf{x}_{1,t}) & k(\mathbf{x}_{1,t-1},\mathbf{x}_{1,t+1}) & \cdots & k(\mathbf{x}_{1,t-1},\mathbf{x}_{1,t+w}) \\ k(\mathbf{x}_{1,t},\mathbf{x}_{1,t-w}) & \cdots & k(\mathbf{x}_{1,t},\mathbf{x}_{1,t-1}) & k(\mathbf{x}_{1,t},\mathbf{x}_{1,t}) & k(\mathbf{x}_{1,t},\mathbf{x}_{1,t+1}) & \cdots & k(\mathbf{x}_{1,t},\mathbf{x}_{1,t+W}) \\ k(\mathbf{x}_{1,t+1},\mathbf{x}_{1,t-w}) & \cdots & k(\mathbf{x}_{1,t+1},\mathbf{x}_{1,t-1}) & k(\mathbf{x}_{1,t+1},\mathbf{x}_{1,t}) & k(\mathbf{x}_{1,t+1},\mathbf{x}_{1,t+1}) & \cdots & k(\mathbf{x}_{1,t+1},\mathbf{x}_{1,t+w}) \\ \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ k(\mathbf{x}_{1,t+w},\mathbf{x}_{1,t-w}) & \cdots & k(\mathbf{x}_{1,t+w},\mathbf{x}_{1,t-1}) & k(\mathbf{x}_{1,t+w},\mathbf{x}_{1,t}) & k(\mathbf{x}_{1,t+w},\mathbf{x}_{1,t+1}) & \cdots & k(\mathbf{x}_{1,t+w},\mathbf{x}_{1,t+w}) \end{pmatrix} \tag{18}$$

Including a window of patterns around t increases the amount of local temporal information available to the model. However, it also increases the kernel size. The kernel defined in Eq. (14) is $n * n$, whereas the kernel defined in Eq. (17) is $(n * ((2 * w) + 1)) * (n * ((2 * w) + 1))$. As w increases, the dimension of $\mathbf{K}$ increases by $2n$. Therefore, it is preferable to keep w small. However, it should be noted that, in an online setting, this formulation is still efficient compared with using a kernel defined on all the data.

2.6. Parameter selection of LOKRR

The formulation outlined above allows for local tuning of the model parameters. Given a model with $S = 288$, 288 separate kernels are defined, each of which has its own set of parameters. The training of these parameters has no added computational cost over the training of a single set of parameters since the model still requires the inversion of a single kernel matrix at each time step. Therefore, each of the 288 models can be trained simultaneously. This enables the model to capture temporal heteroskedasticity by allowing the kernel bandwidth and the ridge parameter to vary by time of day. Parameter selection in kernel based models is an active research area, and various methods, mainly heuristic in nature, have been used to improve the parameter selection process. For example, in SVMs, genetic algorithms (Üstün et al., 2005; Wu et al., 2009; Cai et al., 2009; Li and Yang, 2008), chaotic immune algorithms (Wang et al., 2009a), particle swarm optimisation (Li et al., 2010), immune particle swarm optimisation (Wang et al., 2009b) differential evolution (Lahiri and Ghanta, 2008; Li and Cai, 2008) and ant colony optimisation (Zheng et al., 2008) have been used to select parameters, amongst others. Although this research has been instrumental in improving the speed of training of machine learning algorithms, in an applied setting it is desirable to have a parameter selection process that is interpretable and simple to implement, particularly when the model is to be applied in a large number of cases. In the following subsections, the selection method is described for each of the model parameters in the context of using a Gaussian RBF kernel, which are λ, σ and w. The parameter space is derived from the data, making parameter selection straightforward in any application.

2.6.1. Tuning λ

The regularisation constant λ is a free parameter that needs to be tuned. Usually, λ is determined by cross-validation over a space of values. It is sensible to try to limit this space of values in order to limit the computation time required to train models. Exterkate (2013) showed that appropriate values of λ can be found by relating λ to the signal to noise ratio φ. First the R^2 is obtained from an OLS fit of $\mathbf{y}$ on $\mathbf{X}$. If the OLS fit were the true model, then $\varphi_0 = R^2/(1-R^2)$. From this, λ_0 can be determined for a Gaussian kernel from the signal to noise ratio as $\varphi_0 = 1/\lambda_0$. Exterkate recommends the following grid be used to train λ: $\{\frac{1}{8}\lambda_0, \frac{1}{4}\lambda_0, \frac{1}{2}\lambda_0, \lambda_0, 2\lambda_0\}$. In the study, Monte Carlo simulation was used to compare the mean squared prediction error (MSPE) obtained from this grid with that obtained from the true values of λ, and it was found that estimating λ resulted in an increase in MSPE of only around 0.4%. It follows that the added computational effort associated with training a larger grid of values for λ is not justified. The method of Exterkate is therefore recommended for tuning λ in LOKRR.

2.6.2. Tuning σ

The second free parameter in the LOKRR model is the kernel parameter. In the case of the Gaussian RBF kernel defined in Eq. (8), this is the kernel bandwidth σ, which determines the smoothness of the forecasting function. A large value of σ results in a smooth function while a small value results in a more complex function. There is a wealth of literature on the training of σ in Gaussian kernels, particularly in the context of SVMs, and some of the methods that have been used to date were listed in Section 2.6. However, it is important to recognise that there is a small range of values within which σ can vary, which is based on the variance of the data in a given kernel. Caputo et al. (2002) showed that the optimal values of σ lie between the 0.1 and 0.9 quantiles of $||\mathbf{x}-\mathbf{x}'||^2$ and so can be estimated from the Euclidean distance between the data points in the kernel. This is the scheme used for automatic σ estimation for SVM models in the R package Kernlab (Karatzoglou et al., 2004, 2006). Exterkate (2013) proposed a similar method for selecting σ in the context of KRR.

While it is likely that more sophisticated training schemes will yield slightly more accurate results, the size of the improvement is unlikely to justify the increased computational burden of training extra parameter combinations. Therefore, in this study, the approach of Caputo et al. is used to estimate the median, 0.25 and 0.75 quantiles of each kernel, and these three values of σ are used to train each model. Not only does this reduce the size of the parameter space that needs to be searched, it also provides a principled way of training LOKRR models on other datasets.

2.6.3. Tuning the window size w

The window size w is an important parameter that decides how much local temporal information to include in the model. The larger w gets, the more information is included in the model pertaining to each day of the training data. However, as mentioned in Section 2.5.2, an increase in w of 1 results in an increase in the dimension of the kernel of $2n$. Therefore, it has a significant effect on the computation time required to train and update models. Holding the overall size of the kernel static, an increase in w necessitates a $2w+1$-fold decrease in n. Therefore, a smaller w is desirable in order to maintain computational efficiency. The value of w is determined in the model training process and depends on the application. However, it is recommended to restrict its maximum value. The values tested here are $w = 1, 2, 3$. Because of strong seasonal temporal autocorrelation in the data it is assumed that a maximum window size of 3 is sufficient. In other applications, a strategy of holding the kernel size static, and reducing the training length as w increases is recommended in order to examine the trade-off between w and training data length.

2.6.4. Data normalisation

Data normalisation, also referred to as scaling, is important in kernel methods because it brings all of the independent variables into the same range. This ensures that no single variable dominates over the rest of the variables when the distance between the training vectors is calculated in the kernel. There are various types of normalisation that could be used, including standardization to produce z-scores, scaling by domain (i.e. $[0,1]$) and minmax normalisation (Juszczak et al., 2002). Each kernel is normalised separately. This enables each kernel to capture the distribution of the data around time t, and brings the kernel parameters for each time point into the same range. This also enables the heteroskedasticity in the data to be examined through the model parameters. In the context of traffic data, it would be expected that the variance in σ would be higher during the peak periods than in the inter peak periods.

3. Case study: forecasting travel times with LOKRR

3.1. Data description

The data used in this study are Unit Travel Times (UTTs, seconds/metre) collected on London's road network, as part of the London Congestion Analysis Project (LCAP) coordinated by Transport for London (TfL). LCAP TTs are observed using automatic number plate recognition (ANPR) technology. Cameras operate in pairs, which are called links. The camera at the start of a link is called its start camera, and the camera at its end its end camera. A diagram of an ANPR link is shown in Fig. 3.

Individual vehicle TTs are aggregated at 5 min intervals to produce a regularly spaced time series with 288 observations per day. Due to the comparatively poor quality of data collected during the night time period, only data collected between 6 AM and 9 PM are used in the analysis (180 observations per day). To test the performance of the LOKRR model, the ten

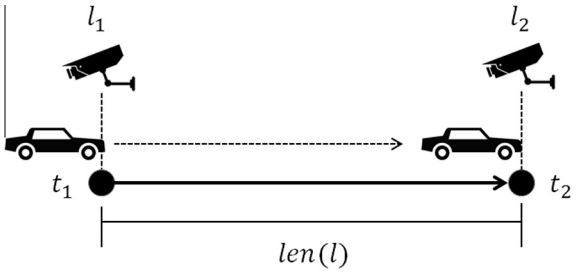

Fig. 3. Observing travel times using ANPR: a vehicle passes camera l_1 at time t_1, and its number plate is read. It then traverses link l and passes camera l_2 at time t_2 and its number plate is read again. The two number plates are matched using inbuilt software, and the TT is calculated as $t_2 - t_1$. Raw TTs are converted to UTTs by dividing by $len(l)$, which is the length of link l.

Table 1
The test links and their patch rates and frequency.

Link ID	24	26	442	454	1815	1799	453	1798	2448	881
% Missing	0.8	0.8	1.15	1.2	1.22	1.3	1.39	1.44	1.44	1.45
Avg. frequency	20.95	15.49	14.09	11.74	68.82	31.69	13.74	28.53	9.37	18.96
Length (m)	366.7	907.5	899.4	710.1	4033.1	1101	892.6	4670.2	1216.5	3744.3

LCAP links with the lowest levels of missing data are selected. These links are shown in Table 1, along with the average data frequency (number of vehicles per observation). The spatial location of the links in the network is shown in Fig. 4. Links with low levels of missing data are selected in order to minimise its effects on the results.

Fig. 5 shows the time series of each of the test links over the course of 10 weeks (weekends included). Evidence of the seasonal component in the data is present in the majority of links. However, there is significant variation from day to day. Higher unit journey times are observed in the AM and PM peak periods to varying degrees in all of the links, but there is clear heterogeneity in the magnitude (height) and the duration of the peaks from day to day. Between links there is also considerable variation in the observed traffic patterns. Links 26, 442, 2448 and 881 appear to have fairly stable traffic patterns. However, other links such as 1815 and 1799 are characterised by few very large peaks.

In total there are 154 days for which data are available, collected between January and July 2011. To test the models, the data are divided into three sets; a training set, a testing set and a validation set, which are 80 days (52%), 37 days (24%) and 37 days (24%) in length, respectively. The testing set runs from Tuesday 28th April to Wednesday 1st June 2011 and the validation set runs from Thursday 2nd June to Friday 8th July 2011. The sizes of the training and testing sets are fixed in this experiment for simple comparison of results with the benchmark models described in Section 3.6.

3.2. Model training

A sliding window validation approach is used to train the models. At the first time step, the training set is used to build a model to forecast the first day of the testing set. Following this, the first day of the training set is removed and replaced with the first day of the testing set. A model is then built using the new training set to forecast the second day of the testing set, and so on and so forth. The selected model is the one that minimises the average training error across the days in the testing set. To measure the training error, the root mean squared error (RMSE) index is used. This is defined as follows:

$$\mathrm{RMSE} = \sqrt{1/n \sum_{t=1}^{n} (y_t - \hat{y}_t)^2} \tag{19}$$

where y_t and $\hat{y}_t$ are the observed and forecast values, respectively.

3.3. Model description

An autoregressive model structure is used to train the models. In an autoregressive model, m previous observations of the series are used to forecast the next value, where m is the embedding dimension. Assume a stream of input/output pairs $\{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots\}$ is to be arranged into a vector of dependent variables $\mathbf{y}_t = [y_t, y_{t-1}, \ldots, y_{t-N+1}]'$ and a matrix of independent variables $\mathbf{X}_t = [\mathbf{x}_t, \mathbf{x}_{t-1}, \ldots, \mathbf{x}_{t-N+1}]'$. In an embedded time series, the independent variable vector at time t is constructed as follows[2]:

[2] An intercept term is also added in the LOKRR model, but is omitted from the description both for clarity and because it is not required for all time series models.

