Fully Bayesian Spatial Temporal Gaussian Process Regression for Air Quality in New South Wales

DARE - Final Project
Robin Aldridge-Sutton & Paco Tseng

May 2021

Abstract

In this project we used Gaussian processes to model air quality in New South Wales over time. Gaussian processes are a family of flexible nonparametric probabilistic models. These Gaussian processes could be applied to the evaluation of acquisition functions for Bayesian optimization of air quality monitoring sites in the future. In this project we implement temporal, spatial, and spatial temporal Gaussian process regression to Nitrous Oxide levels in New South Wales over various time periods as a proof of concept. We implement various inference techniques including type 2 likelihood maximisation, Hamiltonian Markov Chain Monte Carlo, and Variational Bayesian Inference.

Introduction

In this project we used Gaussian processes to model air quality over the New South Wales region over time. We acquired data on the levels of various air pollutants, including Nitrous oxide, Ozone, particulate matter, and others, in the form of hourly and daily averages measured at tens of sites across the region. The sites ranged from Albany in the South West, to Coffs harbour in the North East, with most clustered in the populous regions, especially in Sydney.

Gaussian processes are a family of flexible nonparametric probabilistic models that are well-suited to modelling continuous target variables over continuous covariate domains. They allow straightforward incorporation of prior information with standard Bayesian methods, and produce a full probability distribution for the target variable at any combination of values of the covariates. They are especially well-suited to interpolation and smoothing over noisy observations, where causal inference and extrapolation outside of the range of the data are not the primary objective.

We think that Gaussian processes are useful for modelling air quality over space and time because because the variable domains are continuous, there is likely to be significant prior information that can be incorporated in consultation with domain experts, and the probabilistic models that are produced have many uses. A simple example is inferring the probability that air quality is failing below acceptable levels in a given region over a given period of time.

A more advanced application of Gaussian processes is to the evaluation of acquisition functions for Bayesian optimization. This allows for the inference of optimal locations for the placement of new air quality monitoring sites, either for finding out when and where air quality is best or worst, or for most efficiently reducing overall uncertainty about air quality (gaining the greatest amount of new information) over a given region. It can also be run in reverse, allowing for the inference of existing air quality monitoring sites that contribute the least information to our models, and which may therefore be decommissioned. In this project we implement various Gaussian process models for air quality which may allow for these various applications in the future.

In this project we implement temporal, spatial, and spatial temporal Gaussian process regression to Nitrous Oxide levels in New South Wales over various time periods, as a proof of concept. We present code to process the raw data that is publicly available, as well as a python package for conveniently selecting the desired subset over space and time, fitting models using various inference techniques, and outputting custom-made visualizations of the results. We consider these to form the modelling component of the project corresponding to the requirements for DATA5710 - Applied Statistics for Complex Data.

Inference for Gaussian process models consists of generating appropriate values for certain hyperparameters. The inference techniques in this project are implemented separately from the Gaussian process models themselves, other than simple type 2 likelihood maximisation. The more advanced techniques include Hamiltonian Markov Chain Monte Carlo and Variational Bayesian Inference for sampling from the posterior of the hyperparameters. Gaussian processes are then constructed using posterior mean and maximum a posteriori estimation, and posterior credible intervals are constructed. We consider these to form the inference component of the project corresponding to the requirements for DATA5711 - Bayesian Computational statistics.

The computational code used to conduct the analysis can be made available at this (private) GitHub repository.

Related Work

Gaussian processes became much more popular after [8]. However this text is limited to discussing type 2 likelihood maximisation for hyperparameter tuning. [4] discusses Hamiltonian Markov Chain Monte Carlo, and Neal brought the method into his Bayesian deep learning work from physics in the 1990's. [3] describes variational inference using automatic differentiation, which is the technique that we used.

[6] do similar work to us, implementing HMC, variational mean-field, variational full-rank methods to integrate over the posterior of the hyper-parameters and comparing their performance on four benchmark datasets. Our approach to variational inference is especially motivated by the results shown in this article, as it indicated that mean-field variational inference does surprisingly well at significantly lower computational cost. Mean-field variational inference treats the hyperparameters as independent from each other, which is generally unlikely to be true, but means that the full covariance matrix does not have to be computed, at quadratic computational cost.

Our inference work relied on the python packages [2], for automatic differentiation, and [5], for HMC for Gaussian processes. We used HMC as a benchmark to compare with our own implementation of variational inference. Our visualisation our model outputs relied heavily on [1].

[7] discusses air quality monitoring in New South Wales and may be a useful first source of domain knowledge in future work, but did not strongly influence this project.

