Markov Random Fields

W.G.H.S. Weligampola (E/14/379)

June 2020

1 Introduction

Markov network or undirected graphical model is a countable set of random variables having a Markov property described by an undirected graph. In other words, a random field is said to be a Markov random field (MRF) if it satisfies Markov properties. We will discuss about these Markov in the next section.

A Markov network is similar to a Bayesian network in its representation of dependencies; the differences being that Bayesian networks are directed and acyclic, whereas Markov networks are undirected and may be cyclic. Thus, a Markov network can represent certain dependencies that a Bayesian network cannot. On the other hand, it can't represent certain dependencies that a Bayesian network can (such as induced dependencies). The underlying graph of a Markov random field may be finite or infinite.

When the joint probability density of the random variables is strictly positive, it is also referred to as a Gibbs random field, because, it can then be represented by a Gibbs measure for an appropriate (locally defined) energy function. The Markov random field was introduced as the general setting for the Ising model. The prototypical Markov random field is the Ising model. We will discuss about the Ising model in the applications section. In the domain of artificial intelligence, a Markov random field is used to model various low to mid-level tasks in image processing and computer vision.

2 Local Markov properties

Given an undirected graph $G = (V, E)$, a set of random variables $X = (X_s)$ $s \in S$ indexed by S form a Markov random field with respect to G if they satisfy the local Markov properties:

1. Pairwise Markov property: Any two non-adjacent variables are conditionally independent given all other variables:

   $X_s \perp\!\!\!\perp X_t \mid X_{S \setminus \{s,t\}}$

2. Local Markov property: A variable is conditionally independent of all other variables given its neighbors:

$X_s \perp\!\!\!\perp X_{S \backslash \partial s \cup s} \mid X_{\partial v}$

where ∂s is the set of neighbors of s.

3. Global Markov property: Any two subsets of variables are conditionally independent given a separating subset:

   $X_A \perp\!\!\!\perp X_B \mid X_C$

   where every path from a node in A to a node in B passes through C.

The Global Markov property is stronger than the Local Markov property, which in turn is stronger than the Pairwise one. However, the above three Markov properties are equivalent for a positive probability.

3 Formal definition

A Markov Random Field is a probability distribution $P(X = x)$ of a particular field configuration x in X. Where X are the set of random variables $X = (X_s)_{s \in S}$, defined by an undirected graph G in which nodes correspond to variables X_s. Because X is a set, the probability of x should be understood to be taken with respect to a joint distribution of the X_s.

3.1 Factorization by Maximal Clique

We can define the factors in the decomposition of the joint distribution to be functions of the variables in the cliques (i.e., fully connected subgraphs). Refer appendix A for more details.

If this joint density can be factorized over the cliques of G:

$$p(X = x) = \frac{1}{Z} \prod_{c \in C} \phi_c(x_c) \tag{1}$$

then X forms a Markov random field with respect to G where C denotes the set of cliques of G, and ϕ_c are non-negative functions over the variables in a clique. The functions ϕ_c are sometimes referred to as factor potentials or clique potentials.

The partition function Z given by Eq. (2) is a normalizing constant that ensures that the distribution sums to one.

$$Z = \sum_{x \in \mathbf{X}} \prod_{c \in C} \phi_c(x_c) \tag{2}$$

Where $\mathbf{X}$ denotes the set of all possible assignments of values to all the network's random variables. Thus, given a graph G, our probability distribution may contain factors whose scope is any clique in G, which can be a single node, an edge, a triangle, etc. Note that we do not need to specify a factor for each clique. However, we chose not to specify any unary factors, i.e., cliques over single nodes.