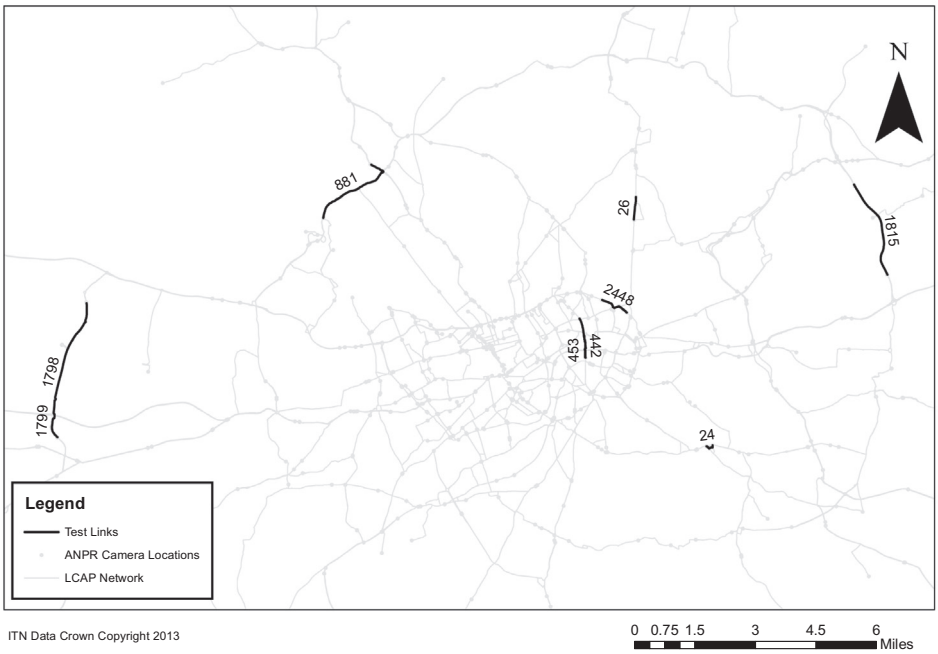

Fig. 4. Location of the test links on the LCAP network.

$$\mathbf{x}_t = [y_t, y_{t-1}, \ldots, y_{t-(m-1)}] \tag{20}$$

In this formulation, the temporal ordering of the data is not made explicit beyond the definition of the window w. Therefore, all data patterns in the kernel are considered equally and a forecast is made based only on the similarity of the data pattern observed at time t with the patterns in the historical data. It has been demonstrated in previous studies that including a time varying average as a variable in the model can improve the predictive performance of pattern based models (Smith and Demetsky, 1996). The sample time varying average $\hat{\mu}_t$ can be calculated as:

$$\hat{\mu}_t = 1 \Big/ n \sum_{i=1}^{n} y_{i,t} \tag{21}$$

where n is the number of days in the training data. The mean is appended to the vector defined in Eq. (20) to give:

$$\mathbf{x}_t = [y_t, y_{t-1}, \ldots, y_{t-(m-1)}, \hat{\mu}_t] \tag{22}$$

Including $\hat{\mu}_t$ in the model introduces the notion of temporal ordering into the feature space defined by the kernel. Essentially, this has the effect of making a prior assumption that those patterns observed at near times to the current pattern are likely to be more informative than those observed at different times. At longer forecasting horizons and larger values of w, the effect of including the average variable is greater.

3.4. Forecasting scenarios

The performance of the LOKRR model is assessed in terms of its ability to forecast travel times at four forecast horizons, which are 15, 30, 45 and 60 min ahead. The data are not aggregated to achieve these forecasts because this reduces the temporal granularity of the data. Instead, forecasts are made of 5 min aggregated UTTs, at 3, 6, 9 and 12 steps into the future.

There are two strategies that can be used to make this type of forecast, which are iterated and direct. In the first strategy, a one step ahead forecast is made and the forecast point is fed back into the model and used to forecast the subsequent time step. This process is iteratively repeated until the desired number of time steps have been forecast. In the second strategy, forecasts are made directly. This is achieved by sampling the series at the desired temporal interval, and using this series to construct a model for each time interval. For example, given 5 min averaged data, every second point is used to forecast at the 10 min interval and every third point to forecast at the 15 min interval. In this study, the second strategy is used as preliminary testing revealed that direct forecasting was more effective than iterated forecasting, both with the LOKRR model and the comparison models described in Section 3.6. Given an embedding dimension m and a time delay τ, the embedded, delayed series $\mathbf{x}_t(\tau)$ is defined as:

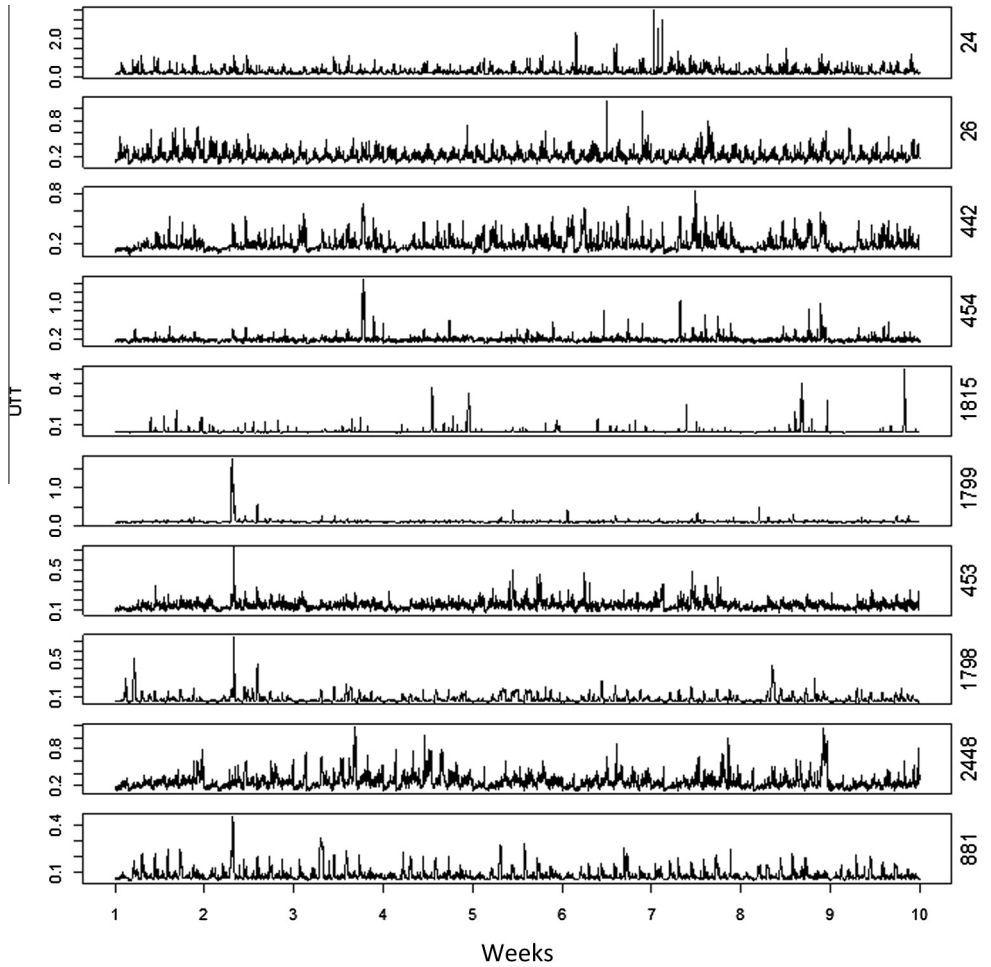

Fig. 5. Time series of each of the test links over the first 10 weeks of the training period.

$$\mathbf{x}_t(\tau) = [y_t, y_{t-\tau}, y_{t-2\tau}, \ldots, y_{t-(m-1)\tau}, \hat{\mu}_t] \tag{23}$$

A separate delayed, embedded series is constructed for each of the forecast intervals.

3.5. Performance measures

Along with RMSE, three additional performance measures are used. The NRMSE is defined as:

$$\text{NRMSE} = \frac{\text{RMSE}}{y_{\max} - y_{\min}} \tag{24}$$

where $y_{\max}$ and $y_{\min}$ are the maximum and minimum values of the series, respectively. NRMSE is scale free, allowing comparison of performance between links. A feature of the RMSE is that it varies with the variability within the distribution of error magnitudes, the square root of the number of errors, and the average-error magnitude. It has been argued that an absolute error measure is more appropriate when comparing the performance of models (Willmott and Matsuura, 2005). In this case, the mean absolute percentage error (MAPE) is used as percentages are easy to interpret conceptually. The MAPE is defined as:

$$\text{MAPE} = 1 \Big/ n \sum_{t=1}^{n} \left| \frac{y_t - \hat{y}_t}{y_t} \right| \tag{25}$$

The mean absolute scaled error (MASE) is a generally applicable measurement of forecast accuracy (Hyndman and Koehler, 2006). The MASE compares the absolute errors in the forecast values with the absolute errors of a naïve forecast, where the previous value of the series is used as the forecast of the next value: $\hat{y}_t = y_{t-1}$. As the naïve model is the simplest possible model, any more complicated time series model should outperform it as a minimum requirement. The MASE is defined as:

$$\mathrm{MASE} = 1 \Big/ n\sum_{t=1}^{n}\left(\frac{|e_t|}{\frac{1}{n-1}\sum_{i=2}^{n}|y_i - y_{i-1}|}\right) \tag{26}$$

where e_t is the forecast error for a given period and the denominator is the average forecast error of the naïve forecast method. A MASE value less than 1 indicates that the forecast model outperforms the naïve model, while a value greater than 1 indicates the opposite.

3.6. Benchmark models

3.6.1. ARIMA

The ARIMA model is a linear statistical time series model that has been widely used in many time series forecasting applications, including traffic forecasting. ARIMA is well described in the literature, and readers are directed to the texts of Box et al. (1994) and Hamilton (1994) for detailed treatments. ARIMA is commonly used as a benchmark model against which to judge the performance of time series forecasting models. An ARIMA(p,d,q) model is described by its autoregressive order p, its moving average order q, and the number of differences d. In time series with cyclical trends, a seasonal (S)ARIMA$(p,d,q) * (P,D,Q)$ model is often used, where P is the seasonal autoregressive order, D is the order of the seasonal difference, and Q is the seasonal moving average order. To maintain consistency with the LOKRR approach, the multi-step forecasts are made in a single step, rather than recursively.[3] As the model building procedure of ARIMA requires a continuous time series, τ embedded, delayed series are constructed and concatenated to form a single series on which the model is trained. For example, consider a time series with its origin at $t = 0$. If $\tau = 3$, three separate series are constructed, one beginning at $t = 0$, one at $t = 1$ and one at $t = 2$, which are merged into a single series, under the reasonable assumption that each series will have the same properties.

The function *auto.arima* in the R package *forecast* is used to train the model (Hyndman and Khandakar, 2007). The Akaike Information Criterion (AIC) is used to determine the parameters of the ARIMA models, which is a common alternative to the Box–Jenkins procedure (Ozaki, 1977). Nonseasonal and seasonal model specifications are considered in the model building procedure.

3.6.2. Support vector regression

SVR is a kernel based machine learning method that has demonstrated strong performance in forecasting of traffic variables in recent years. An excellent introduction to SVR is given in (Smola et al., 2004). SVR is similar to KRR, but has the advantage of maintaining a sparse representation of the solution. The ε-insensitive variant of SVR (ε-SVR) is used here. There are a minimum of three parameters to train in an ε-SVR model, which are:

(1) The kernel parameter(s).
(2) The regularisation constant C (similar to λ in KRR).
(3) The width of the tube ε.

In this case, to maintain consistency, the same RBF kernel is used as shown in Eq. (8) and the kernel parameter σ is determined using the same method as the LOKRR model. The value of C is varied in the range $C = \{0.1, 1, 10, 100\}$. ε is varied in the range $\varepsilon = \{0.0001, 0.001, 0.01, 0.1\}$. After initial feasibility testing, the kernel size is limited to 70 days to ensure training times are acceptable. This means a kernel of size $180 * 70 = 12{,}600$ is created for each link. SVR is implemented using the *kernlab* package in R (Karatzoglou et al., 2004).

3.6.3. Artificial neural network

The ANN is the most widely applied traffic forecasting method in the literature. In time series forecasting, ANN models with recurrent architectures have displayed the best results. The Jordan (1986) and Elman (1990) networks are two examples of these. Here, the Elman network is selected as the comparison model as a preliminary analysis revealed superior performance over the Jordan network on this dataset. It must be noted, however, that different ANN structures may yield different results to the Elman network presented here. There is one parameter to be determined in the Elman network, which is the number of input layers. Here, the number is varied between 1 and 10. To train the Elman network, the *RSNNS* library in R is used, which is based on the Stuttgart Neural Network Simulator (Bergmeir and Benítez, 2012).

4. Results

4.1. Training phase

Table 2 shows the training errors for each of the links at each of the forecast intervals. RMSE is the error criterion used for model selection. The NRMSE, MAPE and MASE are shown to aid interpretability of the results. The results in terms of NRMSE

[3] This approach was found to produce markedly superior results to the iterated approach in preliminary testing.

Table 2
Training errors at: (a) 15 and 30 min forecast horizons and (b) 45 and 60 min horizons.

