Project 1: Discrete GMRFs and image restoration

1 Introduction

The space probe Rosetta and its lander module Philae is currently performing a detailed study of comet 67P/Churyumov-Gerasimenko. On 12 November last year, the mission received much attention when it performed the first soft landing on the comet and returned data from the surface. Fortunately, the equipment on the space probe worked well, and Rosetta has returned beautiful images of the comet. However, imagine a situation where there was something wrong with the camera equipment or the data transfer from a space probe, so that we only obtain partial or corrupted information. This scenario is not that unlikely: The Huygens spacecraft, that took images on Saturn's moon Titan ten years ago, had fatal communications issues resolved only when it was well on its way to Saturn.

On the course homepage you can find three images:

1. `titan.jpg`, taken by the Huygens spacecraft on Titan, see `http://antwrp.gsfc.nasa.gov/apod/ap050115.html`
2. `rosetta_small.jpg`, taken by the Rosetta spacecraft last year, see `www.esa.int/Our_Activities/Space_Science/Rosetta`
3. `rosetta_large.jpg`, also taken by the Rosetta spacecraft.

Start by loading one of the images. A good idea can be to start with the image of Titan, since that is smallest. Once you have the code working, you can then try the Rosetta images.

This document only provides an outline of the assignment, ask David if you need more detailed information.

2 Part 1: Missing data

We will first assume that the values of certain pixels in the image are lost in the data transfer from the spacecraft. For each pixel in the image, independently remove the pixel with a fixed probability p_c, and let $\mathbf{x}_o$ be the vector with with observed pixel values. Start with $p_c = 0.5$, so that approximately half of the pixels are observed. The task is now to reconstruct the missing pixels $\mathbf{x}_m$ given the observed values $\mathbf{x}_o$.

1. Reconstruct the missing values by assuming that the image can be modelled as a mean-zero Gaussian random field with a Matérn covariance function (see Lecture 1). Fix $\nu = 1$, choose κ^2 so that the covariance range is reasonable, and choose σ so that

the variance of the field is similar to that of the observed data[1]: $\sigma^2 = 4\pi\kappa^2$ would give variance one of the field. Using this model, predict $\mathbf{x}_m$ using the conditional mean $\mathsf{E}(\mathbf{x}_m|\mathbf{x}_o)$. *Note: You can do this part on a subimage to reduce computation time.*

2. The Gaussian Matérn field can be approximated by a CAR model with precision matrix $\tau\mathbf{Q}$, where $\mathbf{Q}$ is specified through the stencil

$$\kappa^4 \begin{pmatrix} \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & 1 & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \end{pmatrix} + 2\kappa^2 \begin{pmatrix} \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & -1 & \cdot & \cdot \\ \cdot & -1 & 4 & -1 & \cdot \\ \cdot & \cdot & -1 & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \end{pmatrix} + \begin{pmatrix} \cdot & \cdot & 1 & \cdot & \cdot \\ \cdot & 2 & -8 & 2 & \cdot \\ 1 & -8 & 20 & -8 & 1 \\ \cdot & 2 & -8 & 2 & \cdot \\ \cdot & \cdot & 1 & \cdot & \cdot \end{pmatrix}$$

We will discuss the reason for this later. For now, implement the model and make sure that it is similar to the Matérn model by, for example, comparing model samples and covariances. *Hints: If comparing covariances, look at interior pixels to avoid boundary effects and let $\tau = 2\pi/\sigma^2$ to give the CAR model marginal variances similar to the Matérn model. You can write $\mathbf{Q}$ as $\mathbf{Q} = (\mathbf{G}+\kappa^2\mathbf{I})^\top(\mathbf{G}+\kappa^2\mathbf{I})$, where $\mathbf{G}$ can be expressed using kronecker products based on the RW1 structure matrix (see Lecture 5).*

3. Assume the CAR model for the image $\mathbf{x}$ and use it to predict $\mathbf{x}_m$ using the conditional mean $\mathsf{E}(\mathbf{x}_m|\mathbf{x}_o)$.

4. Investigate the time it takes to calculate $\mathsf{E}(\mathbf{x}_m|\mathbf{x}_o)$ when different types of reorderings of the nodes in the image are used. Also compute the fill-in ratio of the cholesky factor for each reordering. Try without using a reordering, and for example using an approximate minimum degree reordering and a profile reduction reordering. Finally, compare these times with the time it took to compute the corresponding mean for the Matérn model.