The reason for using a potential function instead of a probability function is because in MRFs there does not exist a parent. But in directed graphs, each factor represents the conditional distribution corresponding to its parents. But here we do not restrict the choice of potential functions to a specific probabilistic distribution for flexibility. We can define any potential function as we want. Note that this is still a probability function. Thus, there are some restrictions for defining a potential function for a clique. A potential function $\phi_c(x_c) \geq 0$, to ensure that $p(x) \geq 0$. Therefore it is usually convenient to express them as exponential as given in Eq. (3).

$$\phi_c(x_c) = \exp(-V(x_c)) \tag{3}$$

Where V is called an potential function, and the exponential representation is called the Boltzmann distribution. Since the joint distribution is defined as the product of potentials, the total energy is obtained by adding the potentials of each of the maximal cliques.

$$p(X = x) = \frac{1}{Z}\prod_{c\in C}\phi_c(x_c) = \frac{1}{Z}\prod_{c\in C}\exp(-V(x_c)) = \frac{1}{Z}\exp(-\sum_{c\in C}V(x_c)) \tag{4}$$

In the next section we will discuss how MRFs are applied in image processing, and how Markov properties are applicable

4 Discrete State Markov Random Fields

4.1 Definition of Neighborhood system

Before we can define an MRF, we must first define the concept of a neighborhood system. Let S be a set of lattice points with elements $s \in S$. Then we use the notation ∂s to denote the neighbors of s. Notice that ∂s is a subset of S, so the function ∂ is a mapping from S to power set of S, or equivalently the set of all subsets of S denoted by 2^S. However, not any mapping ∂s qualifies as a neighborhood system. In order for ∂s to be a neighborhood system, it must meet the following constraints.

$$r \in \partial s \rightarrow s \in \partial r \tag{5}$$

Simply, for all $s, r \in S$ if r is a neighbor of s, then s must be a neighbor of r. Notice that this definition is not restricted to a regular lattice. However, if the lattice S is a regular lattice, and the neighborhood is spatially invariant, then symmetry constraint necessitates that the neighbors of a point must be symmetrically distributed about each pixel. Also, we define clique as a set of points, c, which are all neighbors of each other.

$$\forall s, r \in c, r \in \partial s \tag{6}$$

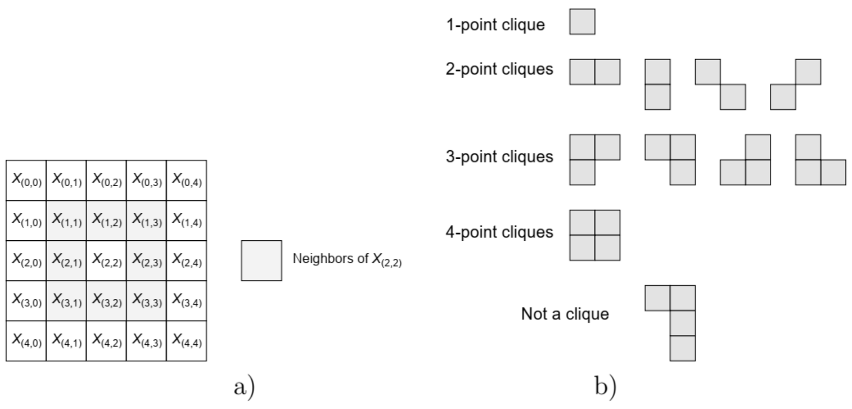
$X_{(0,0)}$ $X_{(0,1)}$ $X_{(0,2)}$ $X_{(0,3)}$ $X_{(0,4)}$
$X_{(1,0)}$ $X_{(1,1)}$ $X_{(1,2)}$ $X_{(1,3)}$ $X_{(1,4)}$
$X_{(2,0)}$ $X_{(2,1)}$ $X_{(2,2)}$ $X_{(2,3)}$ $X_{(2,4)}$
$X_{(3,0)}$ $X_{(3,1)}$ $X_{(3,2)}$ $X_{(3,3)}$ $X_{(3,4)}$
$X_{(4,0)}$ $X_{(4,1)}$ $X_{(4,2)}$ $X_{(4,3)}$ $X_{(4,4)}$
Neighbors of $X_{(2,2)}$
a)
1-point clique
2-point cliques
3-point cliques
4-point cliques
Not a clique
b)

Figure 1: An eight point a) neighborhood system, and b) its associated cliques.