(a)	15 min				30 min			
Link ID	RMSE	NRMSE	MAPE	MASE	RMSE	NRMSE	MAPE	MASE
24	0.1081	0.0964	24.91	0.9351	0.1254	0.1119	29.42	0.8869
26	0.0477	0.0690	15.90	0.8302	0.0505	0.0731	16.78	0.7617
442	0.0553	0.0539	13.69	0.9218	0.0665	0.0647	15.93	0.8467
453	0.0385	0.0622	13.41	0.8983	0.0413	0.0668	14.34	0.8304
454	0.1051	0.0366	16.62	0.9311	0.1259	0.0439	19.48	0.8171
881	0.0189	0.0307	11.27	0.8798	0.0222	0.0360	13.16	0.8286
1798	0.0209	0.0350	9.38	0.8412	0.0304	0.0508	13.94	0.7861
1799	0.0257	0.0360	8.58	0.8453	0.0347	0.0487	9.91	0.7602
1815	0.0183	0.0219	4.71	0.9510	0.0334	0.0399	8.35	1.0853
2448	0.0839	0.0607	16.86	0.9334	0.1038	0.0751	20.76	0.8874
(b)	45 min				60 min			
Link ID	RMSE	NRMSE	MAPE	MASE	RMSE	NRMSE	MAPE	MASE
24	0.1308	0.1167	30.78	0.8343	0.1343	0.1198	32.14	0.8007
26	0.0516	0.0747	17.16	0.7226	0.0526	0.0761	17.62	0.6937
442	0.0745	0.0725	17.82	0.8315	0.0811	0.0790	18.94	0.7990
453	0.0420	0.0679	15.13	0.7883	0.0484	0.0524	16.01	0.7721
454	0.1379	0.0481	20.75	0.7417	0.1461	0.0509	21.17	0.7023
881	0.0260	0.0422	14.38	0.7743	0.0276	0.0448	15.03	0.7479
1798	0.0349	0.0584	17.16	0.7451	0.0388	0.0650	19.56	0.6938
1799	0.0401	0.0563	10.60	0.6950	0.0421	0.0592	11.04	0.6528
1815	0.0389	0.0464	10.62	1.0491	0.0437	0.0522	12.38	1.0305
2448	0.1162	0.0841	23.66	0.8637	0.1241	0.0897	25.28	0.8319

Table 3
Fitted model parameters at each of the forecast horizons.

Link	15 min			30 min			45 min			60 min		
	σ	λ	w	σ	λ	w	σ	λ	w	σ	λ	w
24	0.75	1/8	3	0.75	1/8	3	0.5	1/8	1	0.75	1/8	2
26	0.5	1/8	2	0.5	1/8	2	0.5	1/8	2	0.5	1/8	3
442	0.5	1/8	3	0.5	1/8	3	0.75	1/8	3	0.75	1/8	3
453	0.75	1/8	3	0.75	1/8	3	0.75	1/8	3	0.75	1/8	3
454	0.5	1/8	3	0.75	1/8	2	0.75	1/8	3	0.75	1/8	3
881	0.5	1/8	3	0.5	1/8	3	0.75	1/8	3	0.75	1/8	3
1798	0.5	1/8	1	0.5	1/8	2	0.75	1/8	3	0.75	1/8	3
1799	0.5	1/8	3	0.75	1/8	3	0.75	1/8	3	0.75	1/8	3
1815	0.5	1/8	3	0.5	1/8	2	0.5	1/8	2	0.5	1/8	2
2448	0.25	1/8	2	0.5	1/8	2	0.75	1/8	3	0.75	1/8	3

Note: σ values are quantiles of $||\mathbf{x} - \mathbf{x}'||^2$; λ values are multiples of λ_0.

and MAPE reveal considerable variation in forecast accuracy between links. For instance, at the 15 min interval, the best MAPE for link 24 is 24.91, whereas the best MAPE for link 1815 is 4.7. The errors increase as the forecasting interval increases, which is to be expected as there is less local temporal information available at longer forecast intervals. However, even at the 60 min interval, most of the links exhibit MAPE values of between 10 and 20.

Examining the performance in terms of MASE, it can be seen that all of the models meet the minimum requirement of outperforming the naïve model at each of the forecast horizons, with the exception of link 1815. In general, the performance of LOKRR in terms of MASE improves as the forecasting horizon increases. This implies that the model is successful in capturing the cyclic variation in the data, which is not captured by the naïve model at increasing forecast horizons.

4.2. Testing phase

The parameters obtained in the training phase are used to forecast the remaining 37 days in the validation set. Table 4 shows the testing results. In terms of RMSE, the training and testing errors are broadly similar. At the 15, 30 and 60 min intervals, testing performance is better than training performance on 4 of the 10 links. At the 45 min interval, testing performance is better than training performance on 5 of the links. The reason for the differences in the training and testing errors may reflect the relative forecastability of the data in each period. The MASE gives a good indication of performance in this regard. For each of the links, the MASE is <1, with the exception of link 1815. The pattern of decreasing MAPE with increasing forecast interval observed in the training phase is repeated in the testing phase. These results indicate that the model generalises well to the unseen validation data.

Table 4
Testing errors of the LOKRR model.

	RMSE	NRMSE	MAPE	MASE
15 min				
1798	0.015056	0.061842	9.896842	0.86972
1799	0.020865	0.041542	9.04533	0.86287
1815	0.021719	0.028766	5.557989	1.066146
2448	0.078816	0.063965	17.51586	0.951652
24	0.135792	0.050655	23.8859	0.887596
26	0.056001	0.050071	16.67646	0.838566
442	0.077271	0.026585	16.10907	0.900641
453	0.05	0.024215	12.96563	0.842421
454	0.083559	0.052184	16.19808	0.906937
881	0.027327	0.030507	12.72943	0.872443
30 min				
1798	0.021784	0.089477	14.50633	0.778719
1799	0.025402	0.050759	10.22777	0.789565
1815	0.0303	0.040132	8.325483	1.079486
2448	0.104256	0.084611	21.61859	0.897533
24	0.155577	0.058035	28.45854	0.843025
26	0.062764	0.056117	17.64926	0.756886
442	0.095284	0.032783	19.62794	0.832451
453	0.053543	0.025931	14.08368	0.77528
454	0.112563	0.070298	20.42523	0.841052
881	0.036773	0.041052	14.75696	0.817719
45 min				
1798	0.02598	0.111417	18.02658	0.72652
1799	0.026217	0.052198	11.11021	0.742244
1815	0.035621	0.04718	11.29372	1.116573
2448	0.117528	0.095382	24.21025	0.869196
24	0.168476	0.062847	31.1086	0.825435
26	0.066129	0.059126	18.19328	0.70734
442	0.09972	0.034309	21.12433	0.751533
453	0.054154	0.026227	14.58123	0.742946
454	0.124311	0.077635	23.43005	0.795166
881	0.042819	0.047802	15.85057	0.756781
60 min				
1798	0.028534	0.117203	20.60327	0.671616
1799	0.025176	0.062291	11.66314	0.735095
1815	0.037106	0.049147	12.53772	1.061102
2448	0.124704	0.101206	25.82053	0.833048
24	0.17265	0.064404	32.58387	0.781458
26	0.066611	0.059556	18.4175	0.6703
442	0.103471	0.0356	22.76551	0.720822
453	0.055945	0.027094	15.38157	0.738591
454	0.130278	0.081362	25.11916	0.768823
881	0.046543	0.051959	17.09304	0.721626

4.2.1. Comparison with benchmark models

Table 5 shows the average errors of the LOKRR model and the comparison models in the training and testing periods at each of the forecast intervals. In terms of RMSE, the LOKRR model is the best performing model in both the training and testing phases at each forecast horizon.

In terms of MAPE, LOKRR outperforms each of the comparison models at the 15 min, 30 min and 45 min horizons. The SVR model performs marginally better at the 60 min horizon in terms of average MAPE. Of the comparison models, ARIMA performs best at shorter forecast horizons, but its performance decreases rapidly as the forecast horizon increases. The ANN and SVR models outperform ARIMA at the 45 min and 60 min horizons.

Fig. 6 shows the RMSE of each of the models at the four forecasting horizons. LOKRR is the best performing model in terms of RMSE on 5, 9, 9 and 10 of the links at the 15, 30, 45 and 60 min forecasting horizons respectively, highlighting the consistency in the performance of the model. The ARIMA model exhibits the best performance on 4, 1 and 1 of the 10 links at the 15, 30 and 45 min forecast horizons respectively. The SVR and ANN models exhibit best performance on a single model each.

Fig. 7 shows the MAPE of each of the models at the four forecasting horizons. In terms of MAPE, LOKRR is the best performing model on 7, 6, 6 and 5 of the links at the 15, 30, 45 and 60 min horizons respectively. SVR also performs strongly, achieving the lowest MAPE on 3, 3, 1 and 3 of the links at the 15, 30, 45 and 60 min horizons respectively. ANN is the best performing model on 1, 3 and 2 of the links at the 30, 45 and 60 min intervals respectively. The ARIMA model does not perform well in terms of MAPE, and is not the best performing model in any of the cases.

Table 5
Comparison with benchmark models – average errors at: (a) 15 min; (b) 30 min; (c) 45 min and (d) 60 min forecast horizons.

Model	Training			Testing		
	RMSE	NRMSE	MAPE	RMSE	NRMSE	MAPE
(a)						
LOKRR	0.051578	0.049348	13.24606	0.056641	0.043033	14.05806
SVR	0.063477	0.060735	13.35697	0.067139	0.050494	14.16037
ANN	0.062049	0.059943	14.58549	0.062892	0.056895	14.91349
ARIMA	–	–	–	0.057455	0.048411	14.89455
(b)						
LOKRR	0.062015	0.059734	15.67754	0.069825	0.05492	16.96798
SVR	0.076575	0.073219	16.06394	0.081147	0.062619	17.26276
ANN	0.077809	0.075266	17.25284	0.081186	0.071484	18.05018
ARIMA	–	–	–	0.074985	0.065109	18.90466
(c)						
Online KRR	0.067464	0.064846	17.27787	0.076096	0.061412	18.89288
SVR	0.083118	0.079573	17.7488	0.088623	0.069456	19.18715
ANN	0.085629	0.083494	18.49986	0.089811	0.070026	19.29111
ARIMA	–	–	–	0.090186	0.074243	21.68558
(d)						
LOKRR	0.071581	0.066783	18.40614	0.079102	0.064982	20.19853
SVR	0.088056	0.081424	18.83573	0.093475	0.074376	20.17787
ANN	0.092487	0.090328	19.94198	0.097288	0.074856	21.07094
ARIMA	–	–	–	0.089645	0.07822	23.0101

Note: Training errors are not shown for the ARIMA model due to the difference in the model fitting procedure.

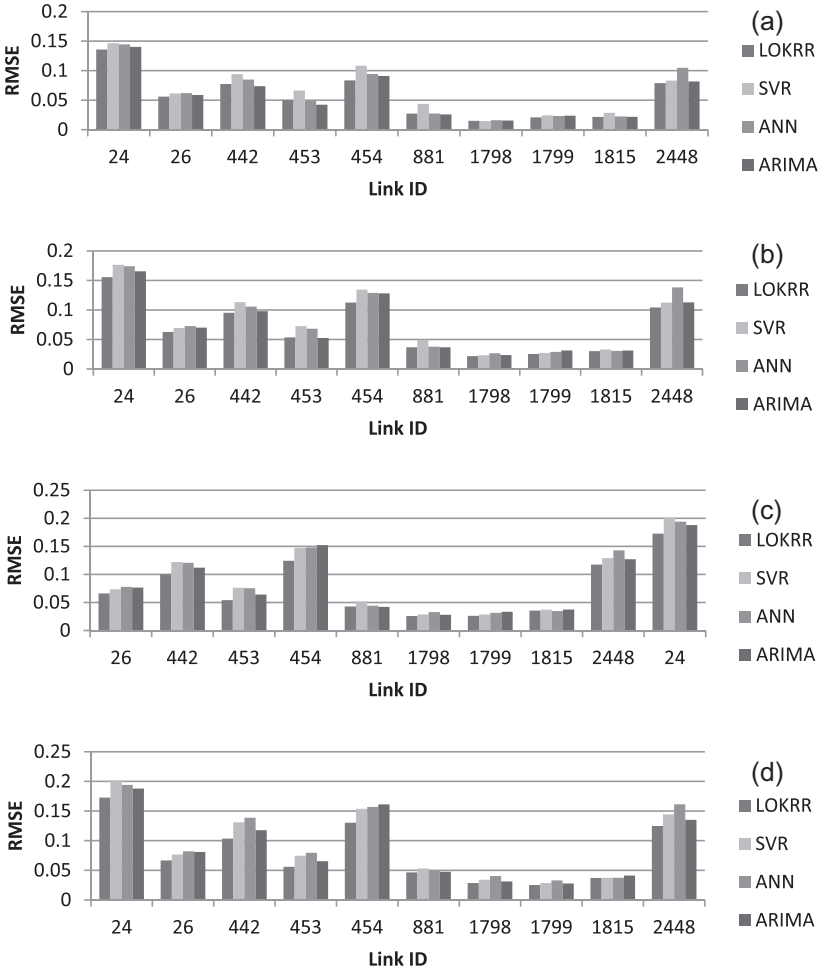

Fig. 6. RMSE of each of the models at: (a) 15 min; (b) 30 min; (c) 45 min and; (d) 60 min forecast horizons.

