LUND UNIVERSITY

SPATIAL STATISTICS WITH IMAGE ANALYSIS

Home assignment 3: Corrupted data

Author:
David Larsson
tfy15dla@student.lu.se

Author:
Sofie Hellmark
mat15she@student.lu.se

December 20, 2018

1 Introduction

The goal of this project is to reconstruct a randomly corrupted image using Gibbs sampling and GMRF. To reconstruct a latent field of the image GMRF using SAR(1) and another latent field indcating which pixels that are corrupted was used. After the image was reconstructed Bayes theorem was used to classify whether the pixels in the original image was corrupted or not since they should belong to different probability densities. With this classification and our reconstructed image a new joint probability for the whole image for the parameters σ^2, p_c and τ can be set up and new values can be drawn. We let this Gibbs sampling algorithm iterate until the parameters converge and then we start to reconstruct the image. We set $\kappa = 0.01$ and use the notation $\theta = (\sigma^2, \kappa, \tau, p_c)$ for all the parameters. The initial value of the other parameters were guessed.

2 Posterior for the parameters and latent fields

The corrupted pictures contains two fields, the latent image $\boldsymbol{x} \in \boldsymbol{N}\Big(\boldsymbol{\mu}, \boldsymbol{Q}^{-1}\Big)$ and the indicator for the corrupted pixels with uniform distribution. The prior distribution for z is $p(z_i = 0) = p_c$ and $p(z_i = 1) = 1 - p_c$, where 1 means the pixel is corrupt and 0 non-corrupt . This gives distribution for all the pixels in the entire corrupted image

$$y_i|x_i z_i \in \begin{cases} \boldsymbol{N}(x_i, \sigma^2) & \text{if } z_i = 0 \\ \boldsymbol{U}(0,1), & \text{if} z_i = 1. \end{cases}$$

To identify the corrupted pixels and then be able to recreate the image before it was corrupted we need first to compute the posterior for the latent field $\boldsymbol{x}$ and indicator field for the bad pixels $\boldsymbol{z}$.

$$\mathrm{p}(\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{\theta}|\boldsymbol{y}) \propto \mathrm{p}(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}, \boldsymbol{\theta})\mathrm{p}(\boldsymbol{x}|\boldsymbol{\theta})\mathrm{p}(\boldsymbol{z}|\boldsymbol{\theta}) = \mathrm{p}(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{z}, \sigma_\epsilon)\mathrm{p}(\boldsymbol{x}|\kappa, \tau)\mathrm{p}(\boldsymbol{z}|p_c) \tag{1}$$

From the full posterior for the latent field and the parameters seen in equation 1 we can see that the conditional probability of getting the fields and parameters wibe proportional to $\mathrm{p}(\boldsymbol{x}|\kappa, \tau)\mathrm{p}(\boldsymbol{z}|p_c)$ allowing the Gibbs sampler converge for the full posterior because of the parameters converging to the real values.

2.1 $\mathrm{p}(\boldsymbol{x}|\boldsymbol{z}, \boldsymbol{\theta}, \boldsymbol{y})$ — Reconstruction in a GMRF

The conditional posterior for the latent field $\mathrm{p}(\boldsymbol{x}|\boldsymbol{z}, \boldsymbol{\theta}, \boldsymbol{y})$ can be written as

$$p(\boldsymbol{x}|\boldsymbol{z}, \boldsymbol{\theta}, \boldsymbol{y}) \in N(\boldsymbol{\mu}_{x|y}(\boldsymbol{z}, \boldsymbol{\theta}), \boldsymbol{Q}_{x|y}^{-1}(\boldsymbol{z}, \boldsymbol{\theta})) \tag{2}$$

