# Project 1 STAT5376

**Bayesian Analysis of Noisy Images**

Li Sun

September 28, 2016

1. Goal: Given observed noisy images, our goal is to perform a Bayesian analysis of this data. We will assume a prior probability model and an observation model to obtain a posterior density, and will generate samples from the posterior.

2. Models
(a)Prior Model:

$$f(I_{i,j}|\text{all other pixels}) = f(I_{i,j}|I_{i,j-1}, I_{i-1,j}, I_{i+1,j}, I_{i,j+1}) = N(m, \sigma_1^2)$$

All conditional density of a pixel of this m by n Markov Random field only depends on 4 adjacent pixels, 3 for edge pixels and 2 for corner pixels. The m is mean of all neighbors

(b)Observation Model:

$$D = I + W$$

where each element of $W$ is an independent normal random variable with mean 0 and variance $\sigma_2^2$. This specifies the conditional likelihood function

$$f(D|I) \sim N(D_{i,j}, \sigma_2^2)$$

(c)So we can have our conditional posterior density

$$f(I|D) = f(D|I)f(I)$$

Given the structure of $f(I)$ and independence of elements of W, the full conditional of this posterior density can be written as:

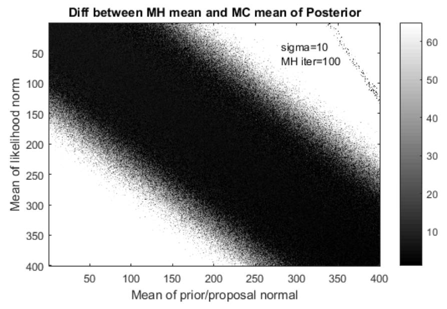
$$f(I_{i,j}|D, \text{all other pexels}) \propto N(m, \sigma_1^2)N(D_{i,j}, \sigma_2^2)$$

3.To generate sample from full posterior, which is not available, we can use Gibbs Sampler to sample from full conditionals sequentially. However, the full conditional density is not a common density which we could sample easily. So we will use MH method for each pixel.
To sample from full conditionals using MH, we chose

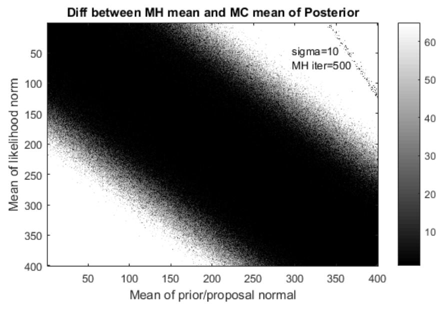
$$q(x) \sim N(m, \sigma_1^2)$$

as our proposal for convenience of calculation. So the acceptance rate is calculated as:

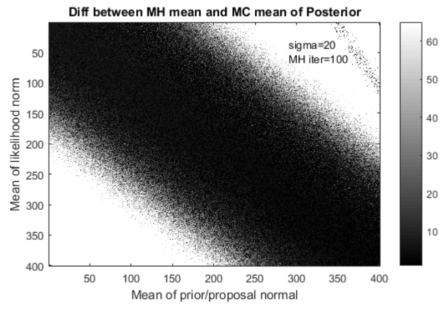
$$\begin{aligned} \rho(x,y) &= min\left\{1, \frac{f(y)q(x)}{f(x)q(y)}\right\} \\ &= min\left\{1, exp(\frac{1}{2\sigma_2^2}((x - D_{i,j})^2 - (y - D_{i,j})^2))\right\} \end{aligned}$$



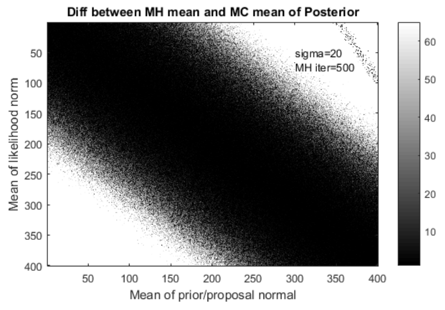
(a) $\sigma_1$=10 with 100 MH iterations



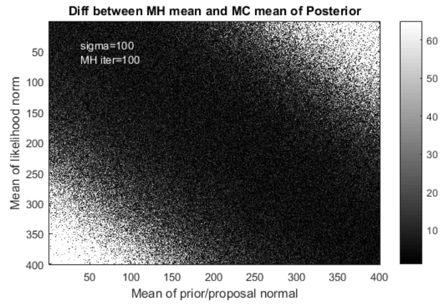
(b) $\sigma_1$=10 with 500 MH iterations



