# 1 Markov Random Fields.

Markov random fiels is $n$-dimensional random process defined on a discrete lattice. Usually the lattice is a regular 2-dimensional grid in the plane, finite or infinite.

Assume that $X_n$ is a Markov Chain taking values in a finite set. Then,

$$P(X_n = x_n | X_k = x_k, k \neq n) = P(X_n = x_n | X_{n-1} = x_{n-1}, X_{n+1} = x_{n+1}).$$

In other words, full conditional distribution of $X_n$ depends of only on the neighbors $X_{n-1}$ and $X_{n+1}$. In the 2-D setting, assume that $S = \{1, 2, \dots, N\} \times \{1, 2, \dots, N\}$ is the set of $N^2$ points, called sites.

For a fixed site $s$ define a neighborhood $\mathbb{N}(s)$. For example for the site $(i, j)$ the neighborhood could be $\mathbb{N}((i, j)) = \{(i-1, j), (i+1, j), (i, j-1), (i, j+1)\}$, where one may take $(0, j) = (N, j), (N+1, j) = (1, j), (i, 0) = (i, N)$ and $(i, N+1) = (i, 1)$.

Markov Property of $X(S)$ is defined via local conditionals,

$$P\left(X(s) = x_s | X(S \backslash s) = x_{S \backslash s}\right) = P(X(s) = x_s | X(\mathbb{N}(s)) = x_{\mathbb{N}(s)}),$$

where $\mathbb{N}(s)$ is the neighborhood of $s$.

Define *clique* $c$ as any set of sites, such that if $s_i, s_j \in c$ then $s_j$ is in the neighborhood of $s_i$, $s_j \in \mathbb{N}(i)$, and of course, $s_i$ is in the neighborhood of $s_j$, $s_i \in \mathbb{N}(j)$. Let $\mathbb{C}$ be the set of cliques.

A set of random variables $X(S)$ is said to be a Gibbs random field (GRF) on $S$ with respect to system of neighborhoods $\mathbb{N}$ if and only if its configurations obey a Gibbs distribution. A Gibbs distribution is defined as

$$P(X) = \frac{1}{Z} \exp\{U(X)/T\},$$

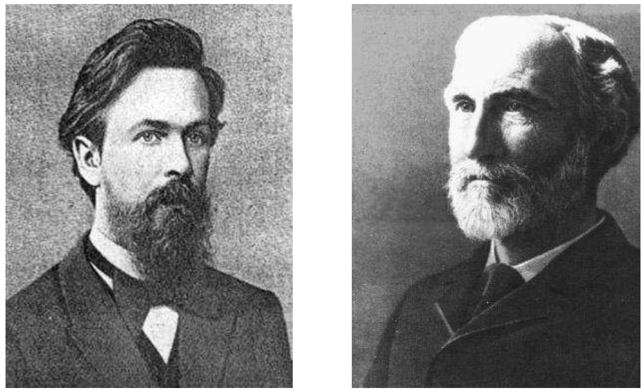
where $Z$ is a normalizing constant called the *partition function*, $T$ is a constant called the *temperature* and $U$ is the energy function. The energy $U(X) = \sum_{c \in \mathbb{C}} V_c(X)$ is a sum of clique potentials $V_c$ over all possible cliques $c \in \mathbb{C}$. The value of depends on the local configuration on the clique $c$. Obviously, the Gaussian distribution is a special member of this Gibbs distribution family.

A GRF is said to be homogeneous if is independent of the relative position of the clique $c$ in $S$. It is said to be isotropic if is independent of the orientation of $c$. It is considerably simpler to specify a GRF distribution if it is homogeneous or isotropic than one without such properties. The homogeneity is assumed in most MRF vision models for mathematical and computational convenience. The isotropy is a property of direction-independent regions.

An MRF is characterized by its local property (the Markovianity) whereas a GRF is characterized by its global property (the Gibbs distribution). The Hammersley-Clifford theorem (Hammersley and Clifford, 1971) establishes the equivalence of these two types of properties. The theorem states that $X(S)$ is a MRF on $S$ with respect to $\mathbb{N}$, if and only if $X(S)$ is a GRF on $S$ with respect to $\mathbb{N}$.