An example of 8 point neighborhood is illustrated in Fig. 1. When modeling images as a MRF, each state (pixel) is accessible from other pixels. Therefore MRF is **irreducible**. Also since the model is undirected states are **periodic**.

4.2 Discrete(Continuous) Markov Random Field

Let $X_s \in \Omega$ be a discrete(continuous) valued random field defined on the lattice S with neighborhood system ∂s. Further assume that the X has probability mass(density) function $p(x)$. Then we say that X is a Markov random field (MRF) if its density function has the property that for all $x \in \Omega$ and $r \neq s$, $p(x_s|x_r) = p(x_s|x_{\partial r})$. Notice that each pixel on is only dependent on its neighbors.

A limitation of MRFs is that their definition does not yield a natural method for writing down the MRF's distribution. For this purpose, we will need to introduce the Gibbs distribution.

4.3 Discrete (Continuous) Gibbs Distribution

Let $p(x)$ be the probability mass(density) function of a discrete(continuous) valued random field $X_s \in \Omega$ defined on the lattice S with neighborhood system ∂s. Then we say that $p(x)$ is a Gibbs distribution if it can be written in the form $p(x) = \frac{1}{Z}\exp(-\sum_{c\in C} V(x_c))$. Where C is the set of all cliques, Z is the partition function, and $V_c(x_c)$ are any functions, of x_c.

4.4 Hammersley-Cliford Theorem

The important result that relates MRFs and Gibbs distributions is the Hammersley-Cliford Theorem stated below.

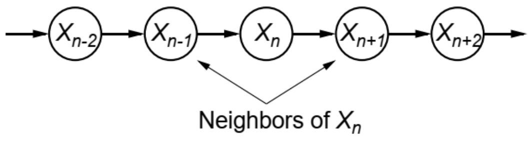

Figure 2: Markov Chain

Let X be a discrete(continuously) valued MRF, and let $p(x)$ be its probability mass(density) function each with lattice S and neighborhood system ∂s. Then X is an MRF if and only if $p(x)$ is a Gibbs distribution.

5 Examples of MRFs

5.1 1-D MRF

1D MRFs are necessarily Markov Chains. Let's look at Markov chain illustrated in Fig. 2. The neighbours of n are $\partial n = \{n-1, n+1\}$ and the cliques have the form $\{n-1, n\}$. The density function can be formulated as,

$$p(x) = p(x_0)\prod_{n=1}^{N} p(x_n|x_{n-1}) = p(x_0)\exp(\sum_{n=1}^{N}\log(p(x_n|x_{n-1}))) \tag{7}$$

Where we can define an potential function as $V(x_n, x_{n-1}) = \log p(x_n|x_{n-1})$. Now in a 1D MRF defined by X_n with $\partial n = \{n-1, n+1\}$, the discrete density has the form of a Gibbs distribution as follows,

$$p(x) = p(x_0)\exp\{\sum_{n=1}^{N} V(x_n, x_{n-1})\} \tag{8}$$

Therefore, we can see that 1D MRFs are Markov chains.

5.2 2D MRF

There are many models proposed in 2D domain. The conditional probability of a pixel x_s in Fig. 3 can be given by Eq. (9).

$$p(x_s|x_{i\neq s}) = \frac{\frac{1}{Z}\exp(-\sum_{c\in C} V_c(x_c))}{\sum_{x_s=0}^{M-1}\frac{1}{Z}\exp(-\sum_{c\in C} V_c(x_c))} \tag{9}$$

5.2.1 Ising model

The Ising model is an example of an MRF that arose from statistical physics. It was originally used for modeling the behavior of magnets. In particular, let $x_s = \{-1, +1\}$ represent the spin of an atom, which can either be spin down or

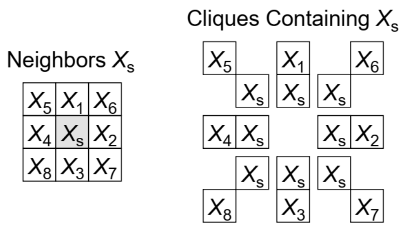

Figure 3: Neighboring pixels of x_s

up. In some magnets, called ferro-magnets, neighboring spins tend to line up in the same direction, whereas in other kinds of magnets, called anti-ferromagnets, the spins “want” to be different from their neighbors.