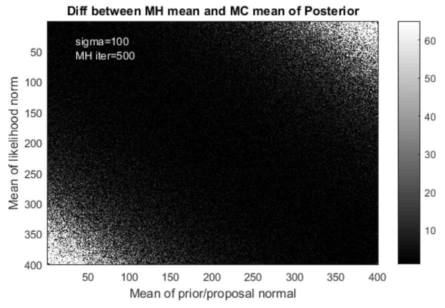
(c) $\sigma_1$=20 with 100 MH iterations



(d) $\sigma_1$=20 with 500 MH iterations



(e) $\sigma_1$=100 with 100 MH iterations



(f) $\sigma_1$=100 with 500 MH iterations

Figure 1: caption

To make our simulation more efficient, we better make our MH algorithm more efficient. So we should do some tests on this MH algorithm. We tested how it perform under several different $\sigma_2(10, 20, 100)$, $m$, $D_{i,j}()$ and different iterations of MH process.
Our aim is to sample from our target density, so we want to make sure the Markov chain produced by our MH algorithm converges before we take values. As a simple way of checking this, we check the convergence of the mean of our target posterior density

$$\frac{1}{Z}N(m, \sigma_1)N(D_{i,j}, \sigma_2^2)$$

The 'true mean' of this density is determined by Monte Carlo integration and the difference between this 'true mean' and our converged mean from our MH Markov chain was calculated based on changing factors of $\sigma_2(10, 20, 100)$, $m$, $D_{i,j}()$ and two different iteration numbers(100 and 500). The result is as followed:

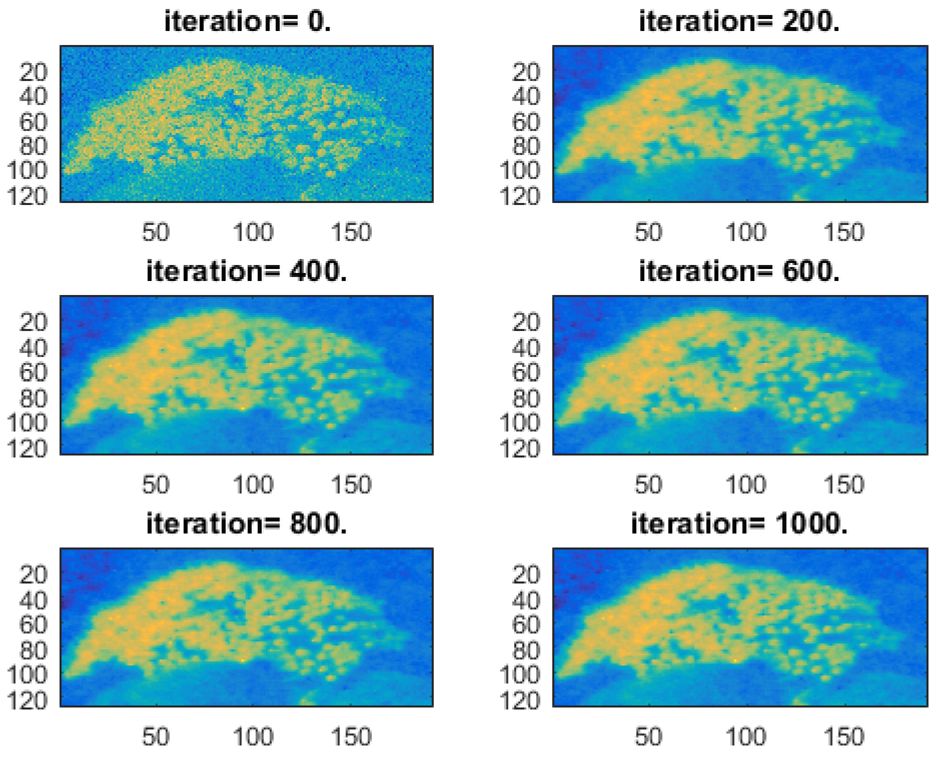
Conclusion is that, our MH algorithm works reasonably well when difference between $m$ and $D_{i,j}$ are close(the difference are smaller than 100) in all cases. 100 iteration seem to perform well enough if the two means are close enough. And we know that

