Home assignment 3

In this assignment, you should choose one of the following problems. Only an outline is provided here, more detailed information on suitable models and algorithms are discussed in the lecture notes for Lecture 11 & 12.

fMRI data: Since the two first projects make use of the same fMRI-dataset a short description is provided here. The data consists of fMRI data from a single trial where the subject judged whether a pair of written words rhymed or not. The experiment consisted of alternating 20-second work and rest blocks. The goal of the fMRI analysis is identify parts of the brain which are active during the task. A slice from the fMRI experiment is available as `img` in `fmri.mat`.

The resulting image has 160 layers representing the experiments 320s runtime (at 2s temporal resolution). The data also contains a matrix, `X`, of temporal regressors consisting of a constant and a linear trend — to capture drift in the equipment and slow motion of the subject — and indicator functions for the work blocks.

1. **Classification of fMRI data using MRFs and Gibbs (MCMC):**
   The task is to classify the pixels in an image, assuming an MRF structure for the pixel classes, and conditionally independent Gaussian observations,

   $$\left[y(\mathbf{s}_i)|z(\mathbf{s}_i,k)=1\right] \in \mathsf{N}\left(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\right)$$

   Study the fMRI data and the initial analysis presented in `fmri_class_init.m`. The idea is to first perform data reduction using either PCA directly on the fMRI data or by first regressing the data onto the temporal regressors in `X` and then performing PCA on the regression coefficients. Here we see the image as having either 160 (the time points) or 11 (the regression coefficients) "colour layers".

   For the classification use PCA to reduce the data dimension, and then classify the images using the increasingly complex models provided by `kmeans.m`, `normmix_gibbs.m` and the MRF model.

   Sampling procedures for the MRF-model can be constructed using the functions for simulation of the field (using Gibbs sampling) and parameters (using MCMC) in MRFs provided in the course files (see list below). A suitable starting point for the MRF-estimation are the parameter estimates and classification obtained from `normmix_gibbs.m`.

   Study the difference between the classifications and **investigate the effects of:** 1) Different initial data reductions; 2) Different neighbourhood sizes (4 or 8); and 3) Having the same or different β:s for the classes.

   Use the special functions for MRF:s
   `mrf_gaussian_post.m` (posterior alpha-parameters for Gaussian data)
   `mrf_sim.m` (simulate samples from an MRF model)
   `gibbs_alpha_beta.m` (pseudo-likelihood MH for MRF-parameters)
   `gibbs_mu_Sigma.m` (Sampling from μ and Σ given a set of observations from a Gaussian-distribution.)

Note that the posterior for α, β is tricky, and it is typically not possible to fully explore the distribution. However, it should converge to a parameter set for which the results of classification works.

For a similar model used in computer tomography see
`https://arxiv.org/abs/1607.02188`

2. **Spatio-Temporal Data:**
One possible model for spatio-temporal data is to use a model with temporal functions where the coefficients vary between locations.

$$y(s,t) = \sum_k \beta_k(s) X_k(t) + \varepsilon(s,t) \qquad \varepsilon(s,t) \in \mathsf{N}\left(0, \tau_\varepsilon^{-1}(s)\right)$$
$$\beta_k \in \mathsf{N}\left(\mathbf{0}, \tau_{Q,k}\mathbf{Q}_{\text{SAR/CAR}}(\varkappa = 0)\right).$$

Here we are using a SAR or CAR field with $\varkappa = 0$. These models, called intrinsic fields, have connections with smoothing splines. The simplification to $\varkappa = 0$ is done to avoid estimating $\varkappa$; a step that is possible but computationally expensive. Also note that we are allowing different precisions for each of the β-fields and different residual precision for each location.

Study the fMRI data and the initial analysis presented in `fmri_ST_init.m`. In a real example we would use one β-field for each work block. However, that model is **very** computationally expensive so to illustrate the principal we will sum the indicators to obtain one indicator for an event during any of the time periods.

