

## Coursework 4M24: High-Dimensional MCMC

The performance of MCMC algorithms for high-dimensional problems is analysed in this coursework. The methods seen in the lecture notes (Gaussian random walk Metropolis Hastings (GRW-MH) and the preconditioned Crank-Nicolson (pCN) method) are compared, when sampling latent variables on a 2D domain. Code for the coursework supplied in `cw_functions.py` contains useful functions for generating matrices, plotting, and skeleton code to be completed. The files `cw_simulation.py` and `cw_spatial.py` contain the code to set up the problem and relevant plotting examples. This report should not exceed 10 pages, including figures and appendices. It should contain the answers and plots relating to questions (a)-(f), along with any extensions.

## Part 1 - Simulation

A Gaussian Process (GP) is defined on a 2D domain  $x_1, x_2 \in [0, 1]$ , and the realisations from this process are defined as  $\mathbf{u} \sim \mathcal{N}(0, C)$ , where  $C_{i,j} = k(\mathbf{x}_i, \mathbf{x}_j)$  and  $k$  is a squared exponential covariance function parametrised by a length-scale  $l$ .

$$k(\mathbf{x}, \mathbf{x}') = \exp\left(-\frac{\|\mathbf{x} - \mathbf{x}'\|^2}{2l^2}\right)$$

The coordinates of the latent variables  $\{\mathbf{x}_i\}_{i=1}^N$  define the GP prior, and should be placed on an  $D \times D$  grid. The observation model is defined as:

$$\mathbf{v} = G\mathbf{u} + \epsilon$$

Where the matrix  $G$  is generated by a random subsample of the latent variable coordinate locations - the function `get_G` generates this matrix from the subsampling indexes given by the function `subsample`. The observation noise is given by  $\epsilon \sim \mathcal{N}(0, I)$ .

- (a) Sample from the GP prior, and then the observation noise to simulate the data  $\mathbf{v}$  - set the parameters  $N = 16, l = 0.3$ , `subsample_factor` = 4. Use the function `plot_3D`, comment on the effect of varying the length-scale parameter on the GP surface. Visualise the prior sample with the data overlaid using `plot_result`.
- (b) Complete the functions `log_prior` and `log_continuous_likelihood`, which should return  $\ln p(\mathbf{u})$  and  $\ln p(\mathbf{v}|\mathbf{u})$  respectively. Complete the TODO sections in the skeleton functions `grw` and `pcn`, which are the GRW-MH, and pCN algorithms - see Lecture 5/6 for details. Sample from  $p(\mathbf{u}|\mathbf{v})$  using both these algorithms (use  $n = 10000, \beta = 0.2$ ), and plot the mean of the inferred  $\mathbf{u}$  alongside the data. Visualise the absolute error field between the original  $\mathbf{u}$  and the inferred  $\mathbf{u}$ , comparing the performance of the two algorithms. Comment on the GRW-MH and pCN acceptance rate for low and high-dimensional latent variable spaces (try  $D = 4$  and  $D = 16$ ). Vary the parameter  $\beta$  between 0 and 1 - comment on the effect on the acceptance rate. What adverse affect would a very small  $\beta$  have on the samples?

*Extension: Demonstrate the robustness of the pCN method to 'mesh refinement'<sup>1</sup> by increasing  $N$  for a given dataset, and comparing the acceptance rate to the GRW.*

Now the model is extended to work on a probit classification problem. The data  $\mathbf{v}$  is put through a probit transform to give the vector  $\mathbf{t}$ , where the  $i^{\text{th}}$  component is given below. This gives the following form of the likelihood, where  $\Phi$  is the standard normal CDF.

$$t_i = \text{sign}(v_i), \quad p(t_i = 1|\mathbf{u}) = \Phi([G\mathbf{u}]_i)$$

- (c) Derive the full log-likelihood for the probit classification problem, and use this to complete the function `log_probit_likelihood` to return  $p(\mathbf{t}|\mathbf{u})$ . Generate samples from the posterior  $p(\mathbf{u}|\mathbf{t})$  using the pCN method. Predict the true class assignments  $\mathbf{t}_{\text{true}} = \text{probit}(\mathbf{u})$ , using a Monte-Carlo (MC) estimate of the predictive distribution (Complete the function `predict_t`). Visualise these simply using the `plot_2D` function, comparing the true assignments to the predictive probabilities.