$$D = I + W$$

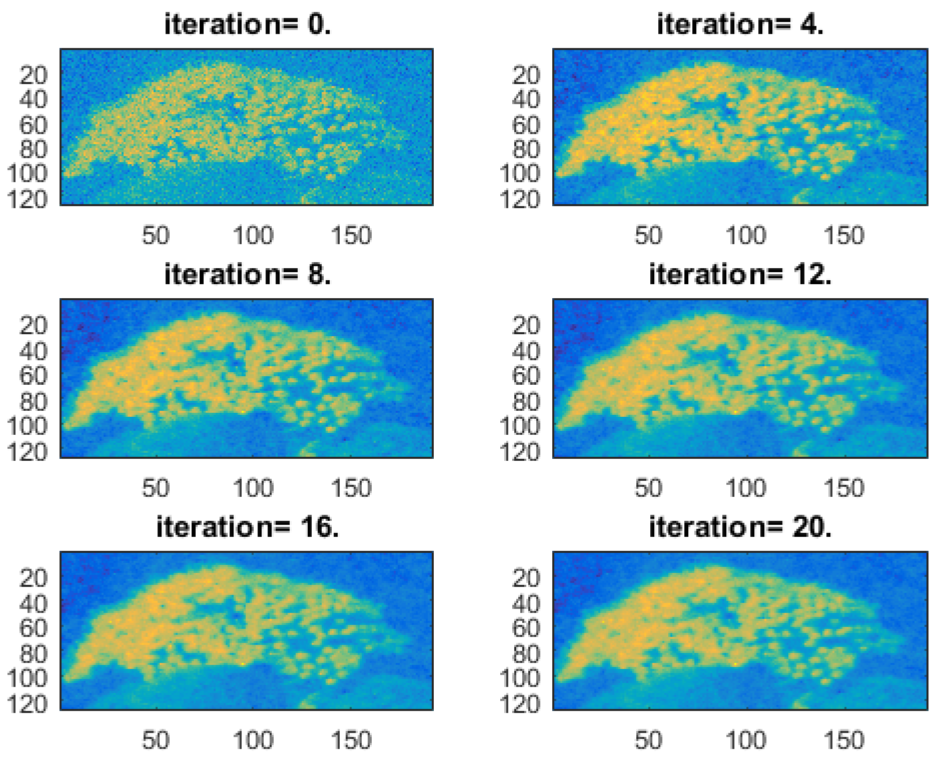
and the standard deviation of W is 30. So the possibility that difference between D and I are larger than 100 is slime. So we will use this MH algorithm and take 100th number as our sampled value each time.

4. Choose the initial value. In order for Gibbs sampling converge quickly, I will set my best guess, which is the noisy image as initial values.

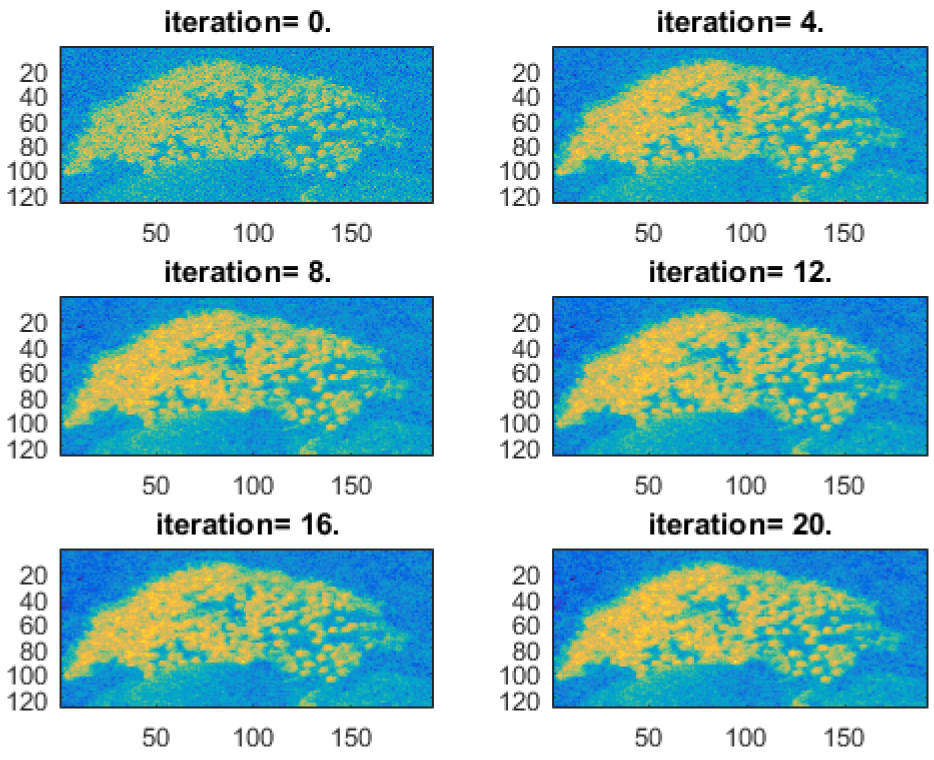
5. Results.
(a)When $\sigma_1 = 10$:
I start using 1000 iteration for Gibbs sampling. And the evolution of D1 when $\sigma_1 = 10$ is as followed