## 1.1 Ising Model.

The classical model for explaining the magnetic behavior is the Ising Model. This model was proposed in the 1924 doctoral thesis of Ernst Ising, a student of W. Lenz. Ising tried to explain certain empirically



(a) (b)

Figure 1: (a) Andrei Andreyevich Markov (1856 - 1922); (b) Josiah Willard Gibbs (1839 - 1903)

observed facts about ferromagnetic materials using a model of proposed by Lenz (1920). It was referred to in Heisenberg's (1928) paper which used the exchange mechanism to describe ferromagnetism.

The Ising Model considers an idealized system of interacting particles, arranged onto a regular planar grid. Each particle can have one of two magnetic spin orientations, generally labeled **up** (+1) and **down** (-1), as in Figure 2 (b). Each particle interacts only with its nearest neighbors; the contribution of each particle to the total energy of the system depends upon the orientation of its spin compared to its neighbors. Adjacent particles that have the same spin $(-1, -1)$ or $(1, 1)$ are in a lower energy state that those with antithetic spins $(1, -1)$ or $(-1, 1)$. Given the spin orientations of all particles in the system, one may compute the total energy. If the variable $x_i$ denotes the spin of particle $i$, then the total energy of the system (Hamiltonian) is

$$E = -J \sum_{x_i \sim x_j} x_i x_j,$$

where the sum is over all neighboring pairs of particles, $x_i \sim x_j$. The constant $J$ represents the strength of the interaction and when positive the agreement between $x_i$ and $x_j$ decreases the energy. Suppose that the set $\mathcal{C}$ represents all possible configurations, and $E(c)$ is the energy of configuration $c \in \mathcal{C}$. Statistical mechanics states that the probability of any particular configuration $c$ is

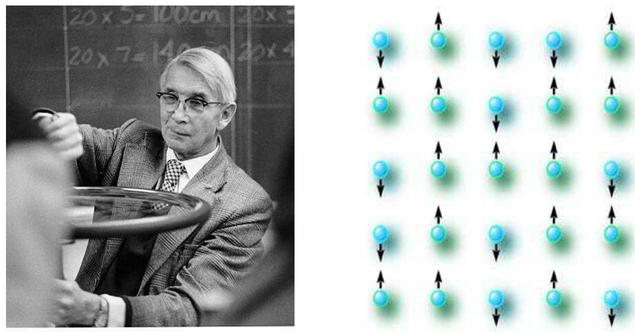
$$\frac{1}{Z} \exp\left\{-\frac{E(c)}{kT}\right\}.$$

where $Z$ is a normalization constant:

$$\sum_{c \in \mathcal{C}} \exp\left\{-\frac{E(c)}{kT}\right\}$$

The parameter $T$ is temperature in $^\circ K$. Low energy configurations are more probable, but the influence of energy is more pronounced at low temperatures. The value $k$ is the Boltzmann's constant[1], a fundamental constant in physics. At low temperatures, the energy of a configuration is very important in determining its likelihood, and so the most likely states are those with lowest energy. At high temperatures, energy is less important, and so the states with high entropy are not unlikely, in fact there are far more states that are disordered than ordered.

[1] $k = 1.3806503 \times 10^{-23} m^2\ kg\ s^{-2}\ K^{-1}$



(a) (b)

Figure 2: (a) Ernst Ising (1900–1998); (b) Ising Model with two types of sites: **up**(+1) and **down** (-1).

## 1.2 Ising Model via Metropolis.

Consider Binary Random Field on a $N \times N$ regular lattice in $R^2$. Assume that the states (pixels) at position $(i, j)$ can take values $x_{i,j} = -1$ or $x_{i,j} = 1$. The double indexing can be recoded to univariate indexing by $(i, j) \mapsto n = i + (N - 1) \times j$.

According to the Ising model

$$P(x) = \frac{1}{Z} \exp\{J \sum_{i \sim j} x_i x_j\},$$

where $x$ is a particular $N \times N$ configuration of $\pm 1$, $x_i \sim x_j$ are pairs of neighboring pixels, and $Z$ is normalizing constant, infamous partition function. The Metropolis algorithm will be used to simulate from this distribution.

Since each pair of neighboring pixels corresponds to one edge inside $N \times N$ regular grid, there is $N - 1$ vertical edges in each of $N$ rows, i.e., $N(N - 1)$ vertical boundary edges. Because of symmetry, the number of horizontal edges is the same, and there is exactly $2N(N - 1)$ pairs. The expression $\sum_{i \sim j} x_i x_j$ is $2N(N - 1) - 2d_x$ where $d_x$ is the number of disagreeing edges (bordering pixels with values 1 and $-1$) in $x$.