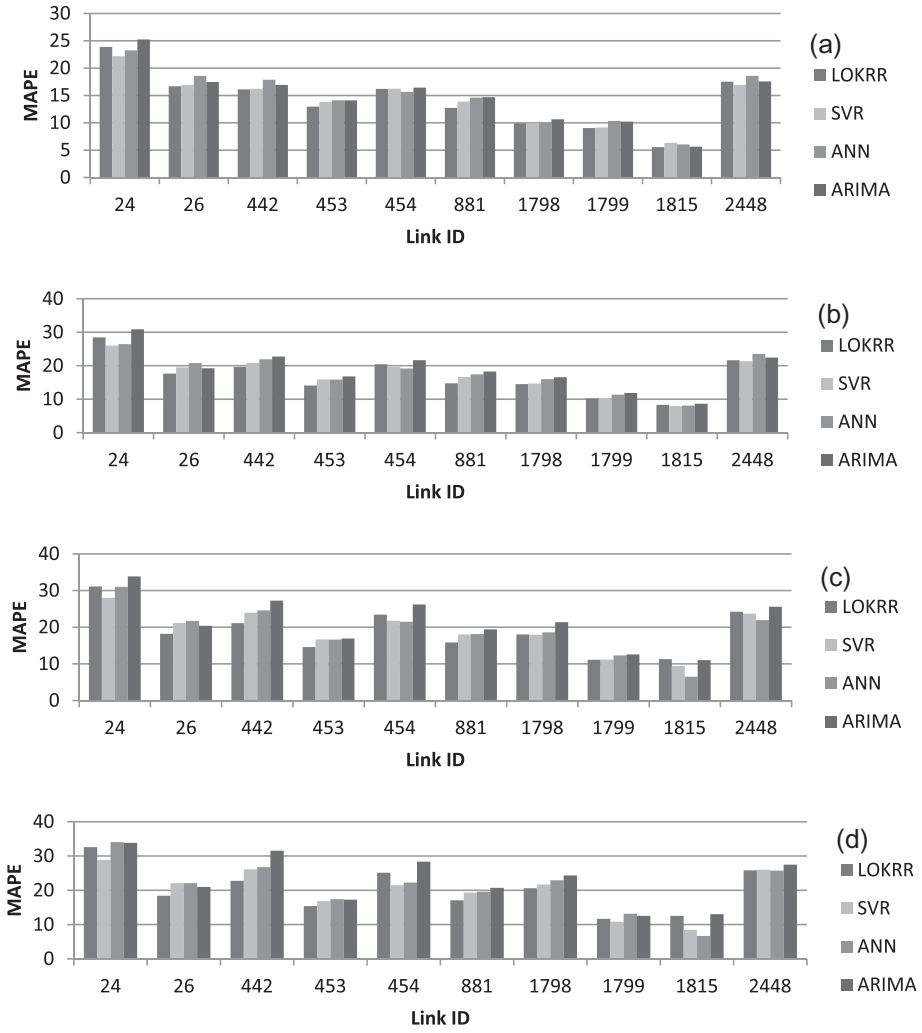

Fig. 7. MAPE of each of the models at: (a) 15 min; (b) 30 min; (c) 45 min and; (d) 60 min forecast horizons.

4.2.2. Visual analysis of results

An intuitive way to judge the performance of a forecasting model is by visual inspection of the observed series against the forecast series. For brevity, we focus on a selection of links to demonstrate the strengths and weaknesses of the LOKRR model compared with the benchmark models. Fig. 8 shows the observed versus forecast data for the LOKRR model and each of the benchmark models on Link 454, over the course of a week, at the 15 min forecast interval. On first glance, the performance of each model appears similar. However, on closer inspection, the LOKRR model performs significantly better in forecasting the onset of the congested afternoon peak period. The comparison models all exhibit a 15 min lag in forecasting the rise in UTT. This is because they are reactive models. In contrast, LOKRR is often able to forecast the onset of congestion with little or no time lag. This improved performance results from the temporal locality of LOKRR.

Weekends and weekdays have not been differentiated in the model specification described in this paper. Therefore, it is important to examine the performance of the model on weekend days. Fig. 8(e) and (f) shows the performance of the model on the weekend. On both Saturday and Sunday, LOKRR forecasts no peak in the afternoon, although there is a small spike to a UTT of around 0.3 on the Saturday (compared with around 0.8 on weekdays). This result demonstrates that, at the 15 min interval, LOKRR is able to forecast both weekdays and weekends without explicitly accounting for their different characteristics in the model specification.

Fig. 9((a)–(g)) shows the performance of the model on the same link and days at the 60 min forecast interval (30 and 45 min intervals are not shown for brevity). As expected, both the LOKRR model and the comparison models perform less well as the forecast interval increases. Most notably, none of the models are able to forecast the peaks that occur in the mornings of the Wednesday and the Sunday, shown in Fig. 9(c) and (f), respectively. However, the LOKRR model is still able to forecast the onset of congestion in the afternoon peak with little or no delay, whereas the comparison models forecast

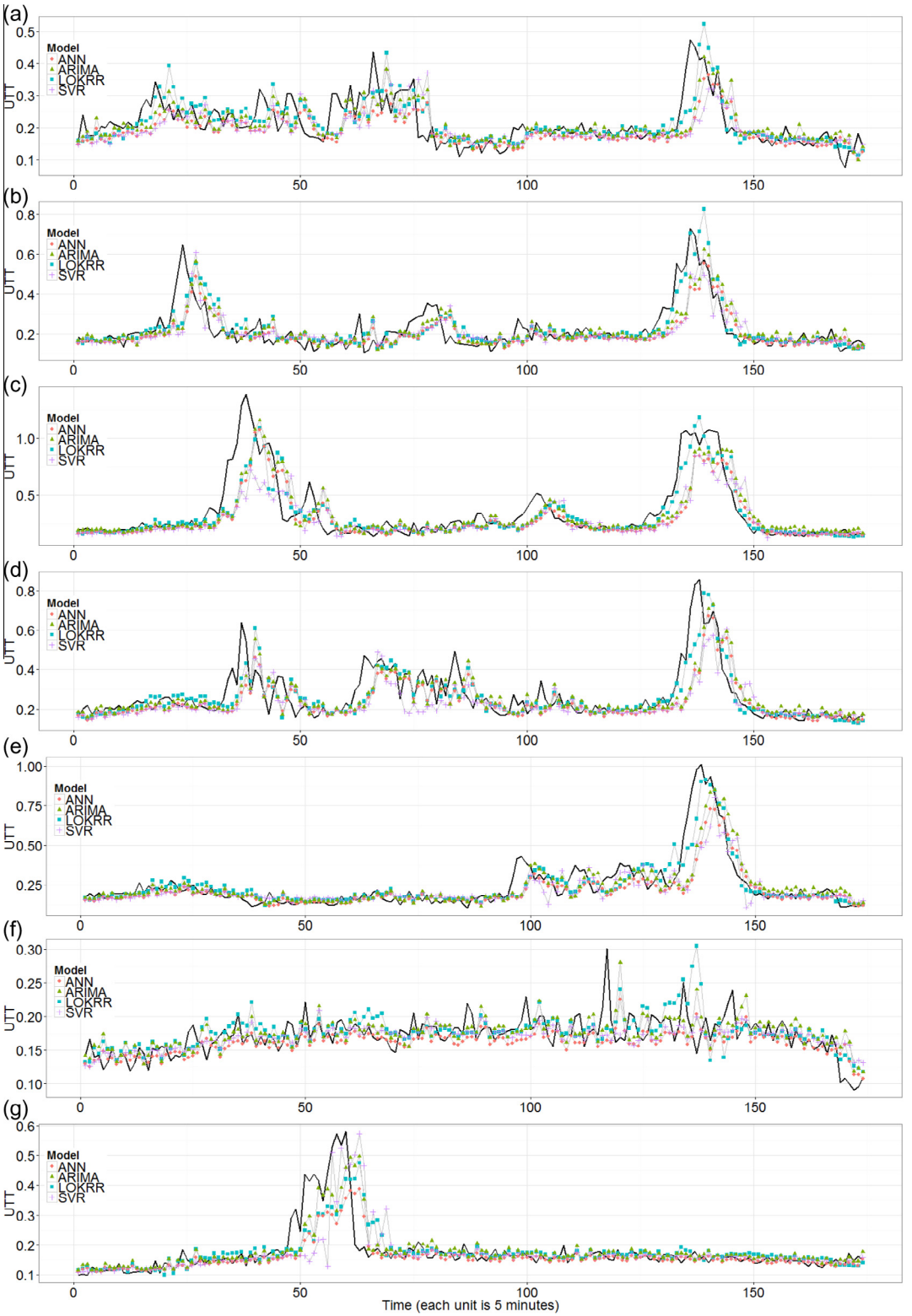

Fig. 8. Time series plots of the observed series (thick black line) against the forecast series at the 15 min interval on (a) Monday 6th June; (b) 7th June; (c) 8th June; (d) 9th June; (e) 10th June; (f) 11th June.

congestion 60 min later, when it has already begun to decrease. Moreover, the comparison models greatly underestimate the level of congestion in the peak. This result clearly illustrates the advantage of incorporating temporal locality in the model

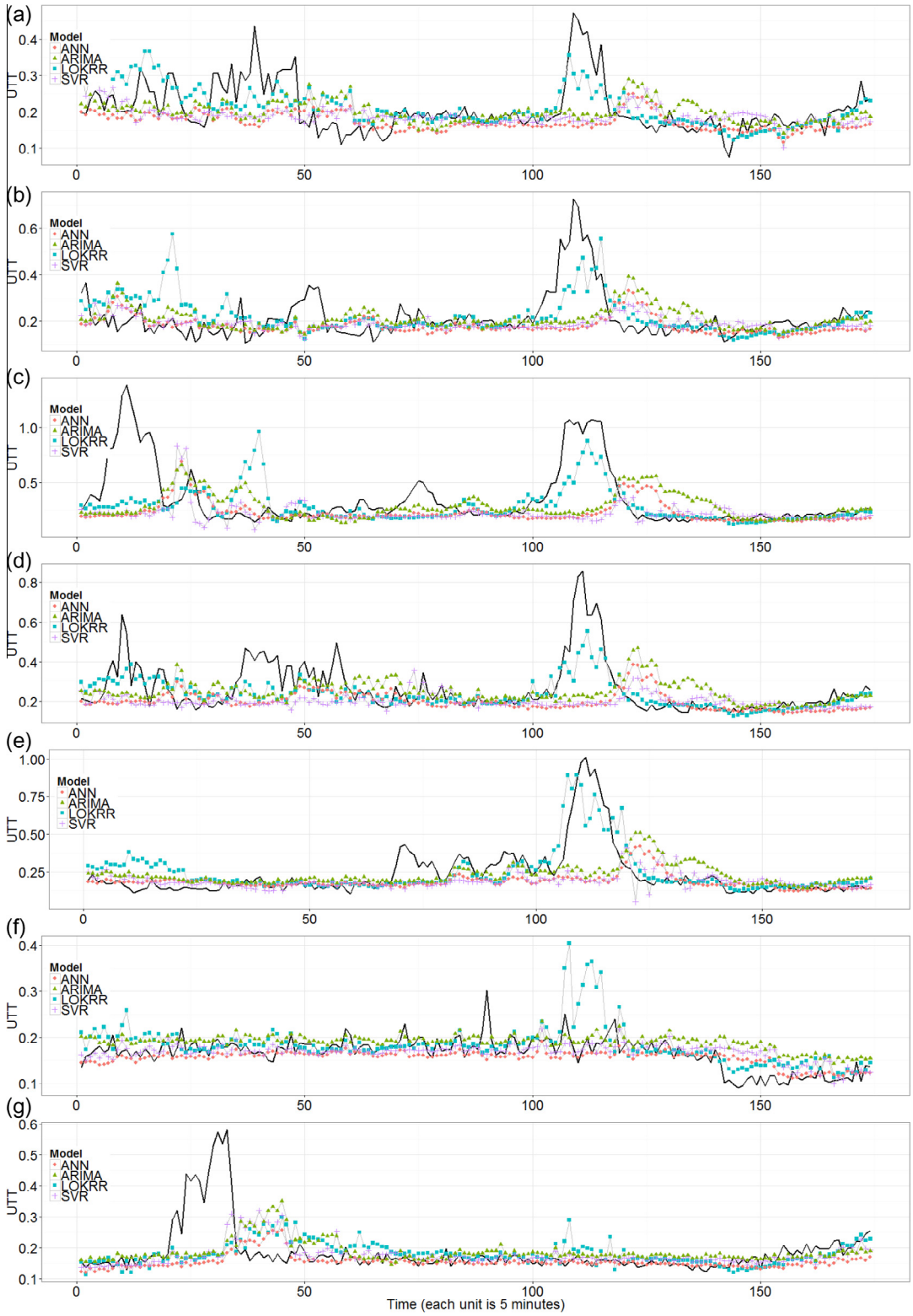

Fig. 9. Time series plots of the observed series (thick black line) against the forecast series at the 60 min interval on (a) Monday 6th June; (b) 7th June; (c) 8th June; (d) 9th June; (e) 10th June; (f) 11th June.

structure. In general, the weekend forecasting performance of the LOKRR model degrades at longer forecast intervals. It can be seen in Fig. 9e that, in contrast to the result at the 15 min interval, LOKRR erroneously forecasts a peak on the Saturday.

Fig. 10. Comparison of LOKRR with benchmark models at the 15 min interval on link 442, (a) on a typical weekday and; (b) during a non-recurrent congestion event.

This problem could be overcome by building separate kernels for weekends and weekdays, or by incorporating a mean variable that varies with day of week as well as time of day.