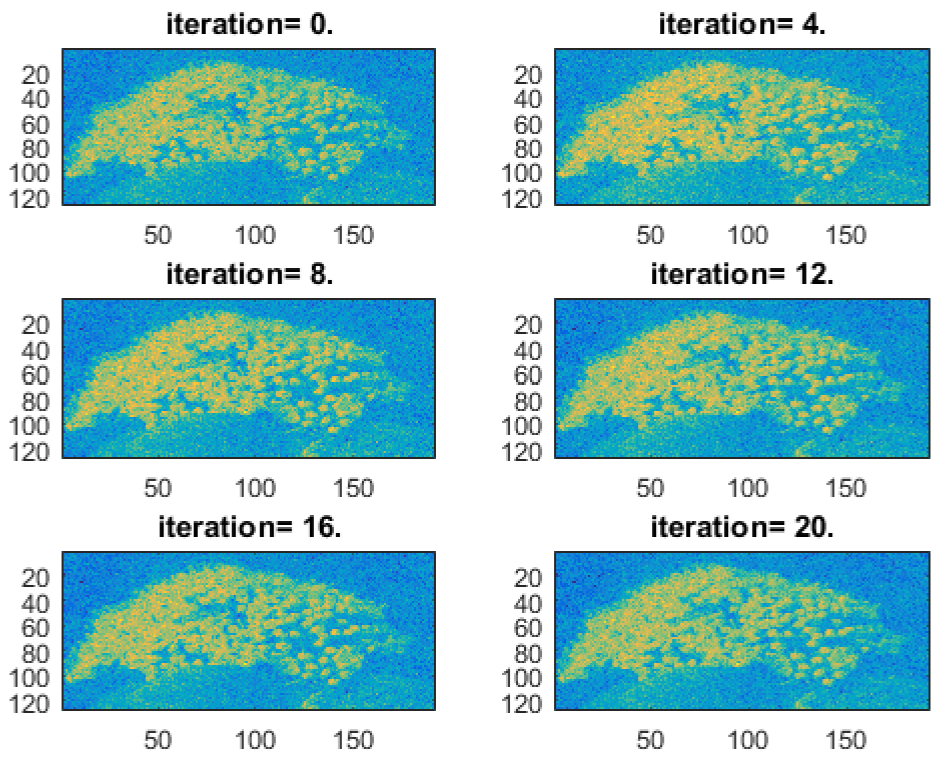
What we observed is much more smooth picture after sampling. However, As you can see that after 200 iteration, the change of the picture are undetectable, which means it converged. So to see the evolution of pictures, we use less iterations instead (20).



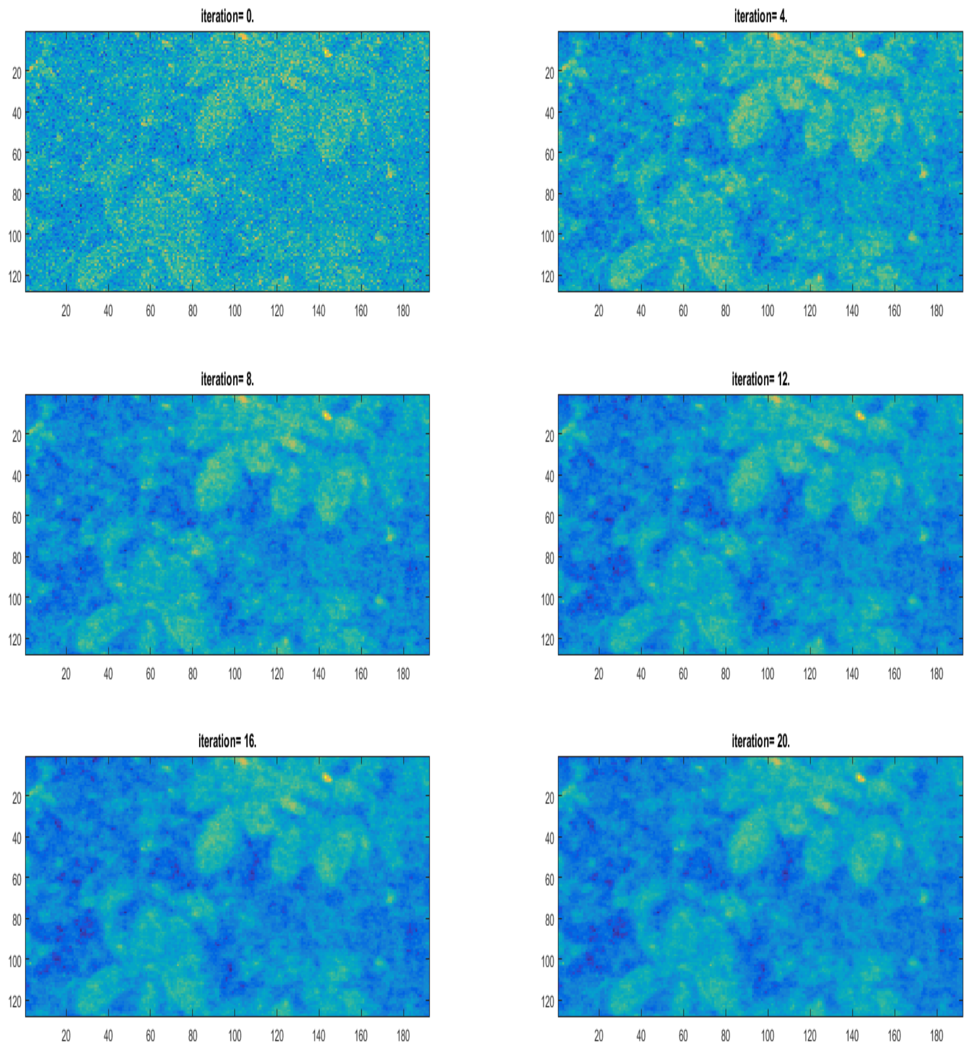
I am surprised how quickly this chain converge, or it is because the difference later is just undetectable to our eyes?
(b)When $\sigma_1 = 20$:



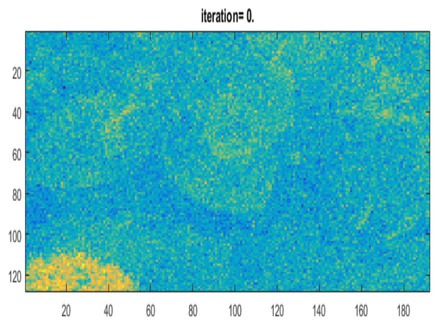
It looks like images converge slower.
(c)When $\sigma_1 = 100$:

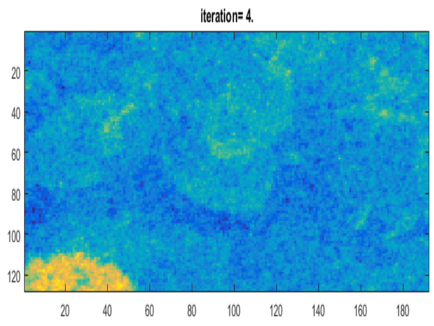


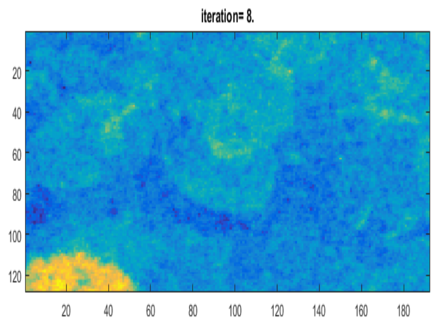
It converges much worse. So for smaller $\sigma 1$, we have faster convergence. This applys to all other picture, so I pick only $\sigma_1 = 10$ and use 20 iteration to see the evolution of pictures. The results see later pages.

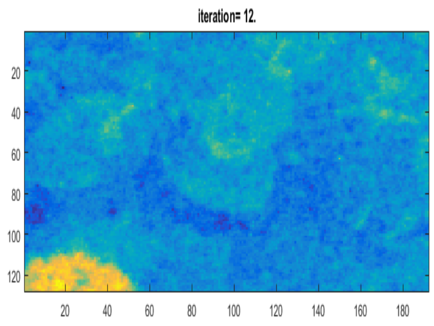


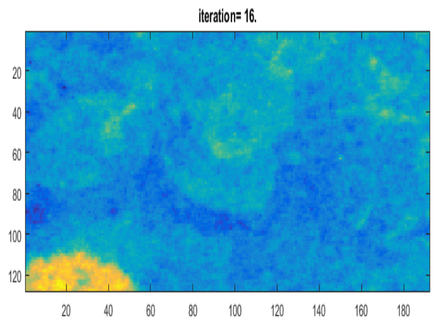
iteration= 0.
iteration= 4.
iteration= 8.
iteration= 12.
iteration= 16.
iteration= 20.
20
40
60
80
100
120
140
160
180

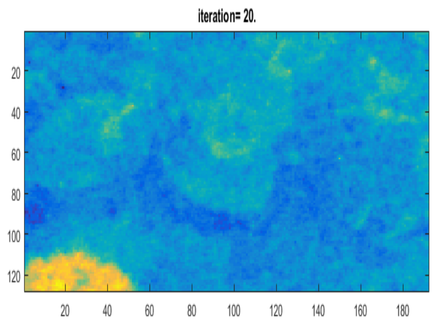

iteration= 0.