*Hint: Use the samples from the posterior to form a MC estimate of the posterior predictive probability (Shown below).*

$$p(t^* = 1|\mathbf{t}) = \int p(t^* = 1, \mathbf{u}|\mathbf{t})d\mathbf{u} = \int p(t^* = 1|\mathbf{u})p(\mathbf{u}|\mathbf{t})d\mathbf{u}$$

<sup>1</sup>See Fig 1 from Cotter's paper - <https://arxiv.org/abs/1202.0709>

- (d) Now make hard assignments by thresholding the predicted class assignments at  $p(t^* = 1|\mathbf{t}) = 0.5$ . This can be compared to the true class assignments to give a mean average prediction error per cell. Up until now, the length-scale parameter has been given to the model - you will now attempt to find the parameter through optimisation. Minimise the prediction error with respect to the length-scale parameter, by performing a line search between  $l = 0.01, l = 10$  and plotting the error. Compare this to the true value of  $l$  used to generate the data - can the length scale be inferred?

*Extension: How else could you infer this parameter? How could you improve on this simple point estimate?*

## Part 2 - Spatial Data

This section looks at predicting the number of bike thefts in Lewisham borough (Figure 1a <sup>2</sup>) over 2015. The dataset given in `cw_data.csv` contains a lists of  $x, y$  locations of  $400m^2$  cells and the corresponding number of bike thefts in that location in 2015. The task is to subsample this count data, then infer the field  $\mathbf{u}$  on the original coordinates given this data, and evaluate performance based on predictions of the original data.

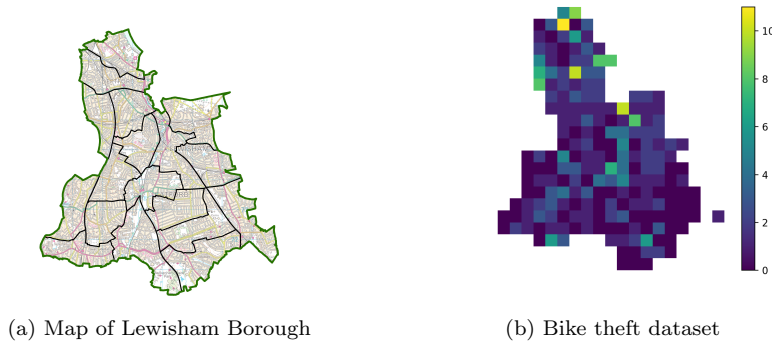


Figure 1

The likelihood needed here is the Poisson likelihood. The field  $\mathbf{u}$  is mapped to  $\boldsymbol{\theta} = \exp(\mathbf{u})$  for positivity, and  $\boldsymbol{\theta}$  is used as the Poisson rate for each data point location. The overall likelihood for the data counts  $\mathbf{c}$  is given by:

$$p(\mathbf{c}|\boldsymbol{\theta}) = \prod_{i=1}^N f(c_i|\theta_i) = \prod_{i=1}^N \frac{e^{-\theta_i} \theta_i^{c_i}}{c_i!}$$

- (e) Derive the full Poisson log-likelihood, and use this to complete the function `log_poisson_likelihood` to return  $p(\mathbf{c}|\mathbf{u})$ . Generate samples from the posterior  $p(\mathbf{u}|\mathbf{c})$  using the pCN method.

The inferred expected counts are given by:

$$\mathbb{E}_{p(c^*|\mathbf{c})} [c^*] = \sum_{k=0}^{\infty} k p(c^* = k|\mathbf{c})$$

We have samples from  $p(\mathbf{u}|\mathbf{c})$ , so using:

$$p(c^* = k|\mathbf{c}) = \int p(c^* = k, \mathbf{u}|\mathbf{c}) d\mathbf{u} = \int p(c^* = k|\mathbf{u}) p(\mathbf{u}|\mathbf{c}) d\mathbf{u} \quad (1)$$

- (f) Show that the expectation can be approximated using the MC estimate below, using the fact that  $\mathbb{E}[f(c_i|\theta_i)] = \theta_i$ :

$$\mathbb{E}_{p(c^*|\mathbf{c})} [c^*] \simeq \frac{1}{N} \sum_{i=1}^N \theta^{(i)}$$

Use this to infer the expected counts and compare these to the bike theft counts from the original dataset. Visualise four plots - the original count data, the subsampled data, the inferred counts, and the error field. Run the model using extreme length-scale parameter values (low and high), and comment on the predicted counts. Suggest a reasonable length-scale value.

<sup>2</sup>[https://www.lgbce.org.uk/\\_flysystem/s3/2018-07/lewisham\\_0.jpg](https://www.lgbce.org.uk/_flysystem/s3/2018-07/lewisham_0.jpg)