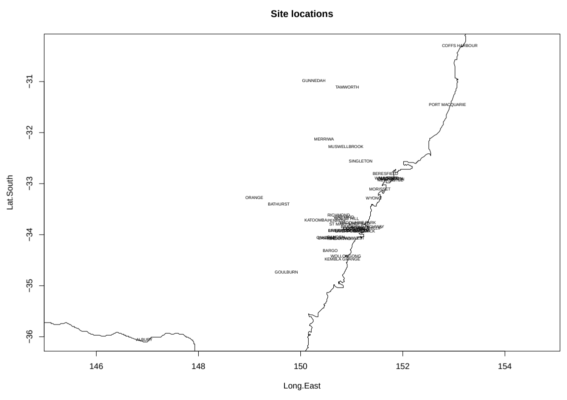
Methods

Data acquisition and processing

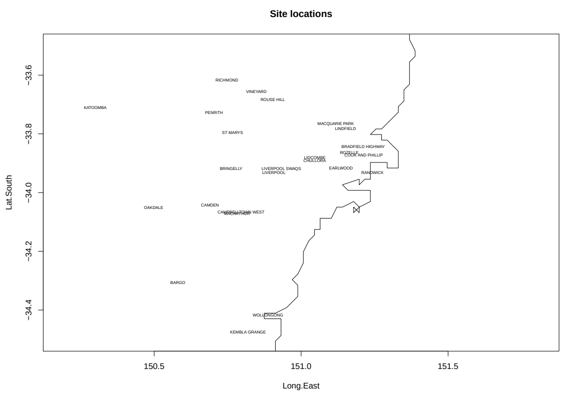
We acquired our data through the New South Wales Department of Planning, Industry and Environment, which provides a data download facility at https://www.dpie.nsw.gov.au/air-quality/air-quality-data-services/data-download-facility. We downloaded two data sets to explore, one with hourly average, and one with daily average observations of air quality and meteorological variables. In both cases we selected all available variables and measurement sites.

We converted the files to .csv and read them into R. The data was originally in a form that was difficult to work with, with separate columns for each variable at each site, so we wrote R code to transform the data. This included tidying up and abbreviating column names, selecting the data from each site for one variable at one time or date, combining the data from all sites for all variables at one time, and stacking these tables to get one table over all times. The time and date variables were converted to simple numeric values. We then attached the latitude, longitude, and altitude data for each site. The final data frame has one column for each variable, including the air quality and meteorological variables, the site name and location, and the time or date when the measurement was recorded. The data in this form was written to a new .csv file for each data set.

We then wrote R code to perform a simple exploratory data analysis on the transformed data. This included plotting the names of the sites where the variables were observed on a map of New South Wales (figures (1a), and (1b)), checking the proportions of missing data for each variable and site, and plotting the observations of individual variables over all times or dates for all sites. We started with the hourly data set and noticed that the data for many sites was not included, which is why we tried getting the daily data set, which turned out to include many more sites. We found that the air quality variables with the least missing data were Nitric oxide and Nitrous oxide. These were unsurprisingly highly correlated, but the Nitric oxide data was quite often zero, so we decided to focus on the daily average Nitrous oxide data (figure (2)). There is an apparent outlier, the Bradfield Highway site, which is not surprising as Nitrous oxide is a component of car exhaust gases.



(a) Map of all sites



(b) Map of greater Sydney sites

Figure 1

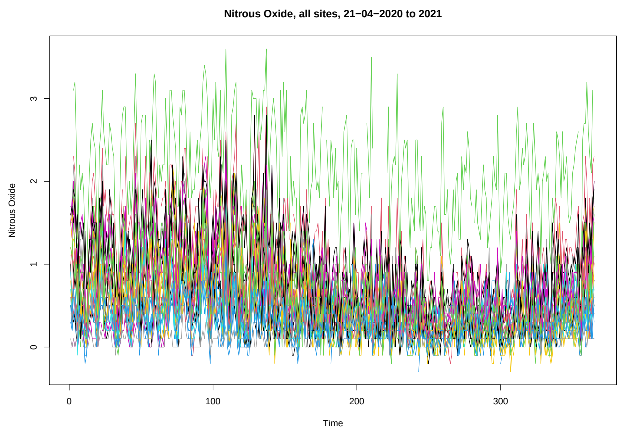


Figure 2: Nitrous Oxide levels over time for all sites

Fully Bayesian Gaussian Process regression

Our data is air quality measurement over New South Wales. In particular, NO_2 daily measurement from April 2020 - April 2021. Consider our observations in format of

$$(X^{(t)}, y^{(t)}) = \left\{ \boldsymbol{x}_i^{(t)}, y_i^{(t)} \right\}_{i=1}^{n}, \; t = 1, \cdots, T$$

where $y_i^{(t)}$ is the Nitrate dioxide (NO_2) measurement on day t and at location $\boldsymbol{x}_i^{(t)}$. It is common practice to assume that the measurements are corrupted with some noise, which the noise is assumed to be normally distributed.

$$y_i^{(t)} = f_i^{(t)} + \epsilon_i^{(t)}, \; \epsilon_i^{(t)} \overset{\text{iid}}{\sim} N(0, \sigma_n^2).$$

At this point, it is clear the dataset of New South Wales air quality across April 2020 - April 2021 is a spatial temporal process. Our project aims to inference such process by assuming a Gaussian Process prior of the latent function f with positive definite kernel function $k_{\boldsymbol{\theta}}\left(\boldsymbol{x}_i^{(t)}, \boldsymbol{x}_j^{(\tau)}\right)$, $\boldsymbol{\theta}$ denotes the hyper-parameters that parameterised the kernel. To simplify our notations, the inputs X is considered to include both spatial and temporal inputs. The overall Gaussian Process hierarchical model is structured as,

$$\boldsymbol{\theta} \sim p(\boldsymbol{\theta}) \tag{1}$$

$$f|X, \boldsymbol{\theta} \sim \mathrm{GP}(\mathbf{0}, K_{\boldsymbol{\theta}}) \tag{2}$$

$$\boldsymbol{y}|f \sim \mathrm{GP}(f, \sigma_n^2 I) \tag{3}$$

In our case, we set the prior distribution of the hyper-parameters to be exponential distribution with rate 1 as this ensures hyper-parameters values to be positive. The covariance function is set to be the product of covariance function defined over spatial values and covariance function defined over temporal values, where both are squared exponential covariance function.

