0.1 Introduction

I have chosen to implement the Metropolis-Hasting MCMC algorithm. The algorithm can be used to sample distributions where only functions proportional to the density is known. This assignment will look at how this algorithm can solve the optimizing problem of finding the minima of the Rosenbruck and Himmelblau functions. And how it can be used to determine parameter values from a created data sample.

0.2 Metropolis hastings algorithm

The metropolis hastings algorithm uses monte carlo markov chains to map out the distribution of functions. A monte carlo markov chain (mcmc) is a chain of steps where the probability of the next step only depends on the previous (markov chain). The step is taken in a random direction (monte carlo). If the markov chain fullfills the principle of detailed balance then for a large number of steps it will converge towards stable states x_i with some distribution $P(x)$. The principle of detailed balance is described by

$$\pi_i P_{ij} = \pi_j P_{ji} \tag{1}$$

Where P_{ij} is the probability of going from state i to state j wich also can be expessed as the markov transition matrix. And π_i is the equilibrium probability for being in state i. A example of detailed balance is when spin states flips and there enters a equilibrium between spin up and spin down states. The algorithm works by taking a random step from one state x_i to another x_{i+1} and then calculating the probability of takeing the step. The probability is described by α

$$\alpha(x_i, x_{i+1}) = \frac{P(x_i)}{P(x_{i+1})} \frac{Q(x_i|x_{i+1})}{Q(x_{i+1}|x_i)} \tag{2}$$

Where $Q(x_i|x_{i+1})$ is the conditional probability of going from the x_i state to the x_{i+1} state. For example with spin flips the probability for a flip depends of the distribution of the surounding spins the temperature and maybe constant magnetic field. The algorithm can be described by. If $\alpha \geq 1$ the step is always taken
if $\alpha \leq 1$ the step is taken with the probability α
The implementation can be described by

```
-Generate a random step S
-Find the new point P_new = P_old + S
```

```
-Calculate alpha = f(P_new)/f(P_old)*pdist(P_new)/pdist(P_old)
where f(P_new) are the function value at P_new and
pdist(P_new) are the parameter distribution at P_new.
-Generate a random number u
-If alpha > u: Take the step P_old = P_new
else: remain at P_old
-save the P_old value in a vector
-Retrun and generate another step untill the desired number
of steps is reached
```

For a function given by $P(x)$ with no conditional probability $Q(x)$ it can be shown that for large values of random steps the histogram of the steps will converge to a function $G(x)$ proportional to the original $P(x)$. This is shown in figure (1)

0.3 The histogram

The points generated with the MCMC algorithm can be represented in a histogram. In the 1-dimensional case the histogram splits the axis where the data lies into bins. The length, number and starting point for the bins has to be specified. Then the histogram function counts how many data points there are in each bin. And returns a vector with the bins and one containing the number of counts in each bin. The histogram can be extended to two or more dimensions where for each dimension the length and startingpoint has to be specified. In the multidimensional case the number of bins is assumed to be the same for all dimensions for this histogram function. The implementation of the histogram function

```
-create a histogram vector from binlength, binnumber and startpoint
-repeat this for all dimensions
-loop over all datapoints
-if the datapoint are in the histograminterval use it
else discart it
-for each dimension subtract the startpoint from the datapoint
divide with the bin length and floor the number
-save each set of these intrences in a count vector
-add up all counts with the same intrences
```

In the implementation the storing of the counts in multi-dimensions are transformed into a one dimensional list/array before the counts are added. The transformation into one dimension is done by

```
-Make the mapping of a n-dimensional matrix into a list/array
such that each entrance in the matrix has a corresponding place
in the array.
-This is done by stepping for each dimension with a stepsize of
the product of the remaning dimensions
```

Each entrance in the one dimensional array is now corresponding to a entrance in the matrix and each count can be added to the one dimensional array/list. Normally it is only desirable to display two or maybe three dimensions of the histogram. Therefore a function can call these entrances and get the counts from the countvector.