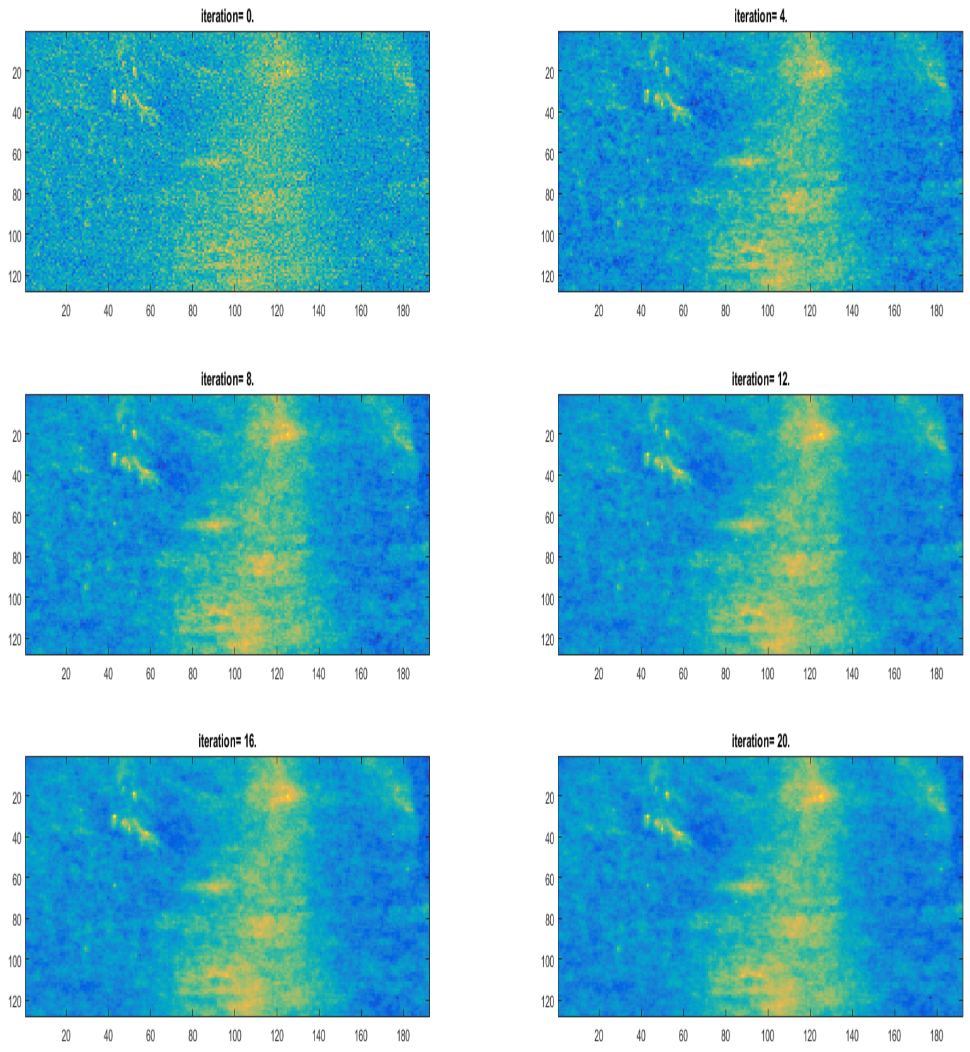

iteration= 4.