$$k_{\boldsymbol{\theta}}\left(\boldsymbol{x}_i^{(t)}, \boldsymbol{x}_j^{(\tau)}\right) = \sigma_{\mathrm{k}}^2 \exp\left\{-\frac{1}{2}\left(\frac{(x_{i1}-x_{j1})^2}{l_1^2} + \frac{(x_{i2}-x_{j2})^2}{l_2^2} + \frac{(t-\tau)^2}{l_T^2}\right)\right\} \tag{4}$$

Based on the hierarchical model, $\boldsymbol{\theta} = (l_1, l_2, l_T, \sigma_n, \sigma_k)$ and the posterior joint distribution of the latent function f and unknowns $\boldsymbol{\theta}$ is given by,

$$p(f, \boldsymbol{\theta}) \propto p(\boldsymbol{y}|f)p(f|X, \boldsymbol{\theta})p(\boldsymbol{\theta})$$

and the predictive distribution over unobserved space X^* (will suppress both unobserved inputs X^* and observed inputs X in the conditional distribution for simplicity) is given by,

$$p(f^*|\boldsymbol{y}) = \int\int \underbrace{p(f^*|f, \boldsymbol{y}, \boldsymbol{\theta})p(f|\boldsymbol{\theta}, \boldsymbol{y})}_{=p(f^*|\boldsymbol{y},\boldsymbol{\theta})} p(\boldsymbol{\theta}|\boldsymbol{y}) df d\boldsymbol{\theta}. \tag{5}$$

The underbraced inner integral integrate to regular Gaussian process predictive posterior distribution given fixed hyper-parameters $\boldsymbol{\theta}$, i.e

$$p(f^*|\boldsymbol{y}, \boldsymbol{\theta}) = \mathrm{GP}(\boldsymbol{\mu}^*, \Sigma^*) \tag{6}$$

where $\boldsymbol{\mu} = K_{\boldsymbol{\theta}}^*(K_\theta + \sigma_n^2 I)^{-1}\boldsymbol{y}$ and $\Sigma* = K_{\boldsymbol{\theta}}^{**} - K_{\boldsymbol{\theta}}^*(K_\theta + \sigma_n^2 I)^{-1}K_{\boldsymbol{\theta}}^{*T}$, and K, K^*, K^{**} denotes the covariance function evaluated between inputs X, X and X^* and $X*$ respectively. For us to conduct full bayesian inference on the resulting process, we will need to numerically integrate (5) using (6). In particular, the fully bayesian Gaussian process draws samples of hyper-parameters $\boldsymbol{\theta}_j, j = 1, \cdots, M$ from posterior distribution $p(\boldsymbol{\theta}|\boldsymbol{y})$. We obtain

$$p(f^*|\boldsymbol{y}) \approx \frac{1}{M}\sum_{j=1}^{M} p(f^*|\boldsymbol{y}, \boldsymbol{\theta}_j),\ \boldsymbol{\theta_j} \sim p(\boldsymbol{\theta}|\boldsymbol{y}) \tag{7}$$

Hamiltonian Monte Carlo

Monte Carlo Markov Chain is considered the gold standard for fully bayesian GPregression inference as the sampling follows the hyper-parameters posterior distribution approximately. After some literature revision [6], HMC is considered the standard practice as it suppresses the random walk behaviour typical of Metropolis and its variants [4]. In our analysis, we follow the setup of `GPflow` [5], which is a python package for constructing Gaussian process models using `TensorFlow`. The package relies on `TensoFlow` for faster computation and handles all gradient computation. To initialise the sampling process, the initial values of hyper-parameters $\boldsymbol{\theta}_0$ are taken as the optimal solution which minimised the training loss. The prior distribution of the hyper-parameters $\boldsymbol{\theta}$ is set to be exponential distribution with rate $\lambda = 1$, leap frog step is set to be 10 and step size 0.01. The sampling chain is set to have 10000 sampled values and 200 steps of burn-ins. Furthermore, all variables sampling are unconstrained in the process as this this makes the optimisation algorithm in the gradient step much more efficient. In the end of the process, the function of `SamplingHelper` convert the values to the constrained space, i.e the exponential transform. This allows us to avoid the positivity requirements for some of the hyper-parameters.

Variational Inference

Our approach to VI closely follow the set up in *Automatics Differentiation Variational Inference* [3]. Different to the HMC setup, we dont transform the support of hyper-parameters to ensure positivity but relies on positive random variables distribution for the prior and variational distributions. In particular, the prior are set to be log-Normal distribution $\mathcal{LN}(0, 0.1)$ for all hyper-parameters and we considered the mean-field variational of the same family as the prior.

$$p(\boldsymbol{\theta}|\boldsymbol{y}) \approx q_{\mathrm{mf}}(\boldsymbol{\theta}) = \prod_{j=1}^{h} \mathcal{LN}\left(\theta_j; \mu_j, \sigma_j^2\right) \tag{8}$$

where h is the number of hyper-parameters. The variational objective, ELBO is maximised using `PyTorch` [2], however the optimiser algorithm is designed for minimisation, hence the negative of the ELBO is taken as the objective function.

$$\mathcal{L}(\boldsymbol{\theta}) = \mathbb{E}\left[\ln\left(q_{\mathrm{mf}}(\boldsymbol{\theta})\right) - \ln p(\boldsymbol{\theta}) - \ln\left(p(\boldsymbol{y}; \boldsymbol{\theta})\right)\right] \tag{9}$$

$$\boldsymbol{\theta}^* = \arg\max_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}) \tag{10}$$

Implementation of Spatial Temporal Gaussian Process regression

We wrote python code to implement spatial temporal Gaussian process regression for our data. We started with code provided by Roman Marchant in his tutorial on this topic but altered and extended it significantly.