Thus,

$$P(x) \propto \exp\{-2Jd_x\} \quad (= \pi(x), \text{ un-normalized probability}).$$

Start with the space of all configurations $\mathcal{C}$ in which each configuration $x$ is represented as a vector

$$x = (x_1, x_2, \ldots, x_{n-1}, x_n, x_{n+1}, \ldots, x_{N^2}),$$

by the indexing $(i, j) \mapsto n = i + (N - 1) \times j$. The Metropolis Algorithm would have the following steps:

1. Start with $x \in \mathcal{C}$.
2. Select a pixel at random from $x$, say $x_n$.
3. Propose new value $x'$ as

$$x = (x_1, x_2, \ldots, x_{n-1}, -x_n, x_{n+1}, \ldots, x_{N^2}).$$

4. Define the probability of move $x \to x'$ is

$$g(x'|x) = 0, \text{ if } x, x' \text{ differ in more than single coordinate or } x = x',$$
$$g(x'|x) = \frac{1}{N^2}, \text{ if } x, x' \text{ differ in exactly one coordinate,}$$

Then, accept $x'$ with probability $\alpha(x'|x)$,

$$\alpha(x'|x) = \min\left\{1, \frac{\pi(x')g(x|x')}{\pi(x)g(x'|x)}\right\},$$

where $\pi(x) = \exp\{-2Jd_x\}$ is unnormalized probability of $x$. Since $g(x|x') = g(x'|x)$, the ratio in $\alpha(x'|x)$ is

$$\exp\{-2J(d_x - d_{x'})\} = \exp\{-2J(d_{x_n} - d_{x'_n})\} = \exp\{-2J(d_{x_n} - a_{x_n})\},$$

where $d_{x_n}$ is the number of disagreeing edges between $x_n$ and pixels from the neighborhood of $x_n$. The number of agreeing edges for $x_n$ is $a_{x_n}$. Note that $d_{x'_n} = a_{x_n}$.

5. Generate uniform random number $u \in \mathcal{U}(0,1)($ and accept $x'$ if $u < \alpha(x'|x)$. Otherwise keep $x$.

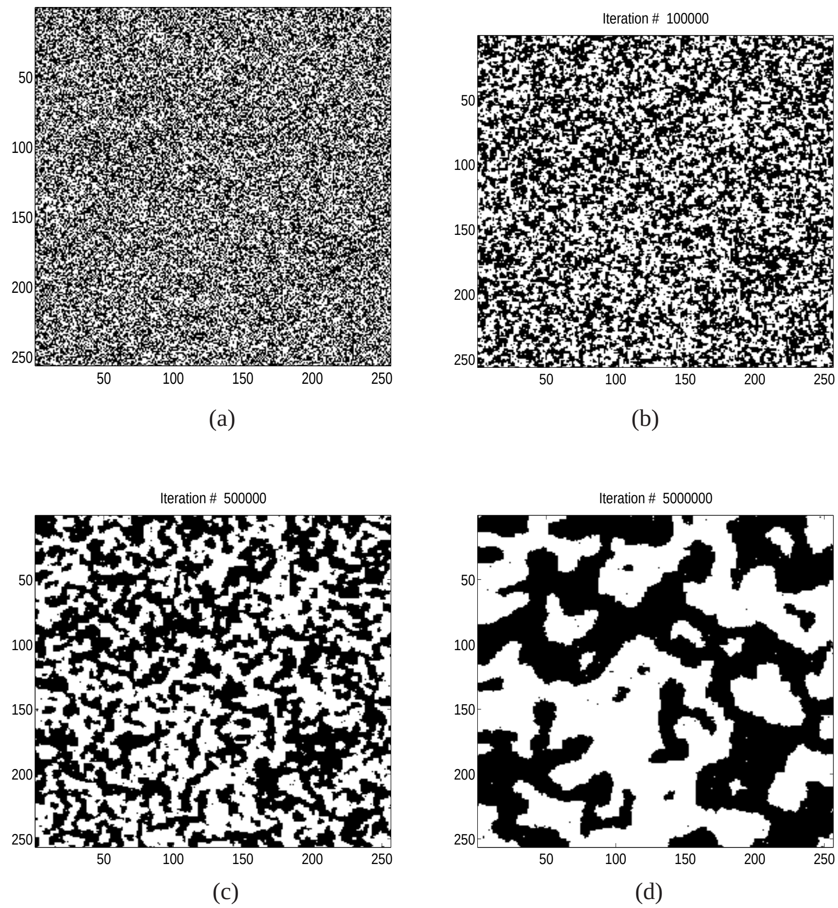



Figure 3: (a) Random Start; (b) After 100,000; (c) 500,000 (d) 5,000,000 steps of Metropolis Algorithm with inverse temperature $J = 0.85$.