We can model this as an MRF as follows. We create a graph in the form of a 2D or 3D lattice, and connect neighboring variables. We then define the following pairwise clique potential:

$$V(x_x, x_t) = \begin{pmatrix} e^{w_{st}} & e^{-w_{st}} \\ e^{-w_{st}} & e^{w_{st}} \end{pmatrix} \tag{10}$$

Here w_{st} is the coupling strength between nodes s and t. If two nodes are not connected in the graph, we set $w_{st} = 0$. We assume that the weight matrix W is symmetric, so $w_{st} = w_{ts}$. Often we assume all edges have the same strength, so $w_{st} = J$ (assuming $w_{st} \neq 0$). If all the weights are positive, $J > 0$, then neighboring spins are likely to be in the same state. If the weights are sufficiently strong, the corresponding probability distribution will have two modes, corresponding to the all +1’s state and the all -1’s state. These are called the ground states of the system. Interestingly, computing the partition function $Z(J)$ can be done in polynomial time for associative Markov networks, but is NP-hard in general.

There is an interesting analogy between Ising models and Gaussian graphical models. First, assuming $x_t = \{-1, +1\}$, we can write the unnormalized log probability of an Ising model as follows:

$$\log p(\mathbf{x}) = -\sum_{s\ t} x_s w_{st} x_t = -\frac{1}{2}\mathbf{x^T W x} \tag{11}$$

The factor of 1/2 arises because we sum each edge twice. If $w_{st} = J > 0$, we get a low energy (and hence high probability) if neighboring states agree.

Sometimes there is an external field, which is an energy term which is added to each spin. This can be modelled using a local energy term of the form $-\mathbf{b^T y}$, where $\mathbf{b}$ is sometimes called a bias term. The modified distribution is given by:

$$\log p(\mathbf{x}) = -\sum_{s\ t} w_{st} x_s x_t + \sum_{s} b_s x_s = -\frac{1}{2}\mathbf{x^T W x} + \mathbf{b^T y} \tag{12}$$

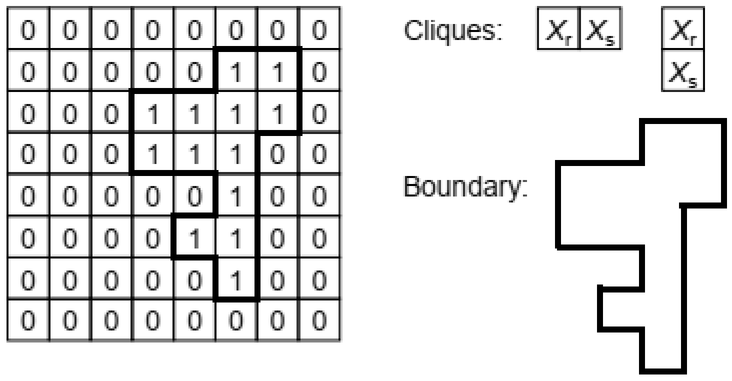

Figure 4: Ising model

If we define $\Sigma^{-1} = W$, $\mu = \Sigma b$, and $c = \frac{1}{2}\mu^T\Sigma^{-1}$, we can rewrite this in a form that looks similar to a Gaussian:

$$p(\mathbf{x}) \propto \exp(-\frac{1}{2}(\mathbf{y} - \mu)^{\mathbf{T}}\mathbf{\Sigma^{-1}}(\mathbf{y} - \mu) + \mathbf{c} \tag{13}$$

One very important difference is that, in the case of Gaussians, the normalization constant, $Z = |2\pi\Sigma|$, requires the computation of a matrix determinant, which can be computed in $O(D^3)$ time, whereas in the case of the Ising model, the normalization constant requires summing over all 2D bit vectors; this is equivalent to computing the matrix permanent, which is NP-hard in general.