This can then be used to reconstruct the field using a GMRF SAR-model, i.e $\boldsymbol{Q} = \kappa^4 \boldsymbol{C} + 2\kappa^2 \boldsymbol{G} + \boldsymbol{G}_2$. To do this we created the field $\tilde{\boldsymbol{x}} = [\boldsymbol{x}\ \beta]$ to include also a nonzero mean. After this we used the initial guesses of the parameters to form a indicator of the bad pixels and from that we constructed an observation matrix $\boldsymbol{A}$ for the guessed non-corrupted pixels. The expected value for the field $\tilde{\boldsymbol{x}}$ was then created using

$$E(\tilde{\boldsymbol{x}}|\boldsymbol{y}) = \boldsymbol{Q}_{x|y}^{-1}\tilde{\boldsymbol{A}}^T\boldsymbol{Q}_\epsilon\boldsymbol{Y}$$
$$V(\tilde{\boldsymbol{x}}|\boldsymbol{y}) = \boldsymbol{Q}_{x|y}^{-1}$$

where $\tilde{\boldsymbol{A}}$ is the entire observation matrix and $\boldsymbol{Y}$ is the pixels previously guessed to be known. $\boldsymbol{Q}_\epsilon$ is the precision matrix containing our guessed σ^2. The precision matrix is given by $\boldsymbol{Q}_{x|y} = \tilde{\boldsymbol{Q}} + (\tilde{\boldsymbol{A}}^T\boldsymbol{Q}_\epsilon\tilde{\boldsymbol{A}})^{-1}$. $\tilde{\boldsymbol{Q}}$ contains the precision matrix for the field, $\boldsymbol{Q}_x$ and the precision for β. Using these equations a reconstruction of the picture was created and this sample of the latent filed is then used as to determine the probabilities to belong to the non corrupted pixels.

2.2 p$(\boldsymbol{z}|\boldsymbol{x},\boldsymbol{y},\boldsymbol{\theta})$ — Classification

Using Bayes theorem we can set the probability for each pixel to belong to a certain class. The probabilities of each class depend on p$_c$ as $p(z_i = 1) = 1 - p_c$ and $p(z_i = 0) = p_c$. The non-corrupted pixels should be normal distributed with the variance σ^2 and the sample that we got from the latent field, $\boldsymbol{x}$. The probability density function for the corrupted pixel is $p(y_i|z_i = 1) = 1$ since they are assumed to be uniformed distributed between [0,1]. This give us the probability of a pixel being non-corrupt:

$$p(z_i = 0|y_i, x_i, \sigma^2, p_c) = \frac{p(y_i|z_i = 0, x_i, \sigma^2)p(z_i = 0)}{\sum_{k=0}^{1} p(y_i|z_i = k)p(z_i = k)} \tag{3}$$
$$\text{where } p(y_i|z_i = 0, x_i, \sigma^2) = \frac{1}{(2\pi\sigma^2)^{1/2}}\exp(-\frac{(y_i - x_i)^2}{2\sigma^2}) \tag{4}$$

Where y_i is the pixels from the given corrupted image and x_i is the pixels from the sampled latent field $\boldsymbol{x}$. This probability for each pixel to be non-corrupted was used to classify pixels in a randomly sampled image. The classification is then used as the latent model for the classification and should be better and better for each loop until it converges.

2.3 Sampling of new parameter values p$_c$, τ and σ^2

From the recent classification the number of known and unknown pixel values can be obtained. These are used to draw a new p$_c$ that is Beta distributed. The joint distribution of p$_c$ is given by $f(p_c) \propto p_c^{n_{obs}}(1 - p_c)^{n_{corrupt}} = p_c^{a-1}(1 - p_c)^{b-1}$ where $a = n_{obs} + 1$ and $b = n_{corrupt} + 1$. a and b was used as in parameters to **betarnd**.

The joint distribution of τ can be written as p$(\tau|\boldsymbol{x}, \kappa) \propto \tau^{N/2}\exp(-\tau\frac{x^T Q_0(\kappa)x}{2})$ since we know that the field $\boldsymbol{x}$ is normally distributed, comparing this to a gamma distribution $f(\tau) \propto \tau^{k-1}\mathrm{e}^{-\tau/\theta}$ the terms $k = N/2 + 1$ and $\theta = \frac{2}{x^T Q_0(\kappa)x}$ could be identified. A new τ was now drawn from this $\Gamma(k, \theta)$ distribution.