For the project you should construct a gibbs sampler which alternates the following steps: 1) Sample from $\boldsymbol{\beta}|\mathbf{y}, \boldsymbol{\tau}_\varepsilon, \boldsymbol{\tau}_Q$. 2) Sample from $\boldsymbol{\tau}_Q|\mathbf{y}, \boldsymbol{\beta}, \boldsymbol{\tau}_\varepsilon$ (Γ-distributions). 3) Sample from $\boldsymbol{\tau}_\varepsilon|\mathbf{y}, \boldsymbol{\beta}, \boldsymbol{\tau}_Q$ (Γ-distributions). Sampling $\boldsymbol{\beta}|\mathbf{y}, \boldsymbol{\tau}_\varepsilon, \boldsymbol{\tau}_Q$ corresponds to sampling from the posterior for a Gaussian field. Due to the size of the latent field it becomes **VERY** important to reorder Q_{xy} before computing the choleskey factor!
Beware: matlab uses $\Gamma(\alpha, 1/\beta) = \Gamma(k, \theta)$ **for** `gamrnd`.

Given parameter estimates, compute the posterior expectation and variance for $\boldsymbol{\beta}$ (you probably need to simulate for the variance since Q_{xy} will be too big to invert). Also study the $\boldsymbol{\tau}_\varepsilon$:s. Does any pixels seem to have large variances? How does the temporal variation and residuals look at these points?

Use the β_3-field to determine which areas of the brain that have significant activity during the work blocks. Compare to the linear regression results in `fmri_ST_init.m`. When doing a hypothesis test for significant pixels we are doing multiple tests. If each test has a 5% significans level and **no** areas have significant effect we would expect, on average, $87 \times 105 \times 0.05 \approx 457$ false positive finds. To account for the multiple testing it is recommended to use a Bonferroni correction
`https://en.wikipedia.org/wiki/Bonferroni_correction`
which gives a corrected significans level for the multiple testing as:
$\alpha_{\text{adj.}} = \alpha_{\text{org.}}/N$.

For a similar model applied to sattelite data see
`https://doi.org/10.1016/j.csda.2008.09.017`

3. **Corrupted data:**
   Previously in the course, a task was to reconstruct missing pixel values, assuming that it was known which pixels were missing. In a more realistic setting, the pixels might not be *missing*, but rather *replaced* by incorrect values.

   The purpose of this project is to use Bayesian methods to both identify the faulty pixels, and to reconstruct the missing data.

   Use a Matérn GMRF (SAR-model, with fixed $\varkappa^2$ — Consider which values for *kappa*^{2} would give a reasonable range) as prior model for the image, assume that each pixel is observed with probability p_c, say, with some small Gaussian measurement noise, and is otherwise replaced with an independent random *Unif*$(0,1)$ value (for example). The hierarchical model now contains a latent image, $\mathbf{x} \in \mathsf{N}\left(\boldsymbol{\mu}, \mathbf{Q}^{-1}\right)$, a set of independent latent indicators for each pixel $z(\mathbf{s})$, showing wether or not the pixel has been corrupted. The prior distribution for $z(\mathbf{s})$ is $p(z(\mathbf{s}) = 0) = p_c$ and $p(z(\mathbf{s}) = 1) = 1 - p_c$. Given the latent image and the indicators the observation for each pixel is

$$y(\mathbf{s})|\mathbf{x}, \mathbf{z} \in \begin{cases} \mathsf{N}\left(x(\mathbf{s}), \sigma_\varepsilon^2\right) & \text{if } z(\mathbf{s}) = 0 \\ \mathsf{U}\,(0,1) & \text{if } z(\mathbf{s}) = 1 \end{cases}$$

   Derive a Gibbs algorithm to sample 1) the parameters p_c (β-distribution), σ_ε (IΓ-distribution) and τ (Γ-distribution) in the SAR field (we're assuming $\varkappa$ fixed, although it could be estimated using e.g. a random walk MCMC), 2) the latent unknown image $\mathbf{x}$ (similar to *computer exercise 3*), 3) the indicator field for bad pixels $\mathbf{z}$ (see `normmix_posterior.m` for hints on computing the posterior probabilities).
   Beware: matlab uses $\Gamma(\alpha, 1/\beta) = \Gamma(k, \theta)$ **for** `gamrnd`.

   Start with the `titan.jpg` image (or another grayscale image) and add your own random noise (for `titan.jpg` a correct implementation should handle $p_c < 0.5$). You might also want to try other images (Which images are handled well and which images cause issues? Why?)

   Present posterior image, and analys how well the method can correctly detect missing pixels (which missing pixels are miss-classified).

Report

Write a clear report presenting your approach to the assignment, discussing the methods and results. Include figures with explanatory texts. As before, submit a PDF and relevant Matlab files using **Canvas**.

The project should also be presented as a mini-seminar (10 minutes) for the other course participants. **The report is due 23:59 on the day before your seminar.** Use the doodle link on the webpage to sign up for a presentation time.