Department of Physics
Computational Inference
Assignment 2 (due 3 Oct by close of play)

1. Apply the Gibbs sampling using 'randomize-then-optimize' method of Bardsley 2012 (SIAM J. Sci. Comput. Vol. 34, pp A1316-A1332) to perform deblurring and hence sharpen up a photograph of jupiter.

   On the Mod 412 website you will find the following three (Matlab – sorry Max) files:

   (a) `jupiter1.tif` which contains a photograph of Jupiter taken in the methane band (780nm) on a grid of size 256×256 pixels, each takes an integer value from 0 to 255. In the upper right-hand portion of the photograph is one of the Galilean satellites. This satellite is close to a point source, and so we can use that region of the photograph as an approximation to the point-spread function.

   (b) `jupiter.m` contains Matlab code to read in the photograph, display it, and extract a 32×32 window of points around the satellite as the point spread function. Note that the function `double` is used to convert the 16 bit integers of the photograph to the "double precision" real numbers used in Matlab.

   (c) `deconv1.m` contains Matlab code to perform regularized deconvolution by Fourier division. This script scans a range of regularizing parameters, plots a labelled L-curve, then displays the deblurred image for a 'best' value of regularizing parameter.

   Run this script: In the deconvolved picture, you should notice the bands around the planet due to convection of the atmosphere. Impacts from the recent (July 1994) collision of the Shoemaker-Levy 9 comet are visible in the upper part of the disc and the Red Spot is on the limb of the planet on the right side. The bright spot in the lower part of the image is the transit of one of the other Galilean satellites.

   Implement the MCMC method in Bardsley 2012 (steps 0 to 4 on p. A1321) to perform sampling over both deblurred image and hyperparameters. Plot a sample image as well as a histogram over the regularizing parameter (ratio of hyperparameters). You should utilze the Fourier representation of convolution to give computational efficiency when implementing the step in Eqn 3.8. Evaluate the integrated autocorrelation time(s) for this chain (for various statistics of interest).

   Challenge question (for interest only): You will see in the regularized deblurred image that there are some non-physical ripples around the deconvolved Galilean satellite. Can you improve the reconstruction by fiddling with the point-spread function? Since the point-spread function is uncertain, you might like to introduce further hyperparameters in the Bayesian formulation to model this variability.

2. An $N \times N$ colour image c is represented as a square array of 3-component vectors. At pixel $i = 1, 2, \ldots, N^2$, $x_i = [x_i^r, x_i^g, x_i^b]$ with x_i^r, x_i^g, and x_i^b respectively red, green, and blue parameters taking values from -1 to 1. For example, $[1, -1, -1]$ is pure red. Denote by Ω the space of all $N \times N$ color images.

   (a) Design and implement a Metropolis Hastings algorithm generating a Markov chain with unique equilibrium distribution

$$\pi(x) \propto \exp\left\{ -J \sum_{i=1}^{N^2} \sum_{j \sim i} |x_i - x_j| \right\} \qquad x \in \Omega$$

   where $\sum_{j \sim i}$ indicates a sum over the image-lattice neighbours j of pixel i and J is a smoothing parameter (try $J = 1$ say).

   (b) In the $O(3)$ colour-vector model each vector is constrained to have length one, $|x_i| = 1$ at each pixel $i = 1, 2, \ldots, N^2$. The density $\pi(x)$ is unchanged, but the new state space is

$$\Omega^{O(3)} = \left\{ x \in \Omega : (x_i^r)^2 + (x_i^g)^2 + (x_i^b)^2 = 1 \text{ for each } i = 1, 2, \ldots, N^2 \right\}$$

   Explain briefly how you would carry out MCMC for the $O(3)$ model. Give a start state, a proposal mechanism and an appropriate acceptance probability.

Note: in answering Question 2 you may find useful the code fraction on the next page.

```
%Lattice
M = 64;
N = 64;
nbrs=GetNbrs(M,N);

%initialize colors to random color vectors of unit length, sample points
%uniformly at random on the surface of a sphere of unit radius
% Use the rotational symmetry of the Normal pdf with identity covariance
r = randn(3,M,N);
rn = sqrt(sum(r.^2));
%x(i,j,k) is a 3 x M x N array, i is the color index,
%j,k are image position indices
x = r./rn([1 1 1],:,:);

%For plotting colours are scaled into [0,1]^3 so matlab can represent them.
%Also we need to rearrange x so that the last index is the colour index.
figure(3);hf=image(permute((x+1)/2,[2,3,1]));axis square;drawnow;

%now pixel is a random pixel on the MxN lattice
pixel = ceil(rand*M*N);
nn=length(nbrs{pixel});

%notice that the two lattice indices j,k running from 1..M and 1..N
[j,k]=ind2sub([M,N],pixel)
%have been replaced by a single index pixel running from 1..M*N
pixel
%the colors at pixel are [x(1,pixel);x(2,pixel);x(3;pixel)]
x(:,pixel)

%pi(x)=exp(-JF(x)), contribution to F from one pixel is
delta=x(:,nbrs{pixel})-repmat(x(:,pixel),1,nn);
F=sum(sqrt(sum(delta.^2)));

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function nbrs=GetNbrs(M,N)

if nargin==1, N=M; end

nbrs=cell(1,M*N);
for k=1:M*N
  [i,j]=ind2sub([M,N],k);
  subnbr=repmat([i;j],1,4)+[0 0 -1 1;1 -1 0 0];
  II=find(subnbr(1,:)>M | subnbr(1,:)<1);
  JJ=find(subnbr(2,:)>N |
  subnbr(2,:)<1);
  subnbr(:,union(II,JJ))=[];
  nbrs{k}=sub2ind([M,N],subnbr(1,:),subnbr(2,:));
end
```