We know that p$(\sigma_\epsilon|\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{y})$ is inverse-gamma distributed and if σ^2 is inverse-gamma distributed, $\frac{1}{\sigma^2}$ is then gamma distributed. The distribution of $\frac{1}{\sigma^2}$ depend on the non-corrupted pixels in $\boldsymbol{y}$ and the sampled value of the reconstruction for those pixels in $\boldsymbol{x}$. Since all the known non-corrupted pixels should be normal distributed we can write the distribution as: p$(\sigma^2|\boldsymbol{x}, \boldsymbol{y}) \propto$

$p(\boldsymbol{x}|\sigma^2, \boldsymbol{y}) \propto (\frac{1}{\sigma^2})^{N/2} \exp(-\frac{1}{\sigma^2}\frac{(\boldsymbol{y}-\boldsymbol{x})^T(\boldsymbol{y}-\boldsymbol{x})}{2})$. Here we use that we got a new guess on which pixels are known when doing the classification above. Identifying the terms as we did with τ above but now as $f(\frac{1}{\sigma^2})$ we get $k = N/2 + 1$ and $\theta = \frac{2}{(\boldsymbol{y}-\boldsymbol{x})^T(\boldsymbol{y}-\boldsymbol{x})}$. A new $\frac{1}{\sigma^2}$ was drawn from this gamma distribution.

Drawing from these parameters together with sampling from the field will then converge to the original image, however the first samples of the field, indicator and draws of the parameters will be off. To allow the parameters to converge we set a burn in time which meant doing the iteration a couple of times before we stored the data from the reconstructed image. I our case we set this burn-in so that we only stored the last hundred iterations and we set the total number of iterations to around 500 because that seemed to be enough to make the parameters to converge. However the total number of iterations might vary since a larger density of corrupt pixels will make the convergence of the parameters slower and you might need to increase the total number of iterations.

3 Reconstructed images

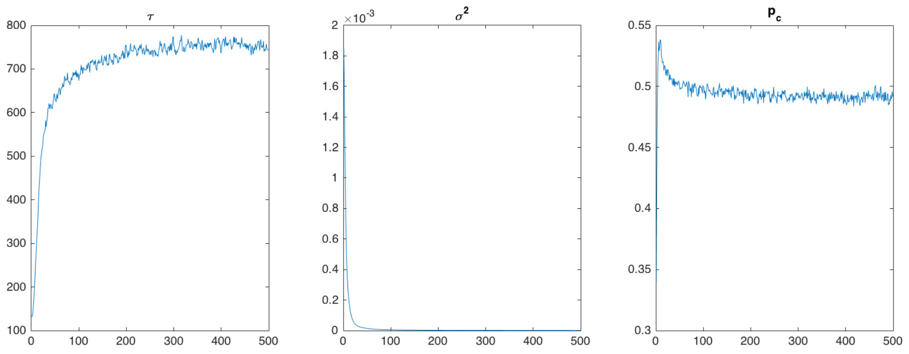

Figure 1: Parameter estimation for our Gibbs Sampler after 500 iterations on the *titan.jpg* image. All parameters seems to converge after 300 samples. The last 100 are used to reconstruct the image.