5.2.2 Sampling from the model

Gibbs Sampling can be used to draw samples from this distribution as follows: given a sample x, produce a candidate new sample x' by flipping a single variable ($x'_i = -x_i$). Next, compute the acceptance probability:

$$\alpha(x'|x) = min(1, \frac{p(x')}{p(x)})$$

and let the next sample be x' with probability $\alpha(x'|x)$, or repeat x otherwise. Clearly, if $p(x') > p(x)$, the state will transition to x' with certainty. However, if $p(x') < p(x)$, the sample will only be accepted with some probability that is based on how much worse it is. The Gibbs sampling process is illustrated on the Fig. 5, where the potential strengths J, b can be manipulated. When $J > 0$, the low energy states are smooth regions, as this minimizes the number of edges that connect nodes of different values. When J is negative, the reverse happens as the model assigns higher probabilities to states with many crossing edges.

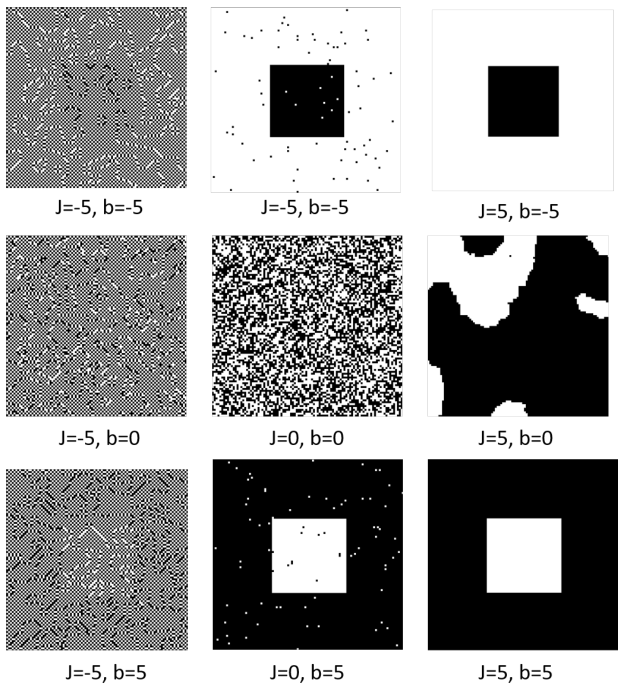

Figure 5: Gibbs sampling of Ising model.

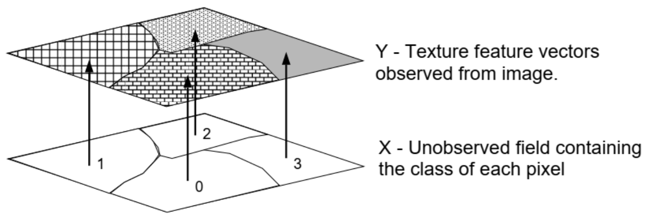
Y - Texture feature vectors observed from image.
X - Unobserved field containing the class of each pixel
1
2
0
3

Figure 6: Segmentation model

6 Applications

Some of the applications of MRFs are:

1. Segmentation
2. Semantic labeling
3. Stereo matching
4. Depth estimation
5. De-noising
6. Super resolution
7. Pose estimation

For this section we will consider Bayesian Segmentation Model. Here we use a discrete MRF to model the segmentation field. Each class is represented by a value $X_s = \{1, ..., M-1\}$. The joint probability of the data and segmentation is $P\{Y \in dy, X = x\} = p(y|x)p(x)$ where $p(y|x)$ is the data model and $p(x)$ is the segmentation field.