5. Investigate how the reconstruction changes if you change the model. You could for example try CAR models based on stencils like

$$\begin{pmatrix} \cdot & -1 & \cdot \\ -1 & 4+\kappa^2 & -1 \\ \cdot & -1 & \cdot \end{pmatrix} \quad \text{or} \quad \begin{pmatrix} -1 & -1 & -1 \\ -1 & 8+\kappa^2 & -1 \\ -1 & -1 & -1 \end{pmatrix}$$

Also try some intrinsic model, such as the ICAR(1), which is obtained using the left stencil above with $\kappa^2 = 0$.

6. For your best model, how large can the amount of missing values be if we want to get a reasonable estimate of the true image? Try all three images here.

[1] Ideally, we should estimate the parameters from the data using e.g. maximum likelihood. However, this is painfully slow for a covariance-based model like this. We will look at parameter estimation in a more realistic scenario in the next section.

3 Part 2: Corrupted data

In the previous section, we assumed that the values of certain pixels in the image were lost in the data transfer from the spacecraft. We will now look at the more realistic setting where the pixels are not missing, but rather replaced by incorrect values.

We model this scenario using a Bayesian Hierarchical model as follows. Let $\mathbf{x}$ be the latent image, modelled using the GMRF model you constructed in the previous section. Furthermore, let $\mathbf{z} = \{z_{ij}\}$ be a set of independent indicators for each pixel, showing whether that pixel has been corrupted or not. We model these using a prior $\mathsf{P}(z_{ij} = 0) = p_c$ and $\mathsf{P}(z_{ij} = 1) = 1 - p_c$, where p_c is the prior probability of a pixel being corrupted. Given the latent field and the indicator field, each pixel is then observed using the likelihood

$$y_{ij}|x_{ij}, z_{ij} \sim \begin{cases} \mathsf{N}(x_{ij}, \sigma_e^2) & \text{if } z_{ij} = 0 \\ \mathsf{U}(0,1) & \text{if } z_{ij} = 1 \end{cases} \tag{1}$$

For this scenario, we will not assume fixed values of the parameters, but instead estimate the parameter from data. Because of this, we assume a β-distribution as prior for p_c, an inverse gamma prior for σ_e, and gamma priors for τ and κ.

First construct a vector $\mathbf{y}$ with observations by drawing from the observation model, with some fixed values for p_c and σ_e^2.

1. Derive the conditional posteriors needed to implement a Gibbs sampler:

   $$\pi(x_i|\mathbf{x}_{-i}, \mathbf{y}, \mathbf{z}, \tau, \kappa, \sigma_e, p_c), \quad \pi(z_i|\mathbf{z}_{-i}, \mathbf{y}, \mathbf{x}, \tau, \kappa, \sigma_e, p_c), \quad \pi(\tau|\mathbf{x}, \mathbf{y}, \mathbf{z}, \kappa, \sigma_e, p_c),$$
   $$\pi(\kappa|\mathbf{x}, \mathbf{y}, \mathbf{z}, \tau, \sigma_e, p_c), \quad \pi(\sigma_e|\mathbf{x}, \mathbf{y}, \mathbf{z}, \tau, \kappa, p_c), \quad \pi(p_c|\mathbf{x}, \mathbf{y}, \mathbf{z}, \tau, \kappa, \sigma_e).$$

   The posterior for κ has no simple distribution, so you do not need to derive that.

2. Implement a single-site Gibbs sampler to sample the joint posterior for the parameters, the latent field, and the indicator field. You will need a Metropolis-within-Gibbs step to sample κ. So it might be a good idea to to start with κ fixed.

3. Improve the Gibbs sampler by blocking, and compare the speed of convergence by estimating the correlation functions for the parameter tracks.

4. Calculate the posterior mean of the latent field and test how well the method can correctly detect missing pixels for different degrees of corruption. How does the results compare to the case with missing values?

4 Report

Write a clear and concise report presenting your approach to the assignment, discussing the methods and results. Include figures with explanatory texts. Submit the report as a PDF and also include a zip-file with your code that can be used to run the analysis. Organize the code so that one main file `proj1part1` (`.R` or `.m` depending on if you use Matlab or R) runs the first part, and another main file `proj1part2` runs the second part. Email the files to `david.bolin@chalmers.se`. The report is due on Monday February 23.