## 1.3 Ising Model via (genuine) Gibbs Sampler.

The name Gibbs sampler comes from the following algorithm sampling from Gibbs distribution.

Start with some $x = (x_1, \ldots, x_{N^2})$

Start it selecting $n$ from the index set $\{1, 2, 3, \ldots, N^2\}$ in some order. The simplest is of course the natural order $1, 2, 3, \ldots, N^2, 1, 2, 3, \ldots$

For each pixel-site $n$ compute, $w_n = \sum_{m \sim n} x_m$, where $m$ are sites from the neighborhood of $n$.

**Exercise:** Show that $w_n = x_n \times (d_{x_n} - a_{x_n})$ where $d_{x_n}$ (or $a_{x_n}$) is the number of pixels in the neighborhood $\mathbb{N}(n)$ that disagree (or agree) with the pixel at site $n$, $x_n$.

Define

$$p^- = \frac{\exp\{-\beta w_n\}}{\exp\{-\beta w_n\} + \exp\{\beta w_n\}} \quad \text{and } p^+ = \frac{\exp\{\beta w_n\}}{\exp\{-\beta w_n\} + \exp\{\beta w_n\}}.$$

Now generate $x_n^{new}$ as $x_n^{new} = 1$ with probability $p^+$ and $x_n^{new} = -1$ with probability $p^-$.

The visits to pixels in $x$ could be random instead of ordered, and the convergence to the stationary Gibbs distribution still holds.

**Example:** Figure 1.3 shows Ising model that starts with black-and-white image of Von Klaus, a two year old purebreed [AKC WR021286/04] Doberman Pinscher var. Warlock living in Marietta, Georgia

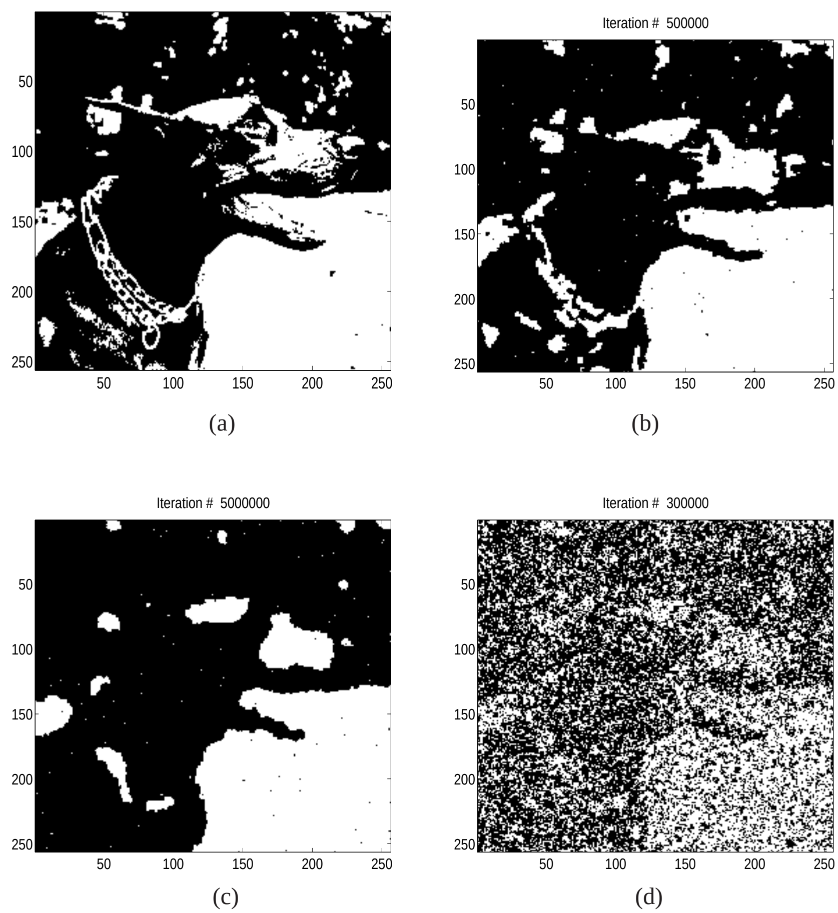



Figure 4: Klaus starts, MRF takes over! Panels (b) and (c) have $\beta = 0.85$, while panel (d) has $\beta = 0.25$.