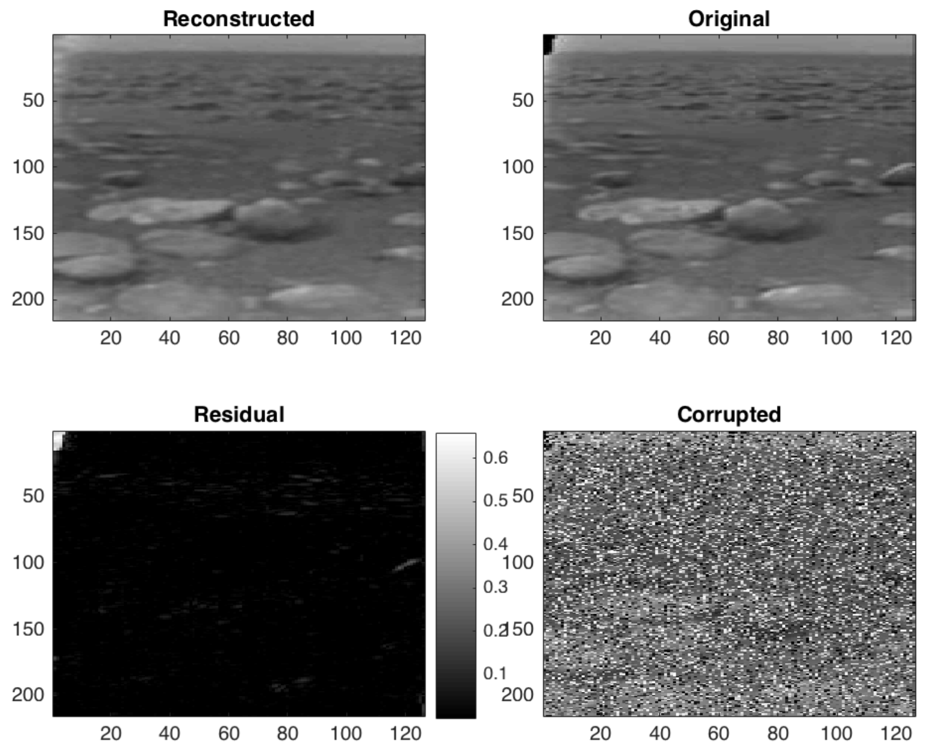

Figure 2: Reconstruction of the image where 50% of the pixels were corrupted. The reconstructed image, original, residual between original and reconstructed as well as corrupted image is shown. Last value of $\theta = \begin{bmatrix}740 & 0.01 & 7.7e^{-7} & 0.49\end{bmatrix}$ with 500 iterations where a mean last 100 was used to reconstruct image

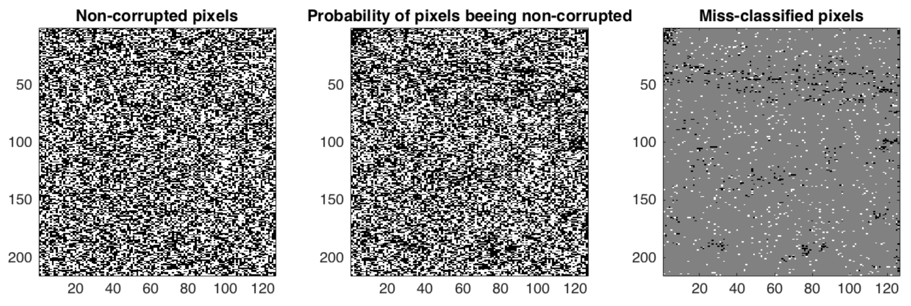

Figure 3: The first image shows the indexes we corrupted with a probability of 50% marked with black color. The second shows the probability for each pixel to be non-corrupted by our Gibbs sampler, these are the values used for reconstruction of the image in the Gibbs sampler. The third image shows the pixels that were wrongly classified given that all the pixels from the original image were non-corrupt. The black pixels were classified as corrupt but should be non-corrupt and the white were classified as ok but should be corrupt.