Accurate forecasting is of particular importance during periods of non-recurrent congestion, where the level of traffic is unexpectedly high. Fig. 10 shows the performance of the LOKRR model compared with the benchmark models on two separate days on link 442. Fig. 10(a) demonstrates the performance in forecasting a typical weekday at the 15 min forecast horizon. It can be seen that the LOKRR, SVR and ARIMA models exhibit broadly similar performance in this case, while the ANN model underestimates the peaks. Fig. 10(b) shows the performance of the models during an abnormal congestion event taking place on a weekend, where the UTT is roughly twice what it would normally be. This event may be due to an incident, or a special event taking place in the vicinity of the link. In this case, the LOKRR model is better able to forecast the evolution of

the congestion event than the comparison models. ARIMA is the best performing competitor in this case as it forecasts only the residual series.

Fig. 11 shows the forecasting performance on the same link and same days as Fig. 10 at the 60 min forecast horizon. Fig. 11(a) shows the relative performance of each of the models on the typical weekday shown in Fig. 10(a). The important feature to note is that, again, the LOKRR model is capable of forecasting the afternoon peak with less of a delay than the other models. This is consistent with the result for link 454, and is due to the fact that knowledge of the recurrent nature of traffic is built into the model. Essentially, LOKRR balances prior knowledge of traffic phenomena with observations of local conditions.

Fig. 11(b) shows the performance of the models during the abnormal congestion event. All four of the models exhibit a delay in forecasting the onset of congestion of 1 h (12 points in the figure). This is to be expected because in the univariate setting is not possible to forecast congestion events that begin after a forecast has been made. However, the LOKRR model

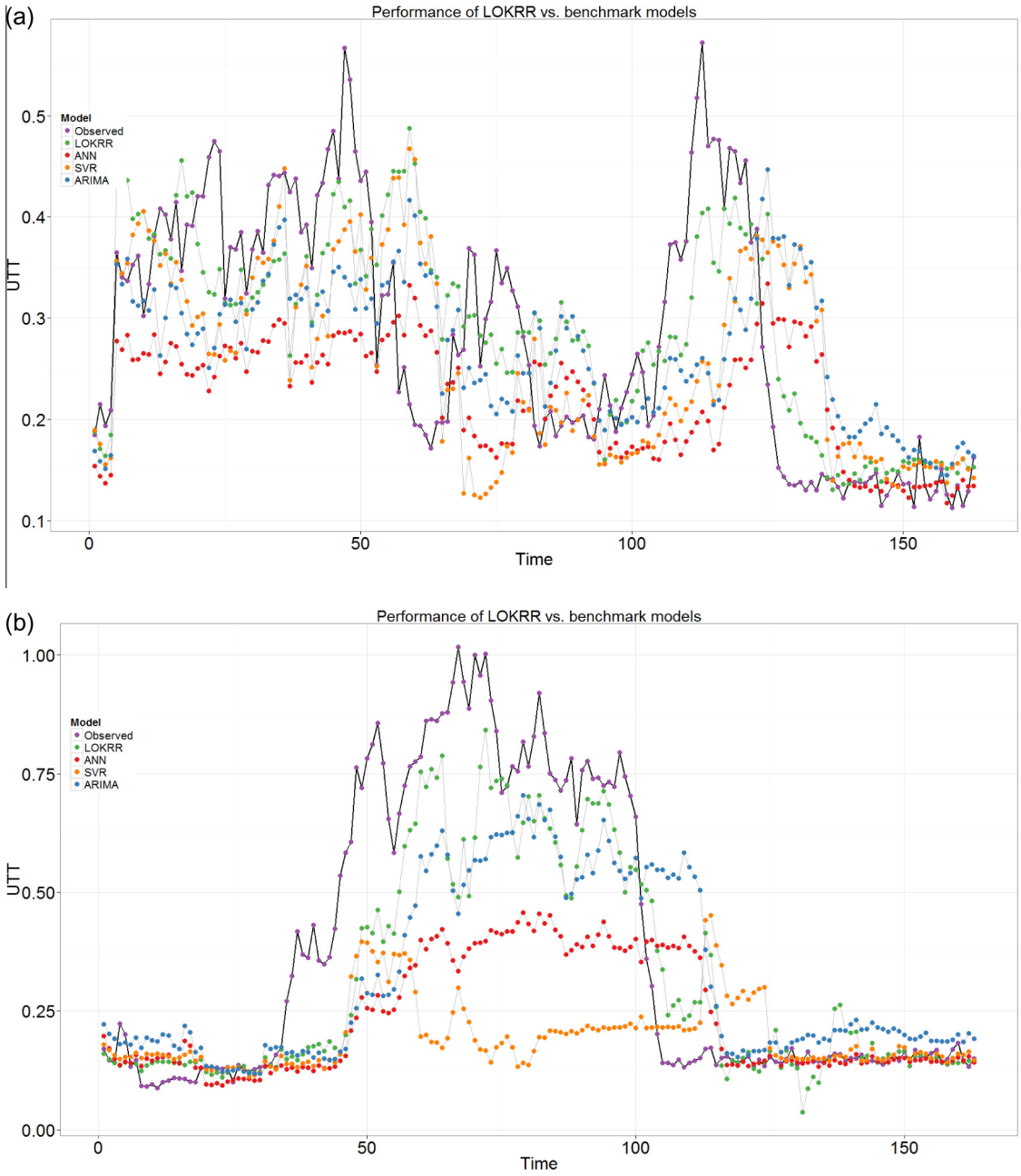

Fig. 11. Comparison of LOKRR with benchmark models at the 60 min interval on link 442, (a) on a typical weekday and; (b) during a non-recurrent congestion event.

forecasts the size and duration of the peak more accurately than the other methods. The SVR model fails in this instance, with the most likely reason being that the set of support vectors determining the solution eliminates congestion of this magnitude. LOKRR performs particularly well in forecasting the end of the congestion event. The time delay is much less than the 1 h delay exhibited by the other models. This is because LOKRR only uses those traffic patterns that were observed around the same time on previous days.

4.2.3. Investigation of model parameters

Table 3 summarises the best parameters for each link and forecast interval. All of the models exhibit best performance with $\lambda = 1/8\lambda_0$. This suggests that the method outlined in Section 2.6.2 for choosing λ may overestimate the noise in the data, and that smaller values of λ could be used (for example: $\lambda = \frac{1}{16\lambda_0}, \frac{1}{32\lambda_0}$). The best values of σ vary between the 0.25, 0.5 and 0.75 quantiles of $||\mathbf{x} - \mathbf{x}'||^2$. As the forecasting horizon increases, there is a tendency towards larger kernel bandwidths. This reflects the fact that the uncertainty in the forecasts increases with forecasting interval, so the models tend towards the historical average. With sufficiently large σ, each forecast would reduce to the average of the observations contained within w.

The model parameters σ and λ of the LOKRR are local and vary throughout the day. Each time point has a different value of σ and λ which is determined from the data. Fig. 12 shows the calculated values of the 0.25, 0.5 and 0.75 quantiles of σ at each time point and each link.

The median value of σ tends to be higher in the morning and evening peak periods than other times of day. This is particularly evident on links 442, 1798 and 1799. In general, σ is not static in time, which reflects the fact that the distribution of the data is conditional on the time of day. The finding of larger values of σ in the peak periods is consistent with empirical studies of traffic data, which have demonstrated that the mean/median and variance/interquartile range of travel times is higher in peak periods (e.g. (Fosgerau and Fukuda, 2012). However, not all of the links exhibit the same two peak profile.

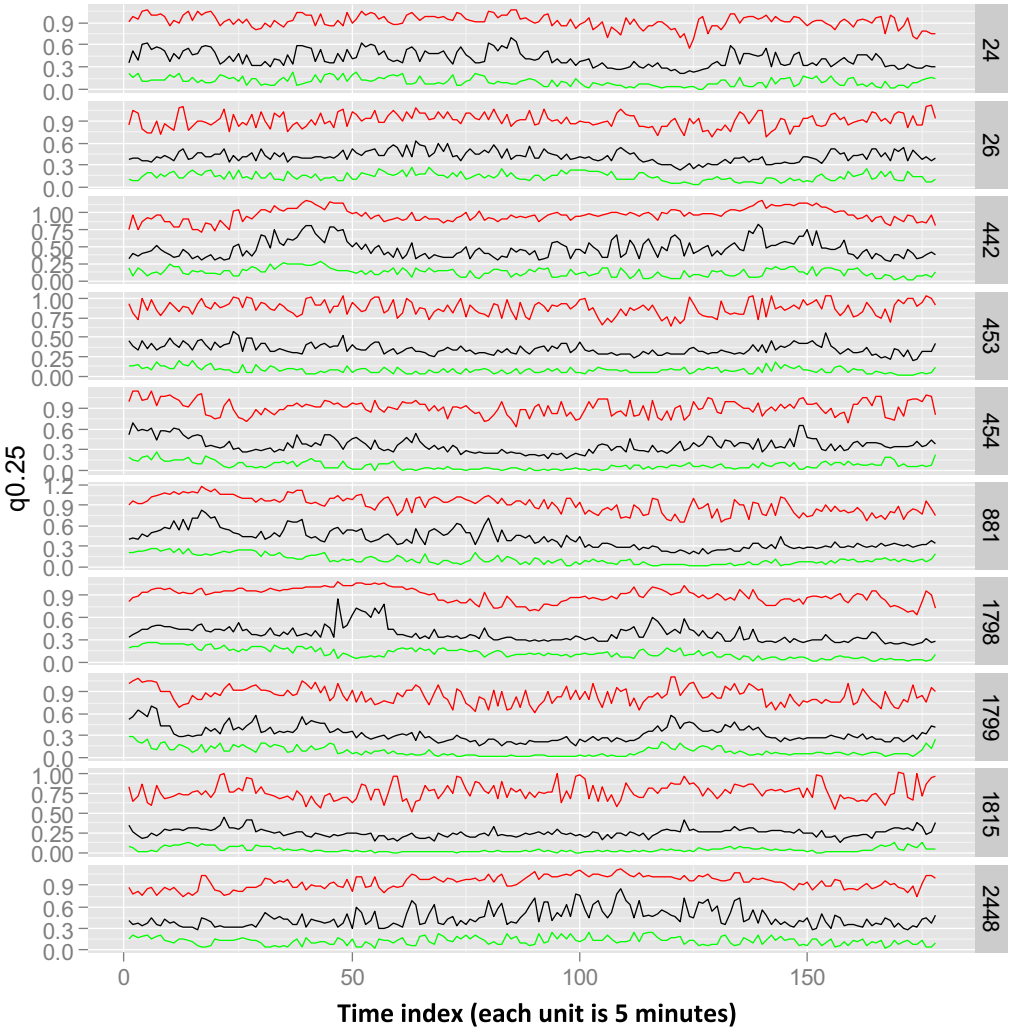

Fig. 12. Fitted values of sigma with a window size of 1. Red, black and green lines are the 0.75 quantile, median, and 0.25 quantiles respectively. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

This is because the traffic patterns of urban road links depend on many factors, including whether they are arterial (inbound or outbound) or radial routes. The LOKRR model is able to capture these differences.

Fig. 13 shows the average value of λ_0 by time of day for each of the test links. The estimated values of λ_0 are higher around the peak periods. Recalling that the values of λ_0 are obtained from the signal to noise ratio φ derived from the OLS fit of the data, this result indicates that there is a higher level of noise in the data around the peak periods. This reflects the basic intuition that traffic conditions are more difficult to predict in peak times. This is particularly marked on link 1815, which is characterised by rare, but very extreme peaks in the morning rush hour period. In this case, the value of λ_0 rises to around 80 because OLS cannot model the highly nonlinear input output relationship in this period. The variation in travel times is interpreted as noise. The implication of the higher values of λ_0 in the peak periods is that forecasts will tend to be closer to the average because high values of λ penalise large weights. This is undesirable when forecasting non-recurrent congestion events. However, searching the grid of multiples of λ_0 in a cross-validation approach circumvents this problem. On other links, the values of λ_0 are higher during the early morning and late evening periods. In these cases, the parameter values correctly identify the increased noise in the data at times of lower flow that is caused by fewer vehicles being observed by the ANPR cameras.