Panels (b-c) show cooling since reciprocal of temperature $\beta$ is large 0.85. Panel (d) is obtained for $\beta = 0.25$.

### 1.4 Application of Ising Model in Denoising.

Ww will show how the binary Markov random field can serve as a prior in the image denoising and restauration. Consider the problem of reconstructing a black-and-white image (i.e., one in which each pixel is either 1 or -1) from the corrupted observations. The observations are corrupted by independent zero-mean Gaussian noise.

Assume the image $\theta$ be of size $M \times N$ pixels, and denote the value at the $(m,n)$'th pixel by $\theta_{mn} \in \{-1, 1\}$. The state space $\Theta$ of all images $\theta$ has cardinality $2^{MN}$. Assume that the observed data are

$$y_{mn} = \theta_{mn} + \epsilon_{mn},$$

where $\epsilon_{mn} \overset{i.i.d.}{\sim} \mathcal{N}(0, \sigma^2)$.

The likelihood function for this problem is given by

$$f(y|\theta) \sim \exp\left\{-\frac{1}{2\sigma^2}\sum_{m,n}(y_{mn} - \theta_{mn})^2\right\},$$

The estimator for $\theta$ (true image) will be found in Bayesian fashion. We consider two types of priors for $\theta$:

(i) Uniform Prior: $\pi(\theta) = 2^{-MN}$, and

(ii) Ising Model Prior: $\pi(\theta) \propto \frac{1}{Z}\exp\{J\sum_{(m,n)\sim(m',n')}\theta_{mn}\theta_{m'n'}\}$.

The prior in (i) is noninformative, while the prior in (ii) favors clumped images in regions whose size and "clumpiness" depends on the parameter $J$.

When $J = 0$ there is no smoothing, whereas if $J$ is large, we favor a images with a few regions of the same color.

#### 1.4.1 Uniform Prior

With the uniform prior the posterior probability is proportional to the likelihood. A Metropolis MCMC algorithm which draws samples from the such posterior probability has the following steps:

1. Let $\theta_n = \theta$ denote the current state of the Markov chain. Select a pixel with coordinates $(u, v)$ at random and change its color. The transition probability $g(\theta'|\theta)$ is $2^{-MN}$ if $\theta'$ and $\theta$ differ in exactly one pixel and it is zero if $\theta'$ and $\theta$ differ in more than a single pixel.

2. Such proposal density is symmetric and cancels in the Metropolis acceptance ratio,

$$\alpha(\theta'|\theta) = \min\left\{1, \frac{\pi(\theta')f(y|\theta')}{\pi(\theta)f(y|\theta)}\right\},$$

in which for a uniform prior, $\frac{\pi(\theta')f(y|\theta')}{\pi(\theta)f(y|\theta)}$ reduces to the likelihood ratio

$$\begin{aligned}\frac{f(y|\theta')}{f(y|\theta)} &= \exp\left\{-\frac{1}{2\sigma^2}\left[\sum_{m,n}(y_{mn} - \theta'_{mn})^2 - \sum_{m,n}(y_{mn} - \theta_{mn})^2\right]\right\} \\ &\exp\left\{-\frac{1}{2\sigma^2}\left[\sum_{m,n}2y_{mn}(\theta_{mn} - \theta'_{mn}) + (\theta'^2_{mn} - \theta^2_{mn})\right]\right\}\end{aligned}$$

Since, $\theta'_{mn} = -\theta_{mn}$,

$$\begin{aligned}\alpha(\theta'|\theta) &= \min\left\{1, \exp\left\{\frac{y_{mn}(\theta'_{mn} - \theta_{mn})}{\sigma^2}\right\}\right\} \\ &\min\left\{1, \exp\left\{\frac{2y_{mn}\theta_{mn}}{\sigma^2}\right\}\right\}.\end{aligned}$$

**3.** Take $\theta_{n+1} = \theta'$ with probability $\alpha(\theta'|\theta)$, otherwise take $\theta_{n+1} = \theta$.

In the example, it is apparent that uniform prior is not denoising the data $y$ well. For the uniform prior, the fate of each pixel is independent of any other, so no neighboring pixels are used to improve the reconstruction.