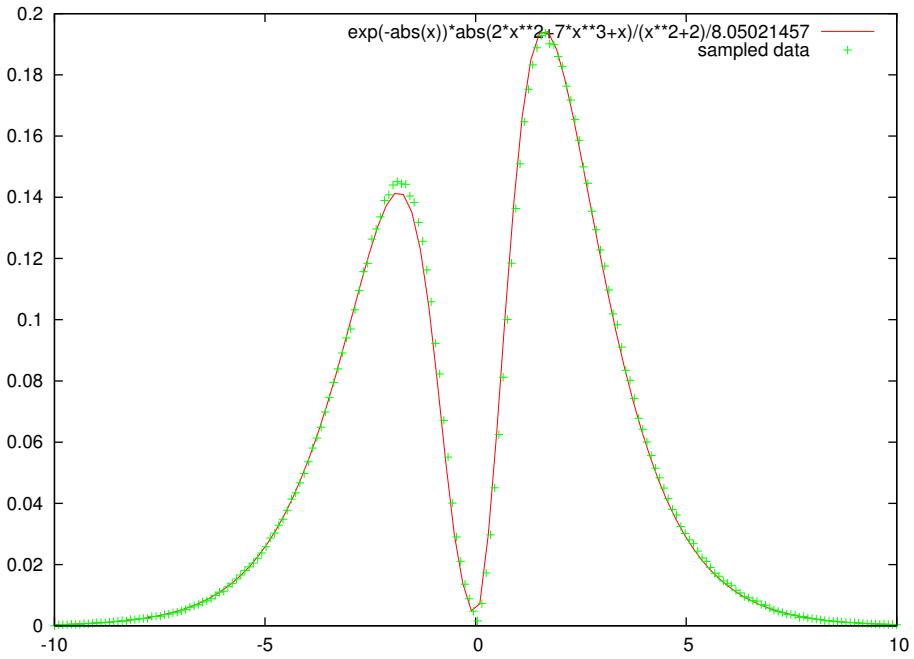

Figure 1: This is a figure of MCMC algorithm used to sample the function $f(x) = e^{-|x|}\frac{|2x^2+7x^3+x|}{x^2+2}$. The histogram and function is normalized.

0.4 Sampling with the MCMC

If it is possible to sample from a function proportional to the distribution the MCMC algorithm can be used to picture the distribution. For a analytical

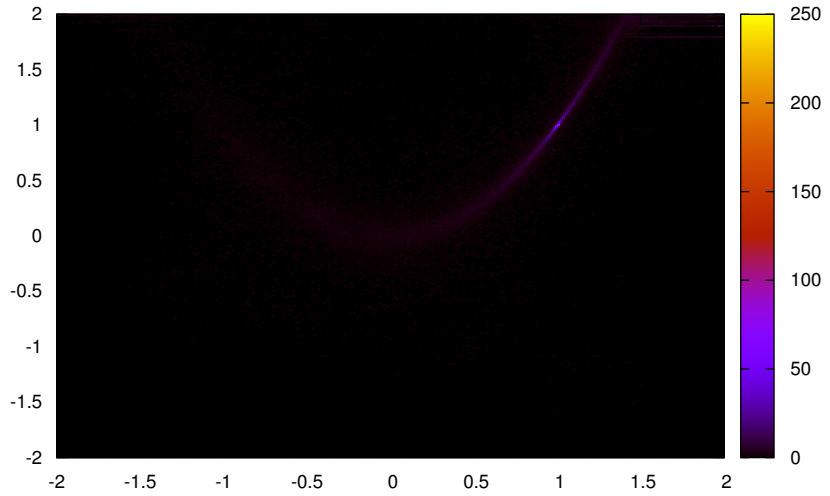

Figure 2: This is a figure of MCMC algorithm used on the Rosenbruck function $f(x) = (1-x)^2 + 100\left(y - x^2\right)^2$. The contour plot is of the square of each histogrambin.