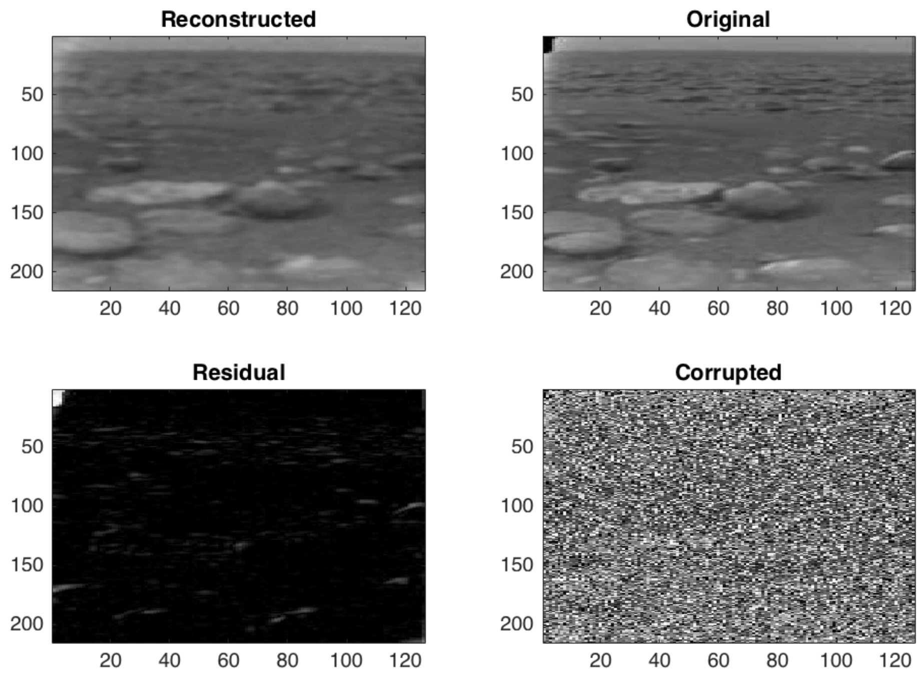

Figure 4: Reconstruction of the image where 70% of the pixels were corrupted. The reconstructed image, original, residual between original and reconstructed as well as corrupted image is shown. Last value of $\theta = \begin{bmatrix} 840 & 0.0100 & 1.3e^{-6} & 0.28 \end{bmatrix}$ with 500 iterations where a mean last 100 was used to reconstruct image

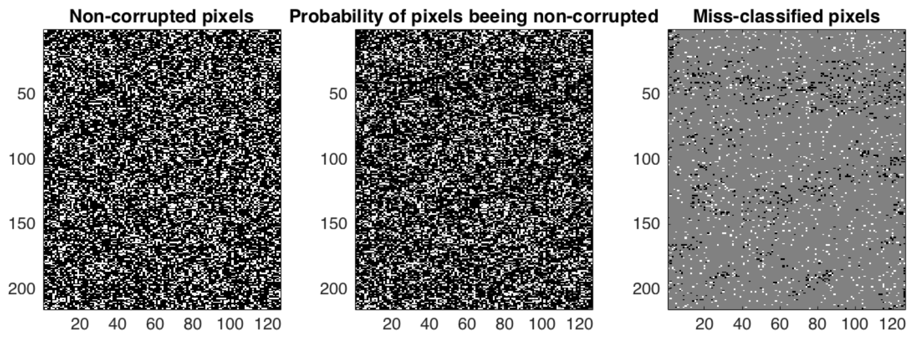

Figure 5: The first image shows the indexes we corrupted in black color. The second shows the probability for each pixel to be non-corrupted, these are the values used for reconstruction of the image in the Gibbs sampler. The third image shows the pixels that were wrongly classified given that all the pixels from the original image were non-corrupt. The black pixels were classified as corrupt but should be non-corrupt and the white were classified as ok but should be corrupt.