4.2.4. Sensitivity of model parameters

The sensitivity of model parameters is an important concern in forecasting models because it determines how easily they can be trained. In this context, the meaning of parameter sensitivity is the effect that a change in the value of a parameter has on the performance of the model. As mentioned in Section 2.6, there is an extensive literature on parameter selection in kernel methods such as SVR and KRR. However, in practice one may be willing to accept a suboptimal model if the advantage

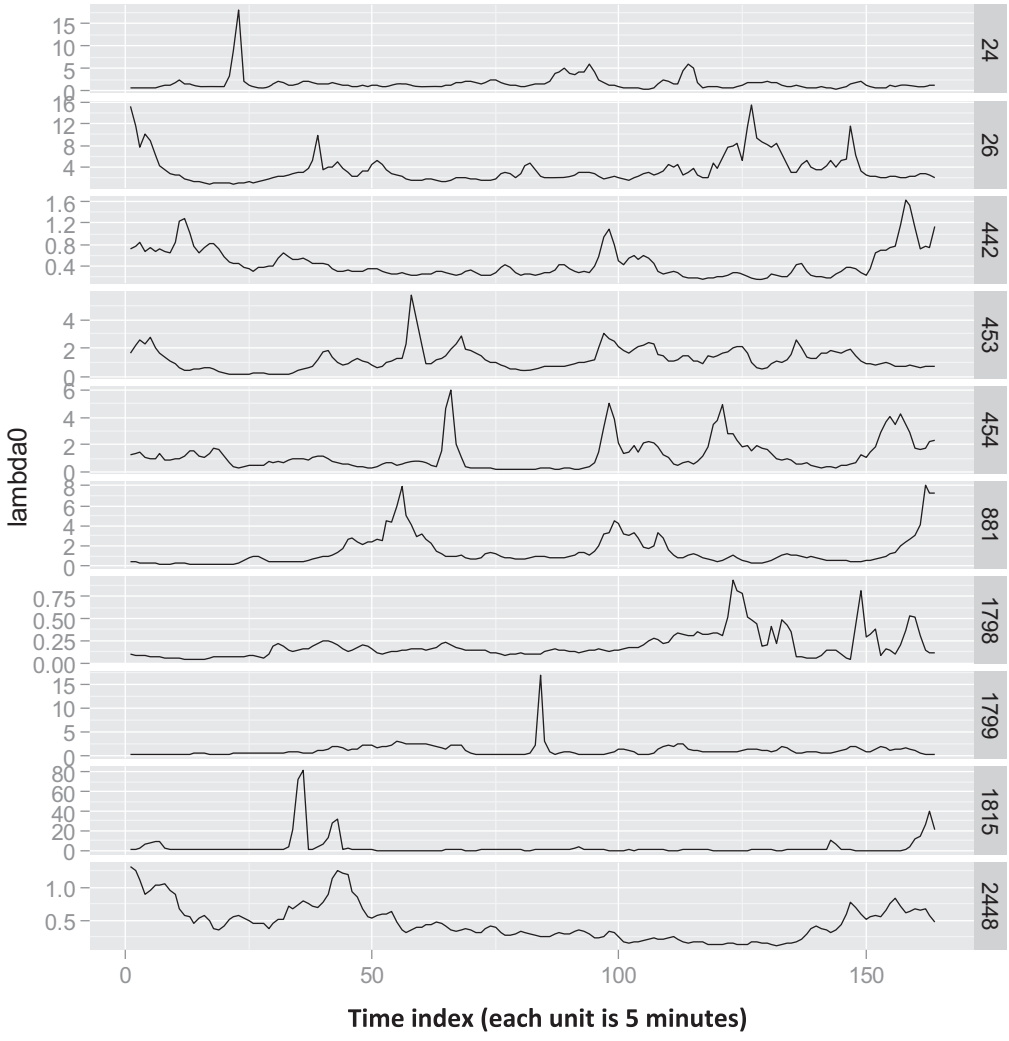

Fig. 13. Fitted values of Lambda 0 with $w = 1$.

gained in training time reduction or ease of implementation outweighs the potential advantage of increased accuracy. This is a feasible option if the parameters are insensitive, and good results can be guaranteed within a small range of values.

Fig. 14 shows the effect of varying σ through its range, while keeping λ fixed in a forecasting situation. The data shown here are on link 442 over the course of a single week. In the figures, the large peak in UTT on the Friday represents an abnormal congestion event. The effect of varying λ on model performance is significant. Choosing a large value of λ results in an overestimation of the noise in the data and an overly smooth solution. Consequently, models using large values of λ do not perform strongly in forecasting abnormally high peaks in travel times. This result, combined with the model parameters of the fitted models shown in Table 3, indicates that it is not necessary to test larger values of λ, supporting the intuition of Exterkate (2013, p. 6) that λ should be less than λ_0 because one expects to obtain a better fit using nonlinear models.

Fig. 15 shows the effect of holding σ fixed and varying λ. Fig. 15((a)–(c)) all appear broadly similar, indicating that the effect of varying σ is not as great as the effect of varying λ. This supports the statement of Caputo et al. (2002) that any kernel bandwidth between the 0.1 and 0.9 quantiles of the kernel will produce good results. The implication of this is that the performance of LOKRR is relatively insensitive to Gaussian kernel bandwidth choice when the method described in Section 2.6.2 is used. Similar results are found on the other links in the network.

Some conclusions can be drawn from the sensitivity analysis that make model selection more straightforward. Firstly, small values of λ always yield greater accuracy than large values, meaning it is unnecessary to test large multiples of λ_0.

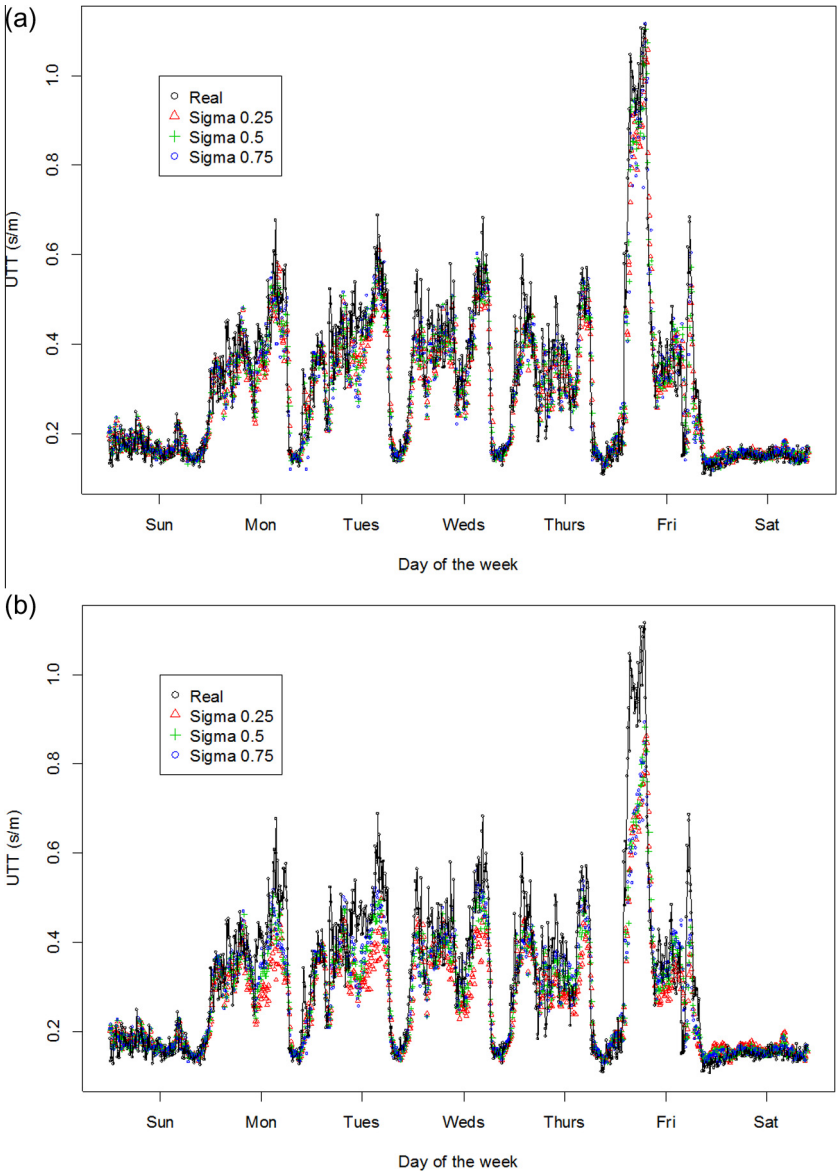

Fig. 14. Effect of varying σ through its range with Lambda fixed at: (a) 1/8 λ_0 and (b) 2 λ_0.

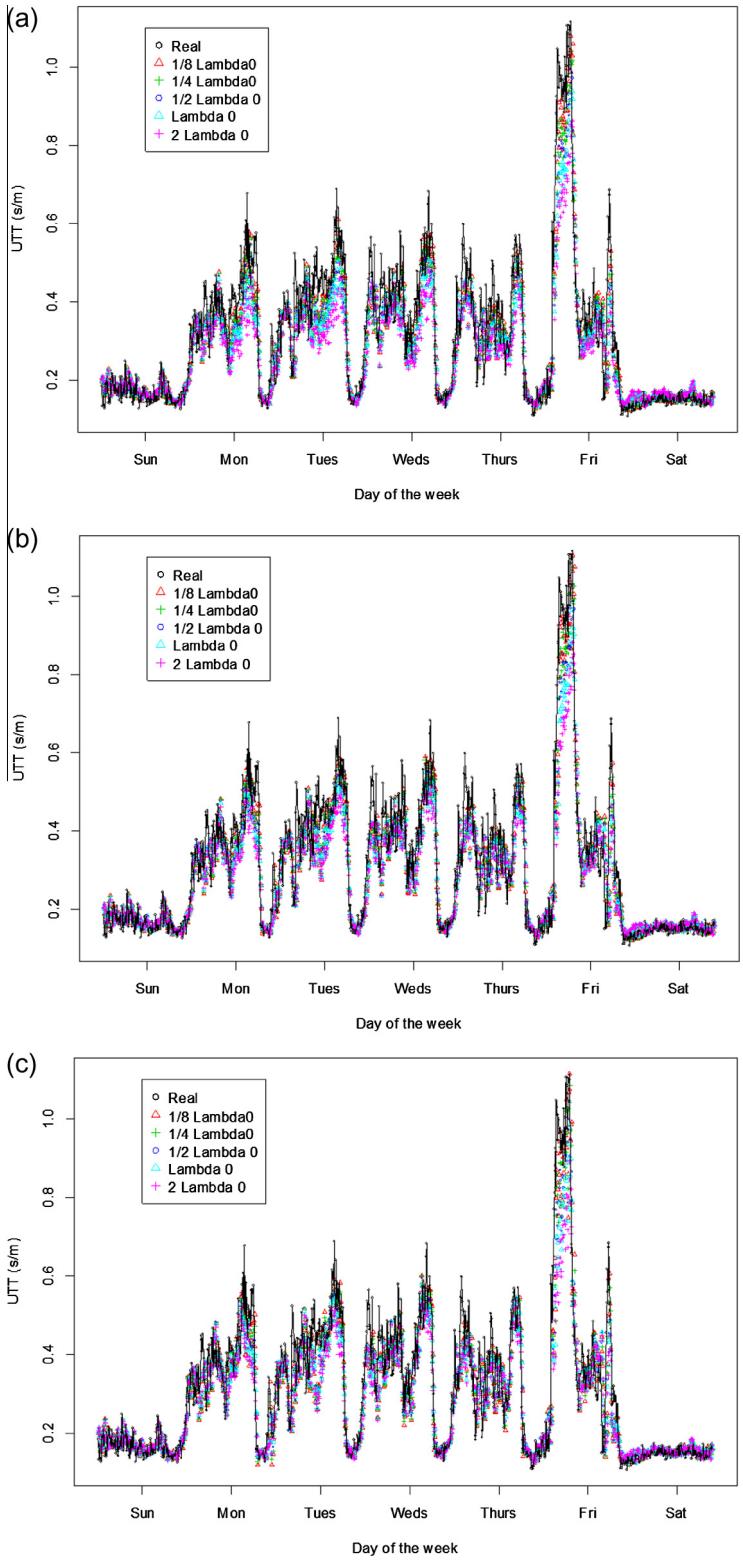

Fig. 15. Effect of varying λ through its range with σ fixed at: (a) 0.25 quantile, (b) median (c) 0.75 quantile.

Secondly, although performance varies with choice of σ, a good fit to the data can be obtained with the limited parameter range suggested.

5. Conclusion and future directions

In this paper, a temporally local kernel based forecasting model, namely LOKRR, has been developed for forecasting traffic series. To incorporate the cyclic patterns inherent in traffic data, a separate kernel with individual parameters is defined for each time point. This structure has two main benefits. Firstly, it enables smaller kernels to be accessed each time a forecast is made, making the algorithm efficient. Secondly, it allows the model to capture the seasonality and heteroskedasticity in the series. The empirical case study demonstrates the strong performance of the model in forecasting up to 1 h ahead. This ability makes the model suitable for real time application, where accurate forecasts of future traffic conditions are important.

The strength of LOKRR lies in its ability to model the cyclic variation in travel times throughout the day. Each forecast is a weighted nonlinear combination of previously observed travel times from within the temporal window w. This constrains the model to only produce forecasts based on what has happened during this window in the past. Therefore, it naturally captures the cyclic pattern in the data. At shorter forecasting intervals, the model is able to use recent traffic patterns to accurately forecast the variation above or below the average conditions. As the forecast horizon increases and more uncertainty is introduced to the forecasts, forecasts tend towards the historical average. However, LOKRR is still able to produce more accurate forecasts than the historical average, indicating that it captures the local conditions even at longer forecast horizons. The ability to choose parameters from a small range of initial values determined from the data is also advantageous.