function like $f(x) = e^{-|x|}\frac{|2x^2+7x^3+x|}{x^2+2}$ the function can simply just be plottet. But it can also be sampled with the metropolis algorithm with $\alpha = \frac{f(x_{new})}{f(x_{old})}$ where $f(x)$ is the function. To compare the analytical function $f(x)$ with the histogram from the MCMC they are both normalized. The integral of the function can be calculated with the adaptive integration from the course.

$$\int_{-10}^{10} dx e^{-|x|}\frac{|2x^2+7x^3+x|}{x^2+2} = 8.05 \tag{3}$$

The histogram in 1 dimension can be normalized by multiplying each value with $\frac{1}{N_{tot}\cdot delta_{bin}}$ where N_{tot} is the number of sampled points and $delta_{bin}$ is the length of each histogram bin. The result can be seen in figure (1). This effect makes the algorithm attractive. It could be advantageus when integrating that most of the steps are at the highest values.

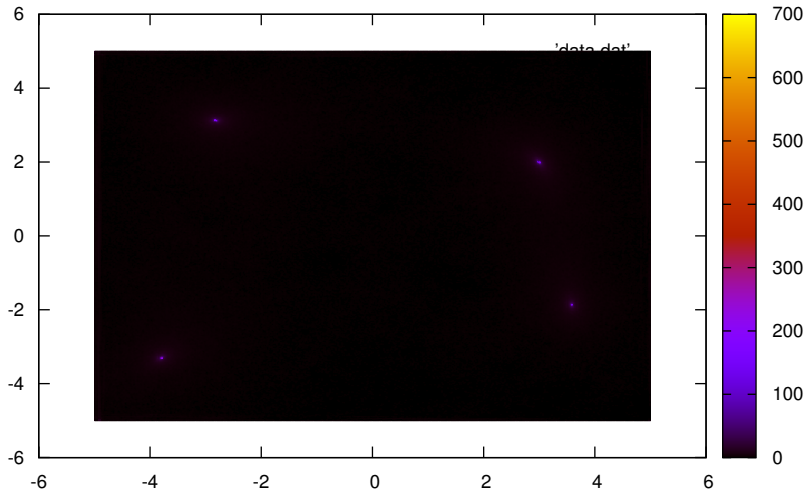

Figure 3: This is a figure of MCMC algorithm used on the Himmelblau function $f(x) = \left(x^2 + y - 11\right)^2 + \left(x + y^2 - 7\right)^2$. The contour plot is of the square of each histogrambin.

0.5 Optimizing

When looking for the biggest or smallest value of a function the algorithm can be used to find those. For the smallest values the step is only takes if the new point lies lower than the previous. This is done for the Rosenbruck and Himmelblau function which also where treated in the exercises. The result are plottet in figure (3,2).

0.6 Length of the random step

The algorithm will converge to the function for all stepsizes. However the stepsize has a effect on how fast the convergence are. The parameterspace will eventually be covered by all stepsizes, unless there are some regions with finite length wich are zero. The small steps gives more information of the path it choses but it will move slower. To minimize the computation time