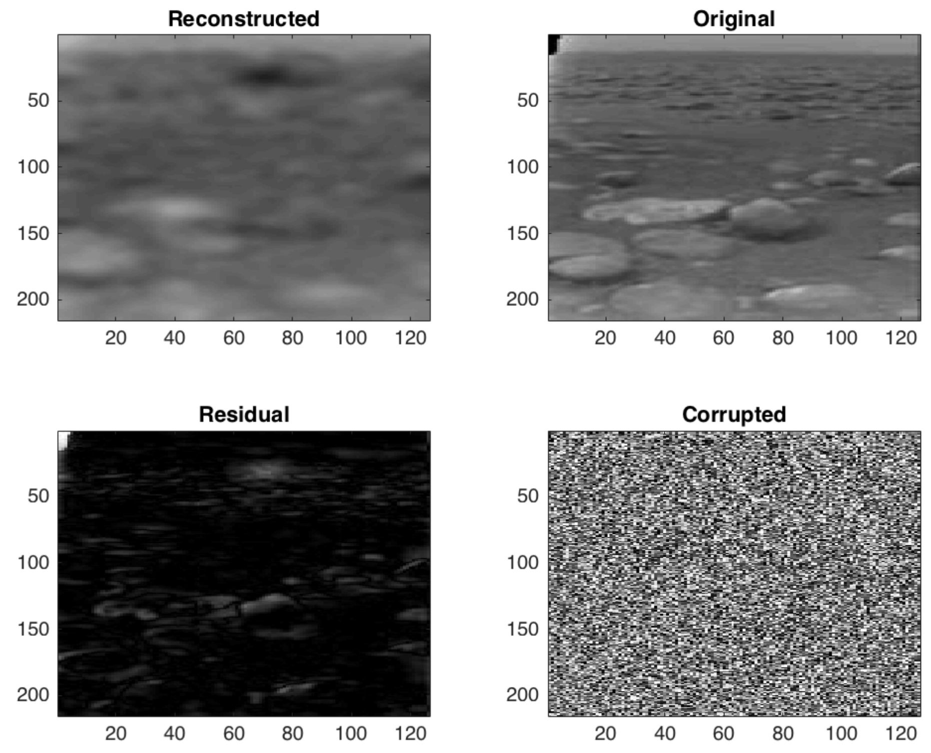

Figure 6: Here 90% of the pixels were corrupted and one can kind of distinguish some objects but no longer see what the image illustrates. This was made from the 100 last of 1000 iterations since the parameters did not converge until after 900 iterations.

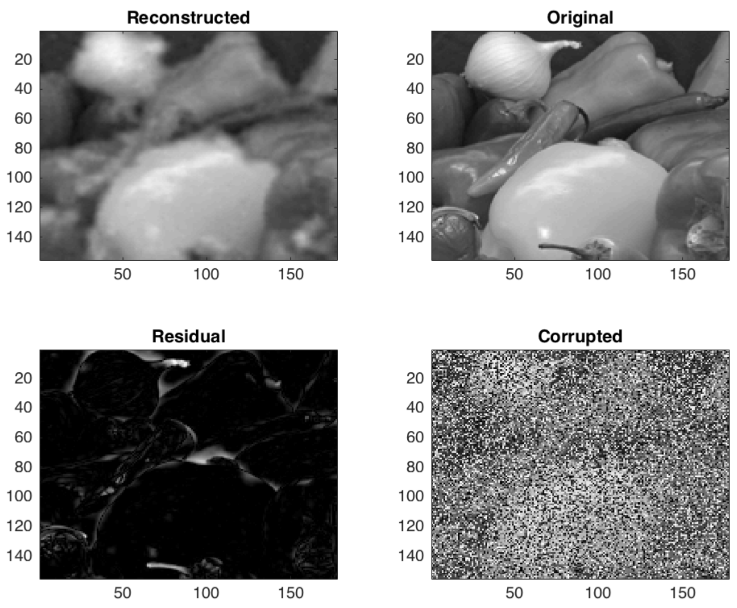

Figure 7: Another image - we see that sharp edges are hard - 70% corrupt pixels - 500 iterations where the last 100 were used to reconstruct the image $\theta = \begin{bmatrix} 800 & 0.01 & 1.49e^{-6} & 0.27 \end{bmatrix}$ although tau did not really converge - went between 700 and 800 the last 100 iterations