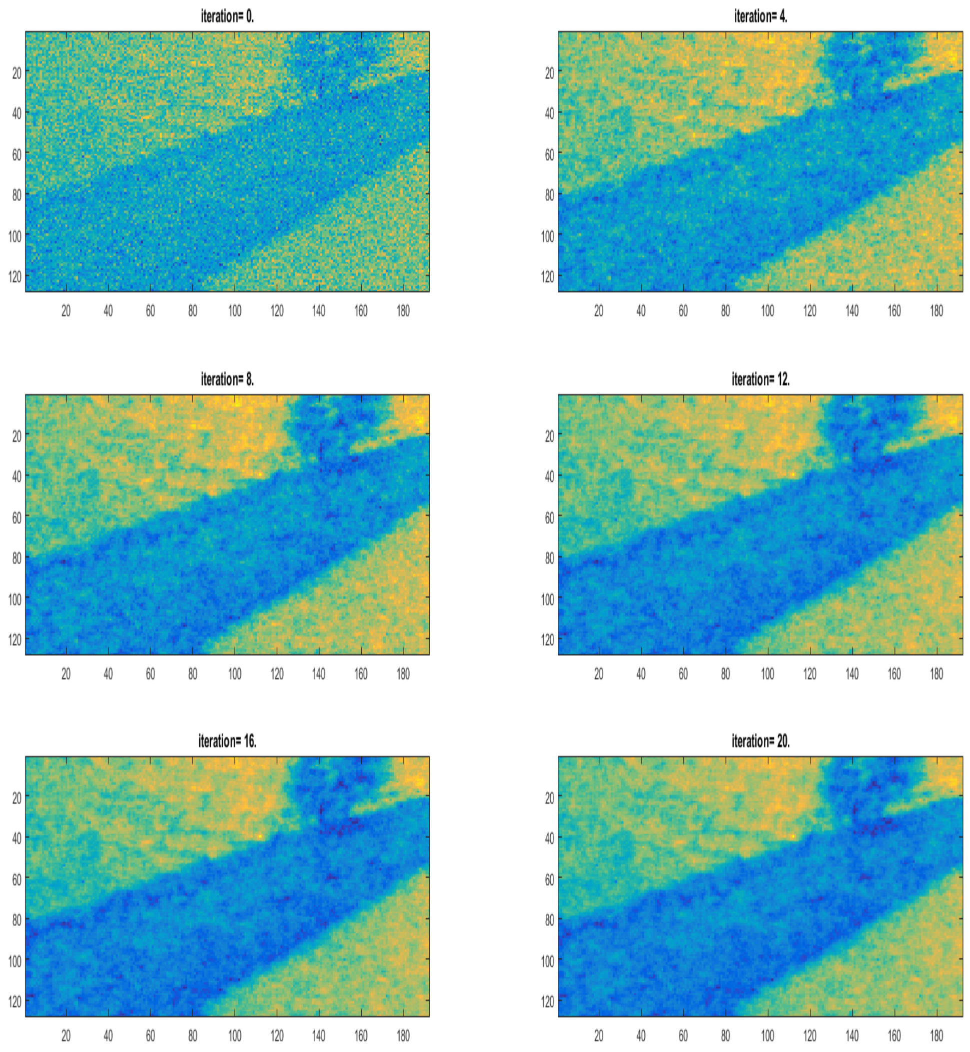

iteration= 8.


iteration= 12.


iteration= 16.


iteration= 20.


iteration= 0.
iteration= 4.
iteration= 8.
iteration= 12.
iteration= 16.
iteration= 20.
20
40
60
80
100
120
140
160
180



Above all, this MCMC program simulates the posterior density of our bayesian model about our pictures. And after iteration of convergence, the noisy figure got smoothed reasonably well under small variance.