For estimating the segmentation lets use the Bayes Theorem. Let $C(x, X)$ be the cost of guessing x when X is the correct answer. Let $\hat{X}$ be the estimated value of X. Then $E[C(\hat{X}, X)]$ is the expected cost(risk). Our objective is to choose the estimator $\hat{x}$ that minimizes $E[C(\hat{X}, X)]$.

Let $C(x, X) = -\delta(x \neq X)$. Then the optimum estimator is given by maximum a posterior (MAP) estimation:

$$\begin{aligned}\hat{X}_{MAP} &= \arg\max_x p_{x|y}(x|Y) \\ &= \arg\max_x \log \frac{p_{y,x}(Y,x)}{p_y(X)} \\ &= \arg\max_x \{\log p_{y,x}(Y,x) + \log p_y(X)\}\end{aligned} \tag{14}$$

Advantage of using MAP is the we an compute through direct optimization. But, the cost function is unreasonable for many applications.

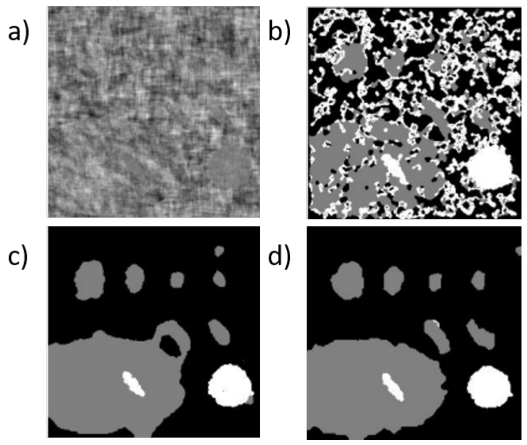

Figure 7: a) Synthetic image with 3 textures b) Iterated Conditional Modes - 29 iterations c) Simulated Annealing - 100 iterations d) Multiresolution - 7.8 iterations

Assume the data model $p_{y|x}(y|x) = \prod_{s\in S} p(y_s|x_s)$ and the prior model (Ising model) $p_y(x) = \frac{1}{Z'}\exp(-\beta t_1(x))$. Then the MAP estimate has the form $\hat{X}_{MAP} = \arg\min_x\{-\log p_{y|x}(y|x) + \beta t_1(x)\}$. Then using an optimization technique we can find the segmentation of the image. Some of the results are illustrated in Fig. 7

Appendices

A Cliques

As the Markov property of an arbitrary probability distribution can be difficult to establish, a commonly used class of Markov random fields are those that can be factorized according to the cliques of the graph. A subset of nodes in a graph G is a clique if,

- There exists a link between all pairs of nodes in the subset.
- The set of nodes in a clique is fully connected.

A maximal clique is a clique that is not possible to include any other nodes from the graph to the set without it ceasing to be a clique. This is illustrated in the Fig. 8.

In the example given in Fig. 8, has five cliques of two nodes $\{x_1, x_2\}$, $\{x_2, x_3\}$, $\{x_3, x_4\}$, $\{x_4, x_2\}$, $\{x_1, x_3\}$. It has two maximal cliques $\{x_1, x_2, x_3\}$, $\{x_2, x_3, x_4\}$. The set $\{x_1, x_2, x_3, x_4\}$ is not a clique because of the missing link from x_1 to x_4.

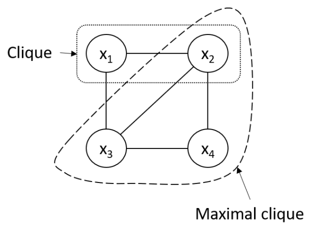

Figure 8: Difference between a clique and a maximal clique.