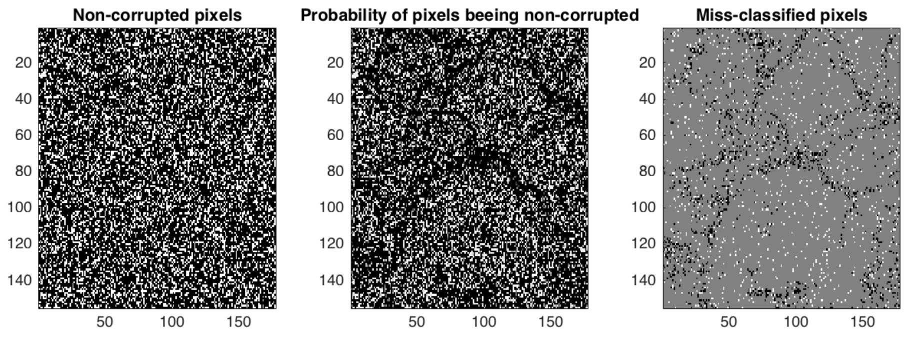

Figure 8: The first image shows the indexes we corrupted in black color. The second shows the probability for each pixel to be non-corrupted, these are the values used for reconstruction of the image in the Gibbs sampler. The third image shows the pixels that were wrongly classified given that all the pixels from the original image were non-corrupt. The black pixels were classified as corrupt but should be non-corrupt and the white were classified as ok but should be corrupt. Here we can distinguish the edges.

4 Discussion

4.1 Convergence of the parameters

From the reconstruction shown in figure 1 the parameters are plotted against the number of iterations of the Gibbs sampler. All of the parameters seems to converge and we know that for this example we constructed an image with the probability of a pixel being corrupted with $p_c = 0.5$ which is the value our sampler converges to. They converged slower for images with more corrupted pixels, and one maybe should make a check whether the parameter has converged before beginning to save the image.

4.2 Reconstruction of the image

By observing figure 2, where a probability of a pixel being corrupted was set to 0.5, the reconstruction seems to be very good and this is also confirmed by the residuals showing almost all zero except for in the upper left corner.

However by looking at the original non-corrupted image the upper left corner seems to deviate from the rest of the picture a lot and our model see the corner as corrupted pixels. This is confirmed by looking at figure 3 which shows the probability of a pixel classified as corrupt by our sampler. By looking at the residuals when comparing to the pixels we know was corrupted most of the upper left corner is classified as corrupted by the sampler. The corner is hard to reconstruct perfectly, and the same would hold for our model if there was a lot of missing samples close to each other anywhere in the image.

The GMRF reconstruction will take the neighbours of the unknown pixels into account when creating the field and in our model we've used the SAR(1) model that uses a 12-neighbour filter to reconstruct a corrupt pixel. As we see the reconstructed images looks a lot better than the corrupted but should be smoother than the original.

In figure 4 reconstruction of an image with 70% corrupted pixels is shown and it still seems to be able to make a good reconstruction. By comparing this residual with the residual for the 50% case shown in figure 2 we now have larger residuals, and a smoother image, but the reconstructed image is still close to the original and definitely better than the corrupted. The parts not captured here that were captured in the 50% image we identify as edges.

Another property of taking the neighbours into account when reconstructing is that quick changes such as sharp edges and details will be hard to reconstruct. This can clearly be seen in the capsicum image, figure 7, that have more structure than the *titan* image and details and sharp edges seems to disappeared completely. This is due that our model depend a lot on the nearest neighbors when making the reconstruction. This results in a field where the edges will be smoothed out and some details are completely missed.

Looking at the third image in figure 3, 5 and 8 the white pixels were classified as okay but were set as corrupt and the black pixels were always classified as corrupt but should be okay. What is interesting here is what we see in the capsicum image, the sharp edges seems to be classified as corrupt. This should be due to and enhance the smoothing effect. The fact that pixels are classified as non-corrupt when actually being corrupt doesn't seem to harm our reconstructed image a lot.

A thought is that CAR(1) would probably not smooth the image as much as SAR(1) since it doesn't depend on as many neighbours. A CAR model would on the other hand probably reconstruct the images as good as the SAR model did if a lot of pixel is missing.

When running our Gibbs sampler on *titan* with 90% corrupted pixels one can no longer distinguish what the image contains, see figure 6 corrupted image and one can get a hint of what the image contains. The parameters also seems hard to estimate the more detailed or corrupted out input image is.

The conclusions we draw is that our Gibbs sampler is good at reconstructing images that doesn't have sharp edges, details and rapid changes in the structure that doesn't take harm from being smoothed out. However all of our reconstructed images are much better than the carder to estimate the more details or corrupt our image is.