The code is now divided into four classes contained in four separate python files, GP, TGP, SGP, and STGP. These correspond to a Gaussian process superclass, Temporal and Spatial Gaussian process subclasses which inherit from it, and a Spatial Temporal Gaussian process subclass which inherits from the Spatial Gaussian process class. Class inheritance in python allows code that applies to multiple kinds of objects to be shared among them without writing it again, which makes it much easier to develop and maintain.

The Gaussian Process superclass

The GP superclass defines core variables and functions that are shared among all the temporal and spatial subclasses. This class is not necessarily ever seen by someone who just wants to use the code to perform Gaussian process regression. Rather, they will use the code for temporal and or spatial regression, and this will call the superclass code in the background when necessary.

When the user wants to use the code they create an object of the kind of class they are interested in, passing it the required data and parameters. Some of those inputs are passed to the GP class constructor. The GP constructor takes in a data frame in the form produced by our R code for the daily average data, the variable in that data frame that regression will be applied to, the name that will be used to refer to that variable in plots and outputs, and the name of the covariance function that will be used.

The GP code calls the select function for the subclass, which selects the data based on the site, latitude, longitude, and/or dates requested by the user. It then centers and scales the data to have zero mean, and unit variance, which makes it easier for numerical optimisation to find appropriate values of the covariance hyperparameters. It automatically selects initial hyperparameter values according to some simple heuristics. The function and noise standard deviations are initialised to account for 90% and 10% of the variance of the target variable respectively. The length scale parameters are initialised depending on the range of the covariates and the number of data points. These initial values can be used to quickly try to roughly apply the regression, or as starting values for type 2 likelihood maximisation, or they can be reset manually by the user. Finally the GP code calculates the covariance of the data in preparation for predicting new values or calculating the log marginal likelihood for hyperparameter tuning.

The GP code defines several functions that are useful to all of the spatial and temporal subclasses. These include functions to center and scale data, calculate covariance matrices based on the Squared Exponential (SE) and Automatic Relevance Determination (ARD) kernels (ARD is SE with separate length scale parameters for each covariate), predict the values of the target variable at a given set of covariate values, load posterior hyperparameter samples produced by Markov Chain Monte Carlo or Variational Inference methods, find the posterior mean or mode and set the hyperparameters to those values, sample from the posterior predictive distribution, find the standard deviations for each value given their covariance matrix, find and print the proportion of the data covered by a given interval, calculate the log marginal likelihood, and perform type 2 likelihood maximisation for the hyperparameters.

The Spatial Temporal subclasses

The spatial temporal subclasses inherit the core Gaussian process features from the Gaussian process superclass, but add features that are different depending on which kind of regression is being performed. These include how the data is selected, how the hyperparameters are initialised, how the covariance matrices are computed, and how predictions are plotted.

The python notebooks Temporal regression, Spatial regression, and Spatial temporal regression demonstrate how to use the spatial temporal classes to perform Gaussian process regression. In each case they follow the process of importing the required packages, loading the data, setting the necessary parameters, creating an object of the relevant class, performing various kinds of regression, and plotting the results.

Gaussian process regression comes down to finding appropriate values of the hyperparameters in the covariance function. The kinds of regression supported in these classes are type 2 likelihood maximisation, estimation by the posterior mean and mode (maximum a posteriori), and posterior credible interval formation.

In the first case the optimisation is performed within the GP class itself, when the fit function is called. Sometimes passing bounds for each of the hyperparameters can help the optimisation find more appropriate values. In the latter three cases of hyperparameter tuning samples from the posterior should first be loaded into the relevant Gaussian process object using the load_hp_samps function. The regression is then performed by calling the set_post_mean, set_max_post, and plot_post functions.

Once the regression has been performed the results can be evaluated in terms of an interval over the data. For the first three kinds of regression this can be performed by calling the plot_preds function. This does two things. First it evaluates a confidence interval at the covariate values of each data point, and prints the proportion of data points for which the value of the target variable lies within the interval. Second it evaluates a confidence interval at a grid of covariate values that covers the values in the data, and plots those confidence intervals along with the mean estimates. In the case of posterior credible interval formation, the interval and mean itself constitutes the regression, and the interval coverage is printed and the plot is displayed when the intervals are computed.

Spatial Temporal Plots

We did a lot of work to develop effective plots for spatial and spatial temporal regression. Sometimes people use heat maps or contour plots, but it is not always easy to see how the results relate to the data in these kinds of plots. Instead we developed our own form of plots using the sophisticated visualization library Plotly [1]. These plots have longitude and latitude on the x and y axes, and the target variable on the z axis. The data is included as a 3D scatterplot, with the names of the sites next to each point. Their locations are also projected onto the plane where z is equal to zero, and the New South Wales coastline is indicated there by a line interpolating Wollongong, Newcastle, Port Macquarie, and Coffs Harbour, with the sea indicated in blue to the east.