The model has a number of limitations that it shares with the benchmark models. Firstly, it cannot forecast the onset of abnormal congestion before it is observed in the data. This problem can be addressed either by using a spatio-temporal approach or by including additional information such as incident data. KMs are inherently suited to incorporating data from various sources, and this will form the focus of future research. Currently, a spatio-temporal extension of LOKRR, namely space–time (ST) LOKRR is being developed. The main challenge in this regard lies in determining what spatio-temporal information is relevant for forecasting traffic on a given link at a particular time. This is a problem of spatio-temporal neighbourhood selection (Haworth and Cheng, 2014), and the authors have tackled this in the context of statistical modelling (Cheng et al., 2014). Secondly, the method for online removal and addition of data from the kernel is arbitrary, and future research will focus on methods for selective data addition/removal. This problem has been addressed recently in the context of the KRLS algorithm, and could be applied to LOKRR (Van Vaerenbergh et al., 2010, 2012). Finally, the time of day varying structure specified in this paper can be relaxed to include many types of time varying structure. For example, separate kernels can be defined for weekends and weekdays, or longer periods such as peak and non-peak. Alternatively, traffic patterns could be grouped according to states such as congested and free flowing. This would provide a less rigid temporal structure, but would require an additional classification step.

At the time of writing, the LOKRR model has not yet been transferred into operation. This step necessitates calibrating the model locally for many hundreds or possibly thousands of measurement locations, LOKRR is computationally efficient for a single location because it requires small kernels to be evaluated at each time point. However, when implemented serially, the computational time scales linearly with the number of locations to be forecast. It is necessary, therefore, to take a parallel approach to model training and implementation. The LOKRR model has the embarrassingly parallel property, meaning that the model for each location is independent of the models for other locations (Harris et al., 2010). This makes LOKRR suitable for parallel implementation. In this study, and in our other work on this network, the UCL Legion High Performance Computing Facility (Legion@UCL) has been used for model training, enabling many locations to be modelled simultaneously. The way in which parallelism is exploited depends on the hardware available to the practitioner, but cloud computing resources and graphics processing units (GPUs) are becoming available at decreasing cost. Routines optimised for parallel technology that can carry out many of the required computations are readily available. Currently, the authors are working on a GPU based implementation of LOKRR.

Acknowledgements

We thank Transport for London for providing the travel time data. The research for this article was carried out under the STANDARD project, sponsored by the U.K. Engineering and Physical Sciences Research Council (EP/G023212/1). Comments from the editor and the anonymous reviewers are greatly appreciated and improved the content and readability of the article. The authors bear sole responsibility for any mistakes that may appear.

References

Anacleto, O., Queen, C., Albers, C.J., 2013a. Forecasting multivariate road traffic flows using Bayesian dynamic graphical models, splines and other traffic variables. Aust. N. Z. J. Stat. 55, 69–86. http://dx.doi.org/10.1111/anzs.12026.

Anacleto, O., Queen, C., Albers, C.J., 2013b. Multivariate forecasting of road traffic flows in the presence of heteroscedasticity and measurement errors. J. R. Stat. Soc. Ser. C Appl. Stat. 62, 251–270. http://dx.doi.org/10.1111/j.1467-9876.2012.01059.x.

Bergmeir, C., Benítez, J.M., 2012. Neural networks in R using the Stuttgart neural network simulator: RSNNS. J. Stat. Softw. 46, 1–26.

Billings, D., Yang, J.-S., 2006. Application of the ARIMA models to urban roadway travel time prediction – a case study. In: 2006 IEEE International Conference on Systems, Man, and Cybernetics. Presented at the 2006 IEEE International Conference on Systems, Man, and Cybernetics, Taipei.

Boser, B.E., Guyon, I.M., Vapnik, V.N., 1992. A training algorithm for optimal margin classifiers. In: Proceedings of the 5th Annual ACM Workshop on Computational Learning Theory. ACM Press, pp. 144–152.

Box, G.E.P., Jenkins, G.M., Reinsel, G.C., 1994. Time Series Analysis: Forecasting and Control. Prentice Hall.

Cai, Z., Li, S., Zhang, X., 2009. Tourism demand forecasting by support vector regression and genetic algorithm. In: Computer Science and Information Technology, 2009. ICCSIT 2009. 2nd IEEE International Conference on. Presented at the Computer Science and Information Technology, 2009. ICCSIT 2009. 2nd IEEE International Conference on, pp. 144–146. http://dx.doi.org/10.1109/ICCSIT.2009.5234447.

Caputo, B., Sim, K., Furesjo, F., Smola, A., 2002. Appearance-based object recognition using SVMs: which kernel should I use? In: Proceedings of NIPS Workshop on Statistical Methods for Computational Experiments in Visual Processing and Computer Vision. Presented at the NIPS Workshop on Statistical Methods for Computational Experiments in Visual Processing and Computer Vision, Whistler.

Castro-Neto, M., Jeong, Y.S., Jeong, M.K., Han, L.D., 2009. Online-SVR for short-term traffic flow prediction under typical and atypical traffic conditions. Expert Syst. Appl. 36, 6164–6173.

Chan, K.Y., Dillon, T., Chang, E., Singh, J., 2013a. Prediction of short-term traffic variables using intelligent swarm-based neural networks. IEEE Trans. Control Syst. Technol. 21, 263–274. http://dx.doi.org/10.1109/TCST.2011.2180386.

Chan, K.Y., Dillon, T.S., Chang, E., 2013b. An Intelligent particle swarm optimization for short-term traffic flow forecasting using on-road sensor systems. IEEE Trans. Ind. Electron. 60, 4714–4725. http://dx.doi.org/10.1109/TIE.2012.2213556.

Chan, K.Y., Dillon, T.S., Singh, J., Chang, E., 2012a. Neural-network-based models for short-term traffic flow forecasting using a hybrid exponential smoothing and Levenberg–Marquardt algorithm. IEEE Trans. Intell. Transport. Syst. 13, 644–654. http://dx.doi.org/10.1109/TITS.2011.2174051.

Chan, K.Y., Khadem, S., Dillon, T.S., Palade, V., Singh, J., Chang, E., 2012b. Selection of significant on-road sensor data for short-term traffic flow forecasting using the taguchi method. IEEE Trans. Ind. Inform. 8, 255–266. http://dx.doi.org/10.1109/TII.2011.2179052.

Chen, C., Wang, Y., Li, L., Hu, J., Zhang, Z., 2012. The retrieval of intra-day trend and its influence on traffic prediction. Transport. Res. Part C Emerg. Technol. 22, 103–118. http://dx.doi.org/10.1016/j.trc.2011.12.006.

Chen, H., Grant-Muller, S., 2001. Use of sequential learning for short-term traffic flow forecasting. Transport. Res. Part C Emerg. Technol. 9, 319–336. http://dx.doi.org/10.1016/S0968-090X(00)00039-5.

Cheng, T., Wang, J., Haworth, J., Heydecker, B., Chow, A., 2014. A dynamic spatial weight matrix and localized space-time autoregressive integrated moving average for network modeling. Geogr. Anal. 46, 75–97. http://dx.doi.org/10.1111/gean.12026.

Cheng, T., Wang, J., Heydecker, B., Haworth, J., 2010. STARIMA for journey time prediction in London. In: Proceedings of the 5th IMA (Institute of Mathematics and Its Applications) Conference on Mathematics in Transport. London, UK.

Dimitriou, L., Tsekeris, T., Stathopoulos, A., 2008. Adaptive hybrid fuzzy rule-based system approach for modeling and predicting urban traffic flow. Transport. Res. Part C Emerg. Technol. 16, 554–573, 16/j.trc.2007.11.003.

Ding, Q.Y., Wang, X.F., Zhang, X.Y., Sun, Z.Q., 2010. Forecasting traffic volume with space-time ARIMA model. Adv. Mater. Res. 156–157, 979–983. http://dx.doi.org/10.4028/www.scientific.net/AMR.156-157.979.

Dougherty, M., 1995. A review of neural networks applied to transport. Transport. Res. Part C Emerg. Technol. 3, 247–260. http://dx.doi.org/10.1016/0968-090X(95)00009-8.

Elman, J.L., 1990. Finding structure in time∗ 1. Cogn. Sci. 14, 179–211.

Exterkate, P., 2013. Model selection in kernel ridge regression. Comput. Stat. Data Anal. 68, 1–16. http://dx.doi.org/10.1016/j.csda.2013.06.006.

Fei, X., Lu, C.-C., Liu, K., 2011. A Bayesian dynamic linear model approach for real-time short-term freeway travel time prediction. Transport. Res. Part C Emerg. Technol. 19, 1306–1318. http://dx.doi.org/10.1016/j.trc.2010.10.005.

Fosgerau, M., Fukuda, D., 2012. Valuing travel time variability: characteristics of the travel time distribution on an urban road. Transport. Res. Part C Emerg. Technol. 24, 83–101. http://dx.doi.org/10.1016/j.trc.2012.02.008.

Hamilton, J.D., 1994. Time Series Analysis. Princeton University Press.

Hardoon, D.R., Szedmak, S., Shawe-Taylor, J., 2004. Canonical correlation analysis: an overview with application to learning methods. Neural Comput. 16, 2639–2664. http://dx.doi.org/10.1162/0899766042321814.

Harris, R., Singleton, A., Grose, D., Brunsdon, C., Longley, P., 2010. Grid-enabling geographically weighted regression: a case study of participation in higher education in England. Trans. GIS 14, 43–61. http://dx.doi.org/10.1111/j.1467-9671.2009.01181.x.

Haworth, J., Cheng, T., 2014. Graphical LASSO for local spatio-temporal neighbourhood selection. In: Proceedings the GIS Research UK 22nd Annual Conference. Presented at the GISRUK 2014, University of Glasgow, Glasgow, Scotland, pp. 425–433.

Hoerl, A.E., Kennard, R.W., 1970. Ridge regression: biased estimation for nonorthogonal problems. Technometrics 12, 55–67. http://dx.doi.org/10.2307/1267351.

Hofleitner, A., Herring, R., Bayen, A., 2012. Arterial travel time forecast with streaming data: a hybrid approach of flow modeling and machine learning. Transport. Res. Part B Methodol. 46, 1097–1122. http://dx.doi.org/10.1016/j.trb.2012.03.006.

Hong, W.-C., 2010. Application of seasonal SVR with chaotic immune algorithm in traffic flow forecasting. Neural Comput. Appl. http://dx.doi.org/10.1007/s00521-010-0456-7.

Hong, W.-C., 2011. Traffic flow forecasting by seasonal SVR with chaotic simulated annealing algorithm. Neurocomputing 74, 2096–2107. http://dx.doi.org/10.1016/j.neucom.2010.12.032.

Hong, W.-C., 2012. Application of seasonal SVR with chaotic immune algorithm in traffic flow forecasting. Neural Comput. Appl. 21, 583–593. http://dx.doi.org/10.1007/s00521-010-0456-7.

Hong, W.-C., Dong, Y., Zheng, F., Lai, C.-Y., 2011. Forecasting urban traffic flow by SVR with continuous ACO. Appl. Math. Model. 35, 1282–1291. http://dx.doi.org/10.1016/j.apm.2010.09.005.

Hotelling, H., 1936. Relations between two sets of variates. Biometrika 28, 321–377. http://dx.doi.org/10.1093/biomet/28.3-4.321.

Huang, S., Sadek, A.W., 2009. A novel forecasting approach inspired by human memory: the example of short-term traffic volume forecasting. Transport. Res. Part C Emerg. Technol. 17, 510–525. http://dx.doi.org/10.1016/j.trc.2009.04.006.

Hyndman, R.J., Khandakar, Y., 2007. Automatic Time Series for Forecasting: the Forecast Package for R.

Hyndman, R.J., Koehler, A.B., 2006. Another look at measures of forecast accuracy. Int. J. Forecast. 22, 679–688. http://dx.doi.org/10.1016/j.ijforecast.2006.03.001.

Jordan, M., 1986. Attractor dynamics and parallellism in a connectionist sequential machine. Presented at the Proceedings of the Eighth Annual Meeting of the Cognitive Science Society, Lawrence Erlbaum Associates, pp. 531–546.

Juszczak, P., Tax, D., Duin, R.P.W., 2002. Feature scaling in support vector data description. In: Proc. ASCI. pp. 95–102.

Kamarianakis, Y., Oliver Gao, H., Prastacos, P., 2010. Characterizing regimes in daily cycles of urban traffic using smooth-transition regressions. Transport. Res. Part C Emerg. Technol. 18, 821–840. http://dx.doi.org/10.1016/j.trc.2009.11.001.

Kamarianakis, Y., Prastacos, P., 2005. Space-time modeling of traffic flow. Comput. Geosci. 31, 119–133. http://dx.doi.org/10.1016/j.cageo.2004.05.012.

Kamarianakis, Y., Shen, W., Wynter, L., 2012. Real-time road traffic forecasting using regime-switching space-time models and adaptive LASSO. Appl. Stoch. Models Bus. Ind. 28, 297–315. http://dx.doi.org/10.1002/asmb.1937.

Karatzoglou, A., Meyer, D., Hornik, K., 2006. Support vector machines in R. J. Stat. Softw. 15, 1–28.

Karatzoglou, A., Smola, A., Hornik, K., Zeileis, A., 2004. kernlab-An S4 package for kernel methods in R.

Karlaftis, M.G., Vlahogianni, E.I., 2011. Statistical methods versus neural networks in transportation research: differences, similarities and some insights. Transport. Res. Part C Emerg. Technol. 19, 387–399. http://dx.doi.org/10.1016/j.trc.2010.10.004.