### 1.4.2 Ising Prior

An Ising prior on the space of black-and-white images is of the form

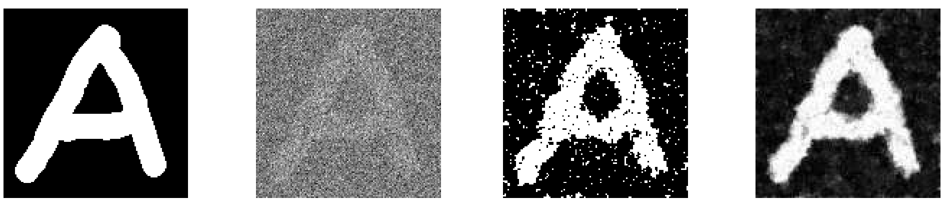
$$\pi(\theta) \propto \exp\{-2Jd_\theta\},$$

where $d_\theta$ is the number of disagreeing edges in the image $\theta$.

With this prior, the ratio

$$\frac{f(y|\theta')\pi(\theta')}{f(y|\theta)\pi(\theta)} = \exp\left\{-\frac{1}{2\sigma^2}\left[\sum_{m,n} -2y_{mn}(\theta'_{mn} - \theta_{mn}) + (\theta'^2_{mn} - \theta^2_{mn})\right]\right\} \times \exp\left\{-2J(d'_\theta - d_\theta)\right\}.$$

enters the Metropolis acceptance probability $\alpha(\theta|\theta) = \min\left\{1, \frac{f(y|\theta')\pi(\theta')}{f(y|\theta)\pi(\theta)}\right\}$.



(a) (b)

Figure 5: (a) Original and noisy **A**. The original is $\{-1, 1\}$ image and the normal noise has size $\sigma = 4$. (b) Posterior after 10000 Metropolis steps and sum of 1000 posteriors each after 10000 steps.

The quantity $d_\theta - d_{\theta'}$ is easy to evaluate for a single-pixel change as it only involves examining at most four edges which link the pixel to its neighbors.

## 1.5 Potts Model.

TBAdded

# References


[1] Cipra, B. A. ”An Introduction to the Ising Model.” Amer. Math. Monthly 94, 937-959, 1987.

[2] Hammersley, J. M. and Clifford, P. (1971). “Markov field on finite graphs and lattices”. unpublished.

[3] Heisenberg, W. ”Zur Theorie des Ferromagnetismus.” Zeitschr. f. Physik 49, 619-636, 1928. [German].

[4] Ising, E. ”Beitrag zur Theorie des Ferromagnetismus.” Zeitschr.. f. Physik 31, 253-258, 1925. [German].

# Matlab Programs

## Ising by Metropolis

```
%Realization of Binary Markov Random Field  by Metropolis
 clear all
 close all
%-----------------figure defaults
disp('Ising by Metropolis')
lw = 2;
set(0, 'DefaultAxesFontSize', 16);
fs = 14;
msize = 5;
randn('state',3) %set the seeds (state) to have
rand ('state',3) %the constancy of results

pixelX = 256;
pixelY = 256;
 J = 0.85;
 el = 0;
F = ( 2 .* ( rand( pixelX, pixelY ) > 0.5 ) - 1 );

while 1,
for k = 1 : 10000
% Select a pixel at random
ix = ceil( pixelX * rand(1) );
iy = ceil( pixelY * rand(1) );
Fc = F( iy, ix ); %the value at position (ix, iy)
pos = ( ix - 1 ) * pixelY + iy; %  univariate index of pixel
neighborhood = pos + [-1 1 -pixelY pixelY]; % Find indicies of neighbours
neighborhood( find( [iy == 1    iy == pixelY    ix == 1   ix == pixelX] ) ) = [];
% pesky boundaries...thrash them
  nagree =  sum( Fc == F(neighborhood) );
  ndisagree = sum( Fc ~= F(neighborhood) );
change = nagree - ndisagree;
if rand(1) < exp( -2 * J * change ) % if change<0, proposal is taken wp 1
    F( iy, ix ) = -Fc;                %accept proposal
end
el = el + 1;
end
figure(2); imagesc(F); colormap(gray); title(['Iteration #  ' num2str(el)]);
drawnow
end
```