The regression mean is indicated by a transparent red surface, and the upper and lower interval bounds are indicated by transparent grey surfaces. The plots are interactive, allowing the user to rotate and zoom in and out to inspect the regression surfaces and see how they relate to the data. In the case of spatial temporal regression one spatial plot is generated for each time point, and these plots are joined together in an animation that plays through time. The animation can be paused and the spatial plot examined interactively at any point. These interactive and animated plots are accessible in the HTML files Temporal, Spatial, and Spatial Temporal Regression, in the GPcode folder included with this report. They form a significant part of this project but are unable to be included in this PDF document. We strongly encourage the reader to download the HTML files here, open them in their web browser, play the animations, and exploit the interactive plots to evaluate the results. Static images of a few examples are included in the results section below.

We think that these plots make it much easier to evaluate how well the regression is fitting to the data, which is especially important in this non-parametric setting where incorporating prior beliefs about the smoothness of the underlying function, and the relative variability of the function and the noise in the observations are essential to the process. For example it is relatively easy to see that the air quality in terms of Nitrous Oxide concentration tends to become both worse and more variable in more built up areas, that the overall levels vary significantly from day to day, and that the peaks shift spatially, particularly from east to west, presumably due to variation in wind strength and direction over consecutive days. It is also possible to check that the regression mean is more or less running through the center of the data, that the regression interval is more narrow where there is more and less variable data, and that the data points that are excluded are generally more extreme for their space and time. What is more difficult without domain expertise is making judgments about how much measurement noise to expect, and how much to expect true air quality to vary over the relevant space and time scales.

Results

In our experiments, we applied our Gaussian process package to temporal, spatial, and spatial temporal datasets. We tried to choose datasets which had less missing data, and which were not too large to handle in a reasonable amount of time. In the temporal case we selected data for the Liverpool site from 20 April 2020 to 20 April 2021. In the spatial case we selected data from all available sites on 1 August 2020. In the spatial temporal case we selected data from all available sites from 10 January 2021 to 20 April 2021.

Below we include the HMC trace plots, posterior distributions of the hyperparameters obtained by HMC and VI, and an example of a regression plot produced by one of the inference techniques. The full set of experiments are included in python notebooks to be conveniently reproduced. Viewing the plots may require installation of some additional software extensions so the results have been exported to `HTML` files which can be downloaded here and simply viewed in a web browser.

Temporal GP: Liverpool

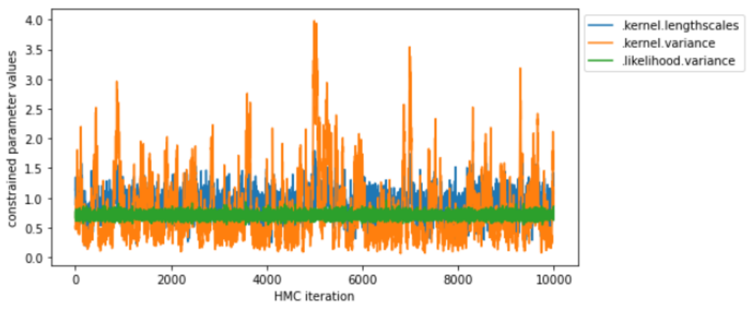
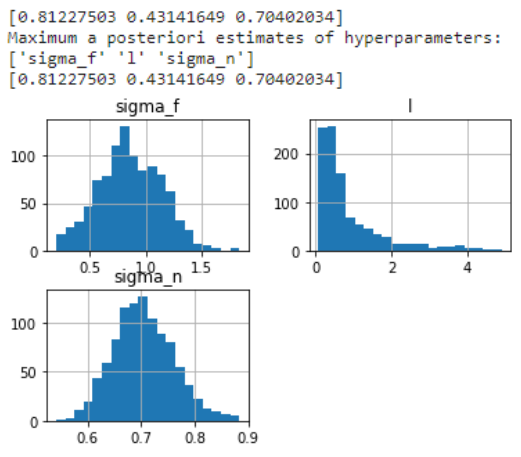
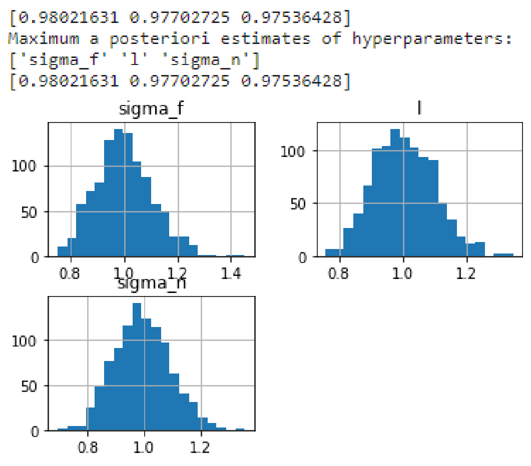


Figure 3: The HMC chain show the sampled values of length scales, kernel variance, and noise variance



```
[0.81227503 0.43141649 0.70402034]
Maximum a posteriori estimates of hyperparameters:
['sigma_f' 'l' 'sigma_n']
[0.81227503 0.43141649 0.70402034]
```