B Pseudo code: Gibbs sampling of Ising model

Listing 1: Pseudo code

```
// naive gibbs sampler for the ising model
x = randomState()
while true:
 // calculate probability of this state and a proposal
 px = p(x) // p is the un-normalized probability as defined above
 xnew = flipOneBit(x)
 pnew = p(xnew)

 // calculate transition probability alpha
 transitionProbability = min(1, pnew/px)
 if uniformRandom(0,1) < transitionProbability:
  x = xnew // transition to x'!
```

C Python code: Image segmentation using MRF

```
# Image segmentation using MRF model
from PIL import Image
import numpy
from pylab import *
from scipy.cluster.vq import *
from scipy.signal import *
import cv2
import scipy

def main():
 # Read in image
 im=Image.open('7.png')
 im=numpy.array(im)

 # If grayscale add one dimension for easy processing
 if (im.ndim==2):
   im=im[:,:,newaxis]

 # Initial kmean segmentation
 nlevels=4
 lev=getinitkmean(im, nlevels)

 # MRF ICM
 win_dim=256
 while (win_dim>7):
   print(win_dim)
   locav=local_average(im,lev,nlevels,win_dim)
   lev=MRF(im, lev, locav,nlevels)
   win_dim=win_dim//2
 title('Level')
 imshow(lev*20)

 # Get the color average based on locav
 out=ACAreconstruction(lev,locav)
 figure()
 title('Seg Image')
 imshow(out)
 show()

def ACAreconstruction(lev,locav):
 out=0
 for i in range(locav.shape[3]):
   mask=(lev==i)
   out+=mask[:,:,newaxis]*locav[:,:,:,i]
 return out

def getinitkmean(im, nlevels):
 obs=reshape(im,(im.shape[0]*im.shape[1],-1))
 obs=whiten(obs)

 (centroids, lev)=kmeans2(obs,nlevels)
 lev=lev.reshape(im.shape[0],im.shape[1])
 return lev

def delta(a,b):
 if (a==b):
   return -1
 else:
   return 1

def MRF(obs, lev, locav, nlevels):
 (M,N)=obs.shape[0:2]
 for i in range(M):
   for j in range(N):
    # Find segmentation level which has min energy (highest posterior)
    cost=[energy(k,i,j, obs, lev, locav) for k in range(nlevels)]
    lev[i,j]=cost.index(min(cost))
 return lev

def energy(pix_lev,i, j, obs,lev,locav):
 beta=0.5
 std=7
 cl=clique(pix_lev,i,j,lev)
 closeness=numpy.linalg.norm(locav[i,j,:,pix_lev]-obs[i,j,:])
 return beta*cl+closeness/std**2

def local_average(obs, lev, nlevels, win_dim):
 # Use correlation to perform averaging
 mask=numpy.ones((win_dim,win_dim))/win_dim**2

 # 4d array (512, 512, ncolors, nlevels)
 locav=ones((obs.shape+(nlevels,)))

 for i in range(obs.shape[2]): # loop through image channels
   for j in range(nlevels): # loop through segmentation levels
    temp=(obs[:,:,i]*(lev==j))
    locav[:,:,i,j]=fftconvolve(temp,mask,mode='same')
 return locav

def clique(pix_lev, i, j, lev):
 (M,N)=lev.shape[0:2]

 #find correct neighbors
 if (i==0 and j==0):
   neighbor=[(0,1), (1,0)]
 elif i==0 and j==N-1:
   neighbor=[(0,N-2), (1,N-1)]
 elif i==M-1 and j==0:
   neighbor=[(M-1,1), (M-2,0)]
 elif i==M-1 and j==N-1:
   neighbor=[(M-1,N-2), (M-2,N-1)]
 elif i==0:
   neighbor=[(0,j-1), (0,j+1), (1,j)]
 elif i==M-1:
   neighbor=[(M-1,j-1), (M-1,j+1), (M-2,j)]
 elif j==0:
   neighbor=[(i-1,0), (i+1,0), (i,1)]
 elif j==N-1:
       neighbor=[(i-1,N-1), (i+1,N-1), (i,N-2)]
    else:
       neighbor=[(i-1,j), (i+1,j), (i,j-1), (i,j+1),\
           (i-1,j-1), (i-1,j+1), (i+1,j-1), (i+1,j+1)]

    return sum(delta(pix_lev,lev[i]) for i in neighbor)

if __name__=="__main__":
    main()
```