Keerthi, S.S., Lin, C.-J., 2003. Asymptotic behaviors of support vector machines with gaussian kernel. Neural Comput. 15, 1667–1689. http://dx.doi.org/10.1162/089976603321891855.

Kendall, M.G., Ord, J.K., 1990. Time Series. E. Arnold.

Krige, d g, 1951. A Statistical Approach to Some Mine Valuation and Allied Problems on the Witwatersrand.

Lahiri, S.K., Ghanta, K.C., 2008. The Support Vector Regression with the parameter tuning assisted by a differential evolution technique: study of the critical velocity of a slurry flow in a pipeline. Association of Chemical Engineers.

Ledoux, C., 1997. An urban traffic flow model integrating neural networks. Transport. Res. Part C Emerg. Technol. 5, 287–300. http://dx.doi.org/10.1016/S0968-090X(97)00015-6.
Li, J., Cai, Z., 2008. A novel automatic parameters optimization approach based on differential evolution for support vector regression. Adv. Comput. Intell., 510–519.
Li, R., Rose, G., 2011. Incorporating uncertainty into short-term travel time predictions. Transport. Res. Part C Emerg. Technol. 19, 1006–1018. http://dx.doi.org/10.1016/j.trc.2011.05.014.
Li, W., Yang, Y., 2008. Hybrid kernel learning via genetic optimization for TS fuzzy system identification. Int. J. Adapt. Control Signal Process. n/a–n/a. http://dx.doi.org/10.1002/acs.1089.
Li, X., Shao, M., Ding, L., Xu, G., Li, J., 2010. Particle swarm optimization-based LS-SVM for building cooling load prediction. J. Comput. 5. http://dx.doi.org/10.4304/jcp.5.4.614-621.
Min, W., Wynter, L., 2011. Real-time road traffic prediction with spatio-temporal correlations. Transport. Res. Part C Emerg. Technol. 19, 606–616. http://dx.doi.org/10.1016/j.trc.2010.10.002.
Min, X., Hu, J., Chen, Q., Zhang, T., Zhang, Y., 2009. Short-term traffic flow forecasting of urban network based on dynamic STARIMA model. In: Intelligent Transportation Systems, 2009. ITSC '09. 12th International IEEE Conference on. Presented at the Intelligent Transportation Systems, 2009. ITSC '09. 12th International IEEE Conference on, pp. 1–6. http://dx.doi.org/10.1109/ITSC.2009.5309741.
Min, X., Hu, J., Zhang, Z., 2010. Urban traffic network modeling and short-term traffic flow forecasting based on GSTARIMA model. In: Intelligent Transportation Systems (ITSC), 2010 13th International IEEE Conference on. Presented at the Intelligent Transportation Systems (ITSC), 2010 13th International IEEE Conference on, pp. 1535–1540. http://dx.doi.org/10.1109/ITSC.2010.5625123.
Mitchell, T.M., 1997. Machine Learning, 1st ed. McGraw-Hill Science/Engineering/Math.
Okutani, I., Stephanedes, Y.J., 1984. Dynamic prediction of traffic volume through Kalman filtering theory. Transport. Res. Part B Methodol. 18, 1–11.
Ozaki, T., 1977. On the order determination of ARIMA Models. J. R. Stat. Soc. Ser. C Appl. Stat. 26, 290–301. http://dx.doi.org/10.2307/2346970.
Poggio, T., Girosi, F., 1990. Regularization algorithms for learning that are equivalent to multilayer networks. Science 247, 978–982. http://dx.doi.org/10.1126/science.247.4945.978.
Rasmussen, C.E., Williams, C., 2006. Gaussian Processes for Machine Learning. MIT Press, Cambridge, MA.
Saunders, C., Gammerman, A., Vovk, V., 1998. Ridge regression learning algorithm in dual variables, in: (ICML-1998) Proceedings of the 15th International Conference on Machine Learning. pp. 515–521.
Schölkopf, B., Smola, A., Müller, K.-R., 1997. Kernel Principal Component Analysis. In: Gerstner, W., Germond, A., Hasler, M., Nicoud, J.-D. (Eds.), Artificial Neural Networks — ICANN'97, Lecture Notes in Computer Science. Springer, Berlin Heidelberg, pp. 583–588.
Shawe-Taylor, J., Cristianini, N., 2004. Kernel Methods for Pattern Analysis. Cambridge University Press, New York, NY, USA.
Sun, Shiliang, Zhang, Changshui, Guoqiang, Yu, 2006. A bayesian network approach to traffic flow forecasting. Intell. Transport. Syst. IEEE Trans. 7, 124–132. http://dx.doi.org/10.1109/TITS.2006.869623.
Smith, B., Demetsky, M., 1996. Multiple-interval freeway traffic flow forecasting. Transport. Res. Rec. J. Transp. Res. Board 1554, 136–141. http://dx.doi.org/10.3141/1554-17.
Smola, A.J., Sch/ölkopf, B, 2004. A tutorial on support vector regression. Stat. Comput. 14, 199–222.
Stathopoulos, A., Karlaftis, M.G., 2003. A multivariate state space approach for urban traffic flow modeling and prediction. Transport. Res. Part C Emerg. Technol. 11, 121–135. http://dx.doi.org/10.1016/S0968-090X(03)00004-4.
Sun, S., Zhang, C., Yu, G., Lu, N., Xiao, F., 2004. Bayesian network methods for traffic flow forecasting with incomplete data. Mach. Learn. ECML 2004, 419–428.
Sun, S., Zhang, C., Zhang, Y., 2005. Traffic flow forecasting using a spatio-temporal bayesian network predictor. Artif. Neural Netw. Form. Models Their Appl.-ICANN 2005 273–278.
Üstün, B., Melssen, W.J., Oudenhuijzen, M., Buydens, L.M.C., 2005. Determination of optimal support vector regression parameters by genetic algorithms and simplex optimization. Anal. Chim. Acta 544, 292–305. http://dx.doi.org/10.1016/j.aca.2004.12.024.
Van Der Voort, M., Dougherty, M., Watson, S., 1996. Combining Kohonen maps with arima time series models to forecast traffic flow. Transport. Res. Part C Emerg. Technol. 4, 307–318. http://dx.doi.org/10.1016/S0968-090X(97)82903-8.
Van Hinsbergen, C.P.I., van Lint, J.W.C., van Zuylen, H.J., 2009. Bayesian committee of neural networks to predict travel times with confidence intervals. Transport. Res. Part C Emerg. Technol. 17, 498–509. http://dx.doi.org/10.1016/j.trc.2009.04.007.
Van Lint, J.W.C., 2006. Reliable real-time framework for short-term freeway travel time prediction. J. Transport. Eng. 132, 921–932. http://dx.doi.org/10.1061/(ASCE)0733-947X(2006) 132:12(921).
Van Lint, J.W.C., Hoogendoorn, S.P., van Zuylen, H.J., 2005. Accurate freeway travel time prediction with state-space neural networks under missing data. Transport. Res. Part C Emerg. Technol. 13, 347–369. http://dx.doi.org/10.1016/j.trc.2005.03.001.
Van Vaerenbergh, S., Lazaro-Gredilla, M., Santamaria, I., 2012. Kernel recursive least-squares tracker for time-varying regression. IEEE Trans. Neural Netw. Learn. Syst. 23, 1313–1326. http://dx.doi.org/10.1109/TNNLS.2012.2200500.
Van Vaerenbergh, S., Santamaria, I., Liu, W., Principe, J.C., 2010. Fixed-budget kernel recursive least-squares. In: 2010 IEEE International Conference on Acoustics Speech and Signal Processing (ICASSP). Presented at the 2010 IEEE International Conference on Acoustics Speech and Signal Processing (ICASSP), pp. 1882–1885. http://dx.doi.org/10.1109/ICASSP.2010.5495350.
Van Vaerenbergh, S., Via, J., Santamana, I., 2006. A sliding-window kernel RLS algorithm and its application to nonlinear channel identification. In: Acoustics, Speech and Signal Processing, 2006. ICASSP 2006 Proceedings. 2006 IEEE International Conference on. pp. V–V.
Vanajakshi, L., Rilett, L.R., 2004. A comparison of the performance of artificial. neural networks and support vector machines for the prediction of traffic speed. In: 2004 IEEE Intelligent Vehicles Symposium. Parma.
Vanajakshi, L., Rilett, L.R., 2007. Support Vector Machine Technique for the Short Term Prediction of Travel Time. In: Proceedings of the 2007 IEEE Intelligent Vehicles Symposium. Istanbul.
Vapnik, V., Lerner, A., 1963. Pattern recognition using generalized portrait method. Autom. Remote Control 24, 774–780.
Vlahogianni, E.I., Golias, J.C., Karlaftis, M.G., 2004. Short-term traffic forecasting: overview of objectives and methods. Transport. Rev. Transnatl. Transdiscipl. J. 24, 533. http://dx.doi.org/10.1080/0144164042000195072.
Vlahogianni, E.I., Karlaftis, M.G., Golias, J.C., 2005. Optimized and meta-optimized neural networks for short-term traffic flow prediction: a genetic approach. Transport. Res. Part C Emerg. Technol. 13, 211–234. http://dx.doi.org/10.1016/j.trc.2005.04.007.
Vlahogianni, E.I., Karlaftis, M.G., Golias, J.C., 2007. Spatio-temporal short-term urban traffic volume forecasting using genetically optimized modular networks. Comput.-Aided Civ. Infrastruct. Eng. 22, 317–325. http://dx.doi.org/10.1111/j.1467-8667.2007.00488.x.
Vlahogianni, E.I., Karlaftis, M.G., Golias, J.C., 2014. Short-term traffic forecasting: where we are and where we're going. Transport. Res. Part C Emerg. Technol 43, 3–19. http://dx.doi.org/10.1016/j.trc.2014.01.005.
Wang, J., Shi, Q., 2013. Short-term traffic speed forecasting hybrid model based on Chaos-Wavelet Analysis-Support Vector Machine theory. Transport. Res. Part C Emerg. Technol. Selected papers from the Seventh Triennial Symposium on Transportation Analysis (TRISTAN VII) 27, 219–232. http://dx.doi.org/10.1016/j.trc.2012.08.004.
Wang, J., Wang, Y., Zhang, C., Du, W., Zhou, C., Liang, Y., 2009. Parameter Selection of Support Vector Regression Based on a Novel Chaotic Immune Algorithm. In: Innovative Computing, Information and Control (ICICIC), 2009 Fourth International Conference on. Presented at the Innovative Computing, Information and Control (ICICIC), 2009 Fourth International Conference on, pp. 652–655. http://dx.doi.org/10.1109/ICICIC.2009.287.
Wang, Y., Wang, J., Du, W., Zhang, C., Zhang, Y., Zhou, C., 2009. Parameters optimization of support vector regression based on immune particle swarm optimization algorithm. In: Proceedings of the First ACM/SIGEVO Summit on Genetic and Evolutionary Computation, GEC '09. ACM, New York, NY, USA, pp. 997–1000. http://dx.doi.org/10.1145/1543834.1543992.

Williams, B.M., Hoel, L.A., 2003. Modeling and forecasting vehicular traffic flow as a seasonal ARIMA process: theoretical basis and empirical results. J. Transport. Eng. ASCE 129, 664–672.
Willmott, C.J., Matsuura, K., 2005. Advantages of the mean absolute error (MAE) over the root mean square error (RMSE) in assessing average model performance. Clim. Res. 30, 79.
Wu, C.-H., Ho, J.-M., Lee, D.T., 2004. Travel-time prediction with support vector regression. IEEE Trans. Intell. Transport. Syst. 5, 276–281. http://dx.doi.org/10.1109/TITS.2004.837813.
Wu, C.-H., Tzeng, G.-H., Lin, R.-H., 2009. A Novel hybrid genetic algorithm for kernel function and parameter optimization in support vector regression. Expert Syst. Appl. Int. J. 36, 4725–4735. http://dx.doi.org/10.1016/j.eswa.2008.06.046.
Xie, Y., Zhao, K., Sun, Y., Chen, D., 2010. Gaussian processes for short-term traffic volume forecasting. Transport. Res. Rec. J. Transport. Res. Board 2165, 69–78. http://dx.doi.org/10.3141/2165-08.
Yin, H., Wong, S.C., Xu, J., Wong, C.K., 2002. Urban traffic flow prediction using a fuzzy-neural approach. Transport. Res. Part C Emerg. Technol. 10, 85–98. http://dx.doi.org/10.1016/S0968-090X(01)00004-3.
Zhang, Y., Xie, Y., 2008. Forecasting of short-term freeway volume with v-support vector machines. Transport. Res. Rec. J. Transport. Res. Board 2024, 92–99. http://dx.doi.org/10.3141/2024-11.
Zheng, L., Yu, M., Yu, S., 2008. Support vector regression and ant colony optimization for combustion performance of boilers. In: Natural Computation, 2008. ICNC '08. Fourth International Conference on. Presented at the Natural Computation, 2008. ICNC '08. Fourth International Conference on, pp. 178–182. http://dx.doi.org/10.1109/ICNC.2008.479.
Zheng, W., Lee, D.H., Shi, Q., 2006. Short-term freeway traffic flow prediction: Bayesian combined neural network approach. J. Transport. Eng. 132, 114.