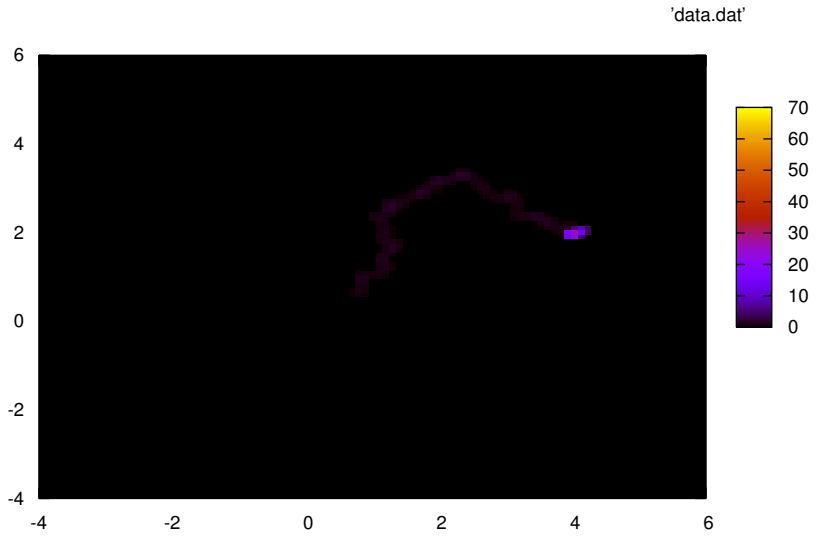

Figure 4: This is a figure where 10.000 gaussian data points with mean value 4 and variance 2 are fittet with the MCMC algorithm to a gaussian function. The white lines are an artifact from the histogram.

it is important to chose a reasonable stepsize for the problem at hand. A rule of thumb is to have a stepsize such that for 3 rejections there is one acceptance.

0.7 The parameter determination with MCMC

The parameter determination is primearily for theories with a large parameter space. The reason for this is if the parameterspace is split into equidistantly spaced parametervalues the total number will go as the product and become very large. An example is in cosmology where their models contains up to 13 fitting parameter it is faster to use the MCMC algorithm to map out the parameter space instead of calculation the fit for a lot of predetermined points. The shape around the best fitting values contains information of the uncertainty on the different parameters.

0.8 Fitting with the MCMC algorithm

From a created set of gaussian distributed numbers with a mean value 4 and a variance 2 the MCMC algorithm will move around in parameterspace and determine the best parametervalues. The result can be seen in figure (4). The random walk around in parameterspace needs a way to compare the fits of different gaussian functions. The comparison is done by

```
-make a histogram of the initial dataset.
-for each step in parameterspace generate a dataset from the theory
and make a histogram from the data.
-compare this histogram with the initial in the L2-norm.
-use the L2-norm from the previous step to determine if the step is take
by the rules of the metropolis-hastings algorithm.
-save the parameters from the walk around parameterspace.
```

In figure (4) the steplength is low so the path through the parameterspace can be followed. I can be seen that it converges towards the correct parameter values.

.1 Fitting and MCMC

In probability theory the Baye's theorem states a relation between conditional probabilities. If the probability of an event A is dependent on another event B. The probability of A given B can be shown to be

$$P(A|B) = \frac{P(B|A)\,P(A)}{P(B)} \tag{4}$$

If the events are spinflip in lattice atoms the probability of the flips depend on the neighboring atoms spins. In this example Baye's theorem becomes expressed by the statement of detailed balance.

$$P(\uparrow \mid \downarrow) = \frac{P(\downarrow \mid \uparrow)\,P(\uparrow)}{P(\downarrow)} \tag{5}$$

Where $P(\uparrow \mid \downarrow)$ describes the probability of going from spin down to spin up and vise versa with $P(\downarrow \mid \uparrow)$. And $P(\downarrow)$ are the probability of being in the spin down state. Another maybe more abstract example is the relation between the uncertainty of discrete data sets and the uncertainty in the determination of the theoretical parameters and vise versa. If the event/data are described by B and the parameter/parameters by Θ

$$P(B|\Theta) = \frac{P(\Theta|B)\,P(B)}{P(\Theta)} \tag{6}$$

The probability $P(B|\Theta)$ is also called the likelihoodfunction.

$$L(\theta) = P(B|\Theta = \theta) \tag{7}$$

If it is possible to calculate something proportional to the rhs of equation (5) the metropolis-hasting algorithm can be applied again to find the maximum likelihhod in parameterspace. And the fisher matrix formalism can be used to describe the uncertainties of the parameters.

made by Ole Lund Frederiksen