(a) HMC sampled values histograms, kernel variance, length scale, and noise variance



```
[0.98021631 0.97702725 0.97536428]
Maximum a posteriori estimates of hyperparameters:
['sigma_f' 'l' 'sigma_n']
[0.98021631 0.97702725 0.97536428]
```

(b) VI samples value of kernel variance, length scale, and noise variance

Figure 4: Sampled hyper-parameter values for temporal Gaussian process

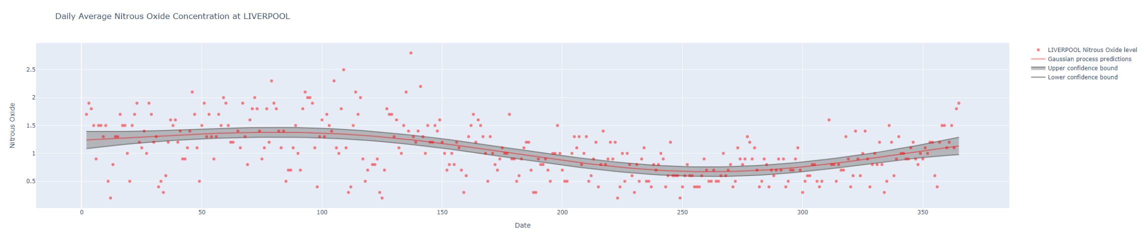


Figure 5: GP regression fitted using Type II Likelihood maximisation

Spatial GP: 01/08/2021 on all sites

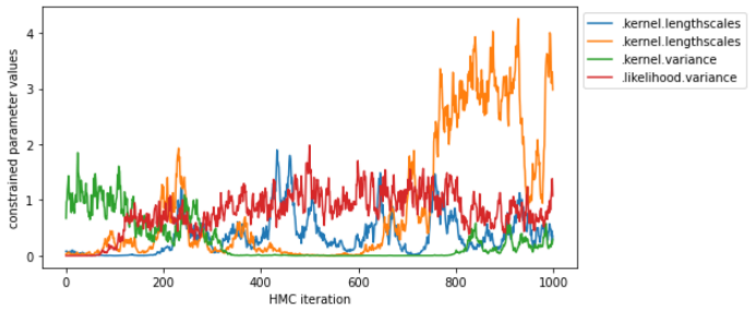
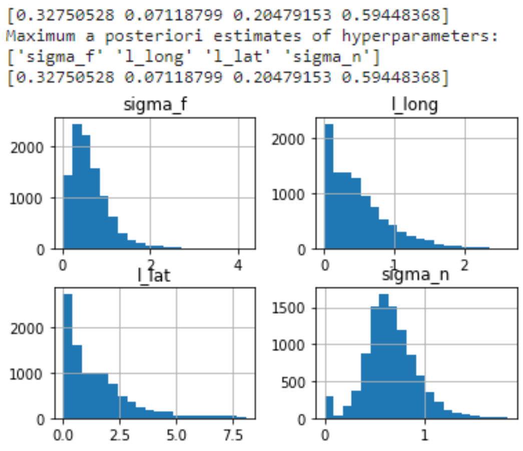
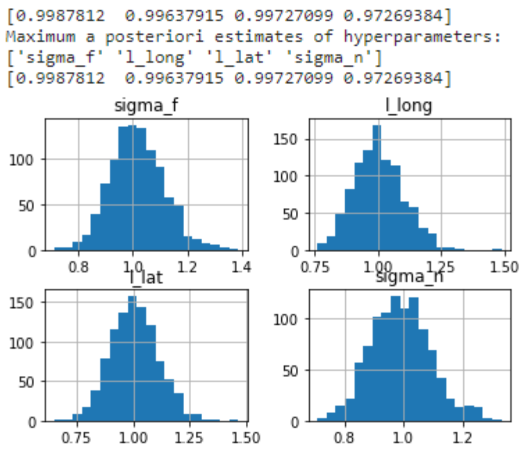


Figure 6: The HMC chain show the sampled values of length scales on longitude and latitude, kernel variance, and noise variance



```
[0.32750528 0.07118799 0.20479153 0.59448368]
Maximum a posteriori estimates of hyperparameters:
['sigma_f' 'l_long' 'l_lat' 'sigma_n']
[0.32750528 0.07118799 0.20479153 0.59448368]
```

(a) HMC sampled values histograms, kernel variance, length scale on longitude and latitude, and noise variance



```
[0.9987812  0.99637915 0.99727099 0.97269384]
Maximum a posteriori estimates of hyperparameters:
['sigma_f' 'l_long' 'l_lat' 'sigma_n']
[0.9987812  0.99637915 0.99727099 0.97269384]
```

(b) VI samples value of kernel variance, length scale on longitude and latitude, and noise variance