## Ising by Gibbs

```
%Realization of Binary Markov Random Field  by Gibbs
 clear all
 close all
%-----------------figure defaults
disp('Ising by Gibbs')
lw = 2;
set(0, 'DefaultAxesFontSize', 16);
fs = 14;
msize = 5;
randn('state',3) %set the seeds (state) to have
rand ('state',3) %the constancy of results
```

```
%----------------

pixelX = 256;
pixelY = 256;
beta = 0.25;
el = 0;
load klaus256; F =  (2.* (ima256 > 0.4) - 1);

%F = ( 2 .* ( rand( pixelX, pixelY ) > 0.5 ) - 1 );%if Klaus is not available...
%while 1,
for jj=1:30
for k = 1 : 10000
% Select a pixel at random
ix = ceil( pixelX * rand(1) );
iy = ceil( pixelY * rand(1) );
Fc = F( iy, ix );
pos = ( ix - 1 ) * pixelY + iy; % Univariate index of pixel
    neighborhood = pos + [-1 1 -pixelY pixelY]; % Find indicies of neighbours
    neighborhood( find( [iy == 1    iy == pixelY    ix == 1   ix == pixelX] ) ) = [];
% Problematic boundaries...
potential = sum(  F(neighborhood) );
   if rand(1) < exp( - beta * potential  )/(  exp( - beta * potential  )...
                   + exp(  beta * potential  ))
    F( iy, ix ) = -1;
   else
    F( iy, ix ) = 1;
   end
el = el + 1;
end
figure(2); imagesc(F); colormap(gray); title(['Iteration #  ' num2str(el)]);
drawnow
end
```

## Denoising by Ising

```
{
%Denoising of letter A: Ising Prior
 clear all
 close all
%-----------------figure defaults
disp('Denoising of A: Ising Prior')
lw = 2;
set(0, 'DefaultAxesFontSize', 16);
fs = 14;
msize = 5;
randn('state',3) %set the seeds (state) to have
rand ('state',3) %the constancy of results

% input matrix consisting of letter A. The body of letter
% A is made of 1's while the background is made of -1's.
F = imread('C:/lettera.bmp'); %or some other path...
[M,N] = size(F);
sigma = 3;
d = double(F); d= 2.*((d-mean(mean(d)))>0)-1; %d either -1 1
% The body of letter
% A is made of 1's while the background is made of -1's.

y = d + sigma*randn(size(d)); %y: noisy letter A, size of the noise is sigma!
```

```
%-----------------------------------------------
figure(1);
subplot(1,2,1);imagesc(d);set(gca,'Visible','off');colormap gray; axis square;
subplot(1,2,2);imagesc(y);set(gca,'Visible','off');colormap gray; axis square;
drawnow

J = 0.5; %Reciprocal Temperature...
theta = ones(M,N); %start with ones
iter=0;
figure(2);
subplot(1,2,1); hf = imagesc(theta); set(gca,'Visible','off');
colormap gray; axis square;  drawnow;
mf = zeros(M,N);
subplot(1,2,2); hm = imagesc(mf); set(gca,'Visible','off');
colormap gray; axis square; drawnow;
SS = 10000;
misfit = [];
adj = [-1 1 0 0; 0 0 -1 1];
while (iter < 10000000)
ix = ceil( N * rand(1) ); iy = ceil( M * rand(1) );
pos = iy + M*(ix-1);
thetap = -theta(pos);
LikRat = exp(y(pos)*(thetap - theta(pos))/sigma.^2);
neighborhood = pos + [-1,1,-M,M];
neighborhood(find([iy==1,iy==M,ix==1,ix==N])) = [];
disagree = sum(theta(neighborhood)~=theta(pos));
disagreep = sum(theta(neighborhood)~=thetap);
DelLogPr = 2 * J * (disagree - disagreep);
alpha = exp(DelLogPr) * LikRat;
   if rand < alpha
       theta(pos) = thetap;
   end
iter = iter + 1;
   if rem(iter,SS) == 0,
        mf = mf+theta; NS = iter/SS; iter
        set(hf,'CData',theta);
        set(hm,'CData',mf); drawnow
        if iter/SS > length(misfit)
               misfit = [misfit,zeros(100,1)];
               misfit(iter/SS) = sum(sum((y-theta).^2))/sigma;
        end
    end
end
}
```