All code please see https://github.com/rikku1983/STAT5376

Thanks!

```
%Project 1
    clear all;
    close all;
    load('DataFile1.mat');
    load('DataFile2.mat');
    load('DataFile3.mat');
    load('DataFile4.mat');
    load('DataFile5.mat');
    imagesc(D1);
    imagesc(D2);
    imagesc(D3);
    imagesc(D4);
    imagesc(D5);
    subplot(321);hist(reshape(D1, 192*128,1),200);title('D1');
    subplot(322);hist(reshape(D2, 192*128,1),200);title('D1');
    subplot(323);hist(reshape(D3, 192*128,1),200);title('D1');
    subplot(324);hist(reshape(D4, 192*128,1),200);title('D1');
    subplot(325);hist(reshape(D5, 192*128,1),200);title('D1');


    %Validify MH algorithm for sigma1=10
    m=-50:350;
    d=-50:350;
    for i=1:401;
        for j=1:401;
            tm(i,j)=MC2norm(i-51,10,j-51,30,10000);
            mcm(i,j)=MH(600,i-51,10,j-51,30,500);
        end
    end
    image(abs(mcm-tm));colorbar;xlabel('Mean of prior/proposal normal');ylabel('Mean of likelihood norm');title('Diff between MH mean and MC mean of Posteri
    m=-50:350;
    d=-50:350;
    for i=1:401;
        for j=1:401;
            tm(i,j)=MC2norm(i-51,10,j-51,30,1000);
            mcm(i,j)=MH(600,i-51,10,j-51,30,100);
        end
    end
    image(abs(mcm-tm));colorbar;xlabel('Mean of prior/proposal normal');ylabel('Mean of likelihood norm');title('Diff between MH mean and MC mean of Posteri

    %Validify MH algorithm for sigma1=20
    m=-50:350;
    d=-50:350;
    for i=1:401;
        for j=1:401;
            tm(i,j)=MC2norm(i-51,20,j-51,30,1000);
            mcm(i,j)=MH(600,i-51,20,j-51,30,500);
        end
    end
    image(abs(mcm-tm));colorbar;xlabel('Mean of prior/proposal normal');ylabel('Mean of likelihood norm');title('Diff between MH mean and MC mean of Posteri
    m=-50:350;
    d=-50:350;
    for i=1:401;
        for j=1:401;
            tm(i,j)=MC2norm(i-51,20,j-51,30,1000);
            mcm(i,j)=MH(600,i-51,20,j-51,30,100);
        end
    end
    image(abs(mcm-tm));colorbar;xlabel('Mean of prior/proposal normal');ylabel('Mean of likelihood norm');title('Diff between MH mean and MC mean of Posteri

    %Validify MH algorithm for sigma1=100
    m=-50:350;
    d=-50:350;
    for i=1:401;
        for j=1:401;
            tm(i,j)=MC2norm(i-51,100,j-51,30,1000);
            mcm(i,j)=MH(600,i-51,100,j-51,30,500);
        end
    end
    image(abs(mcm-tm));colorbar;xlabel('Mean of prior/proposal normal');ylabel('Mean of likelihood norm');title('Diff between MH mean and MC mean of Posteri
    m=-50:350;
    d=-50:350;
```

```
for i=1:401;
    for j=1:401;
        tm(i,j)=MC2norm(i-51,100,j-51,30,1000);
        mcm(i,j)=MH(600,i-51,100,j-51,30,100);
    end
end
image(abs(mcm-tm));colorbar;xlabel('Mean of prior/proposal normal');ylabel('Mean of likelihood norm');title('Diff between MH mean and MC mean of Posteri


%Gibbs Sampling
%Initialization
clear all;
close all;
sig1=10;
nr=128;
nc=192;
npic=5;
m=NaN(nr+2,nc+2);
load('DataFile5.mat')
m(2:nr+1,2:nc+1)=D5;
finalm(:,:,1)=m(2:nr+1,2:nc+1);
steps=20;
for n=1:steps
   for i=2:(nr+1)
       for j=2:(nc+1)
           %sample from density(full conditionals)
           im=mean([m(i-1,j),m(i+1,j),m(i,j-1),m(i,j+1)],'omitnan');
           m(i,j)=MH2(1/2*(im+D5(i-1,j-1)),im,sig1,D5(i-1,j-1));
       end
   end
   finalm(:,:,n+1)=m(2:nr+1,2:nc+1);
end
save('D1_1000m.mat', 'finalm');
%Draw figures
npic=6;
for i=1:npic
    intv=floor(20/(npic-1));
    subplot(3,2,i);imagesc(mean(finalm(:,:,1:max(1,intv*(i-1))),3));title(['iteration= ' num2str(intv*(i-1)) '.']);
end
```

```
function [conm,x,r,conma] = MH(x0, m, sig1, d, sig2, step)
x(1)=x0;
%rej=0;
for n=1:step
    %Generate y from N(m, sigma1)
    y=normrnd(m,sig1);
    rho=min(1,exp(((x(n)-d)^2-(y-d)^2)/(2*sig2^2)));
    r(n)=rho;
    if rand()>rho
        x(n+1)=x(n);
        %rej=rej+1;
    else
        x(n+1)=y;
    end
end
%rejr=rej/step;
conma=cumsum(x)./(1:step+1);
conm=conma(step+1);
%tm=MC2norm(m,sig1,d,sig2,10000);
%close all;
%subplot(221);plot(1:step+1,x);title('trace of x');hold on; plot(get(gca,'xlim'),[tm,tm]);
%subplot(222);plot(1:step+1,conm);title('convergence of mean');;hold on; plot(get(gca,'xlim'),[tm,tm]);
%subplot(223);plot(1:step,r);title('trace of acceptance ratrio');
```

```
function [conm,x,r,conma] = MH2(x0, m, d, sig2)
sig1=
step=100;
x(1)=x0;
%rej=0;
for n=1:step
    %Generate y from N(m, sigma1)
    y=normrnd(m,sig1);
    rho=min(1,exp(((x(n)-d)^2-(y-d)^2)/(2*sig2^2)));
    r(n)=rho;
    if rand()>rho
        x(n+1)=x(n);
        %rej=rej+1;
    else
        x(n+1)=y;
    end
end
conma=cumsum(x)./(1:step+1);
conm=conma(step+1);
```

```
function [avg,mcm,y,z]=MC2norm(m, sig1, d, sig2, step2)
    %function of calculating mean of distribution of product of two normal
    %density with mean m and d, variance sig1^2 and sig2^2.
    y=normrnd(d,sig2,1, step2);
    z=1/sqrt(2*pi*sig1^2)*exp(-1/2/(sig1^2).*((y-m).^2));
    mcm=y.*(1/sqrt(2*pi*sig1^2)*exp(-1/2/(sig1^2).*((y-m).^2)));
    %subplot(221);hist(y,200);title('y');
    %subplot(222);hist(z,200);title('z');
    %subplot(223);hist(mcm,200);title('mcm');
    avg=mean(mcm)/mean(z);
```