Figure 7: Sampled hyper-parameter values for temporal Gaussian process

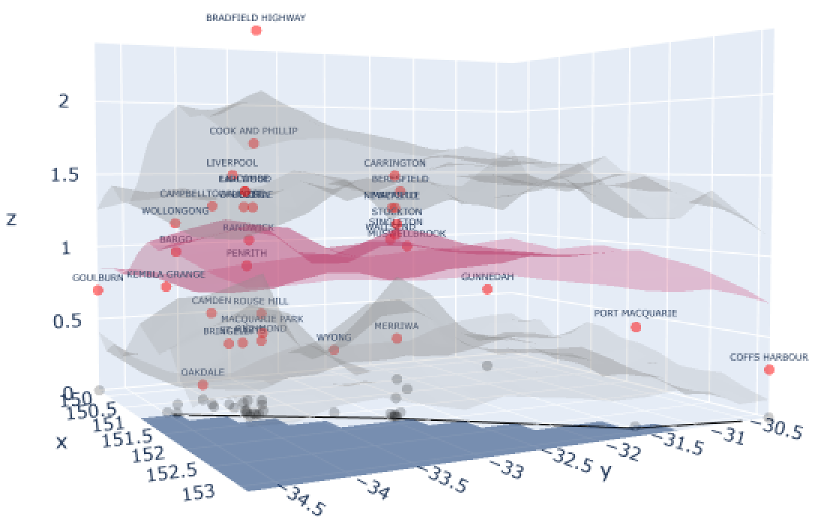


Figure 8: GP regression fitted using HMC for hyper-parameters integration

Spatial Temporal: 100 days on all sites

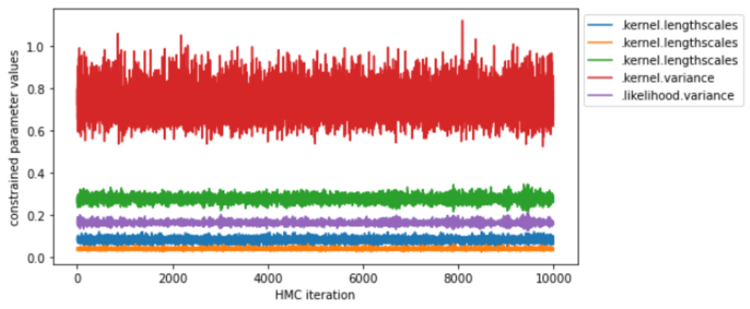
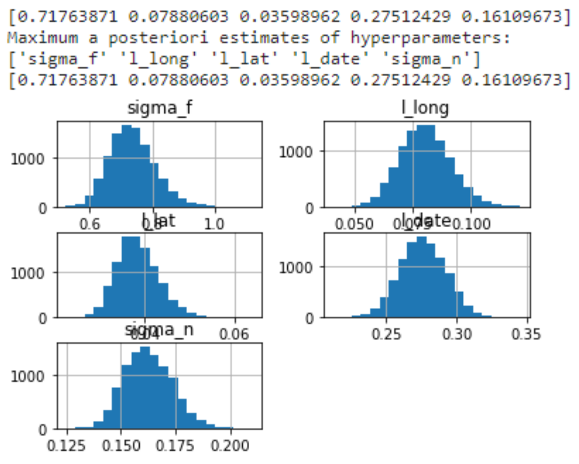
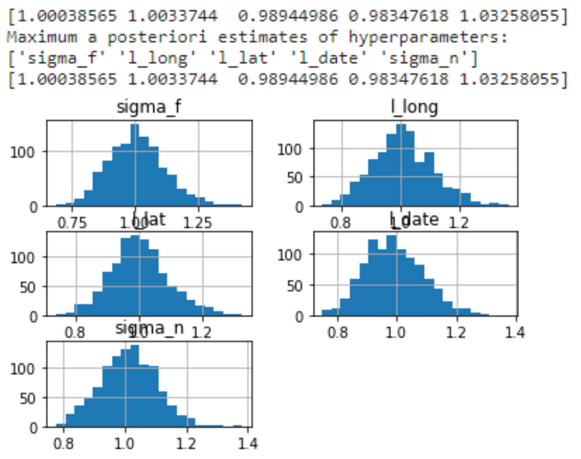


Figure 9: The HMC chain shows the sampled values of length scales on longitude, latitude, and time, kernel variance, and noise variance



```
[0.71763871 0.07880603 0.03598962 0.27512429 0.16109673]
Maximum a posteriori estimates of hyperparameters:
['sigma_f' 'l_long' 'l_lat' 'l_date' 'sigma_n']
[0.71763871 0.07880603 0.03598962 0.27512429 0.16109673]
```

(a) HMC sampled values histograms, kernel variance, length scale on longitude, latitude, and time, and noise variance



```
[1.00038565 1.0033744  0.98944986 0.98347618 1.03258055]
Maximum a posteriori estimates of hyperparameters:
['sigma_f' 'l_long' 'l_lat' 'l_date' 'sigma_n']
[1.00038565 1.0033744  0.98944986 0.98347618 1.03258055]
```

(b) VI samples value of kernel variance, length scale on longitude, latitude, and time, and noise variance

Figure 10: Sampled hyper-parameter values for temporal Gaussian process

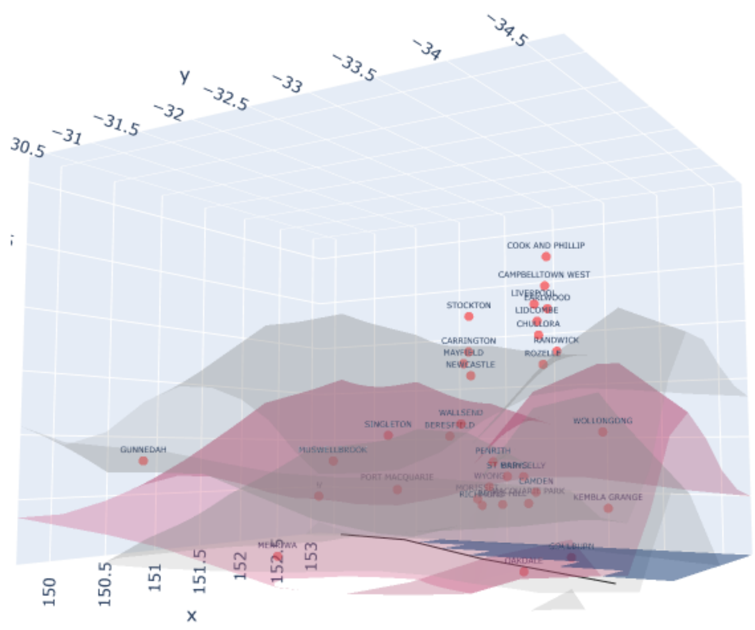


Figure 11: GP regression fitted using variational inference for hyper-parameters integration

In the temporal case all of the inference techniques performed very similarly. In all spatial and temporal cases the posterior credible intervals were less smooth than the others as they were based on just 500 samples from the GP given by posterior samples of the hyperparameters. We limited the number of samples because it took a long time, and we did not implement it at all in the spatial temporal case for the same reason.

In the spatial and spatial temporal cases we implemented two different covariance kernels, Squared Exponential and Automatic Relevance Detection (SE with separate length scales for each dimension). This had the predictable effect that the regression surfaces tended to be more nonlinear and showed different levels of variation in one spatial dimension versus the other. The HMC and VI algorithms were always implemented using the ARD covariance kernel.

The spatial and spatial temporal cases exhibited some variation in the way the GP fitted the data. The biggest difference was that the plots would sometimes become 'spiky', usually with the regression surface also quickly moving to the mean of the data between data points, probably indicating small length scales and relatively large noise variance estimates.

This occurred for example in the HMC MAP estimates for the spatial case. We tend to think that represents a poor fit. The trace plots suggested that the HMC chain converged in the temporal and spatial temporal cases, but in the spatial case it is not clear whether the samples for the length scale parameter had converged.

Discussion

As mentioned previously it is difficult to make strong judgements about the form of the underlying true value of Nitrous oxide levels without consulting a domain expert. We often found that the 95% confidence intervals covered 99 or 100% of the data, which suggests that the models may not have been a good fit. We probably should have tried fitting the GP to the log of the data, since Nitrous oxide measurements are bounded at zero, and exhibit greater variance at higher values. It would also be good to compare the root mean square error between the predictions and the data for each method, as in [6].

Conclusion

We have demonstrated that it is possible to model air quality in New South Wales with Gaussian process regression over space and time. We have written code implementing our work and it is easy to update and extend so it may be useful in future efforts to extend this project. This might include comparing performance when applying the regression to the log of the target variable since it is bounded at zero and is more variable at greater values. We hope that we have demonstrated our ability to implement these models in a variety of cases, and to apply them to real-world data, using a variety of inference techniques.

References

[1] Plotly Technologies Inc. “Collaborative data science” *Plotly Technologies Inc.*, `https://plot.ly`

[2] Paszke, Adam and Gross, Sam and Massa, Francisco and Lerer, Adam and Bradbury, James and Chanan, Gregory and Killeen, Trevor and Lin, Zeming and Gimelshein, Natalia and Antiga, Luca and Desmaison, Alban and Kopf, Andreas and Yang, Edward and DeVito, Zachary and Raison, Martin and Tejani, Alykhan and Chilamkurthy, Sasank and Steiner, Benoit and Fang, Lu and Bai, Junjie and Chintala, Soumith “PyTorch: An Imperative Style, High-Performance Deep Learning Library”, *Advances in Neural Information Processing Systems 32*, Curran Associates, Inc., pp.8024–8035, 2019, `http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf`

[3] Kucukelbir, Alp and Tran, Dustin and Ranganath, Rajesh and Gelman, Andrew and Blei, David M “Automatic differentiation variational inference” *The Journal of Machine Learning Research*,JMLR.org, Vol. 18, No.1, pp.430–474, 2017

[4] Radfor M Neal et al “MCMC using hamiltonian dynamics” *Handbook of markov chain monte carlo*, 2(11):2, 2011

[5] Matthews, Alexander G de G and Van Der Wilk, Mark and Nickson, Tom and Fujii, Keisuke and Boukouvalas, Alexis and León-Villagrá, Pablo and Ghahramani, Zoubin and Hensman, James “GPflow: A Gaussian Process Library using TensorFlow”, *J. Mach. Learn. Res.*, Vol. 18, No. 40, pp.1–6, 2017

[6] Lalchand, Vidhi and Rasmussen, Carl Edward “Approximate inference for fully Bayesian Gaussian process regression”, *Symposium on Advances in Approximate Bayesian Inference*, pp.1–12, 2020,PMLR

[7] Riley M, Kirkwood J, Jiang N, Ross G. and Scorgie Y. “Air quality monitoring in NSW: From long term trend monitoring to integrated urban services”, *Air Quality & Climate Change*, Vol. 54, No. 1, pp.44–51, 2020.

[8] Rasmussen, C. E. and Williams, C. K.I. “Gaussian processes for machine learning”, *MIT press*, 2003