CS 736 - Medical Image Computing Q1

Report on Assignment 2 (ImageSegmentation)

Team: Dharshan - 18D180009 and Senthil - 18D180024

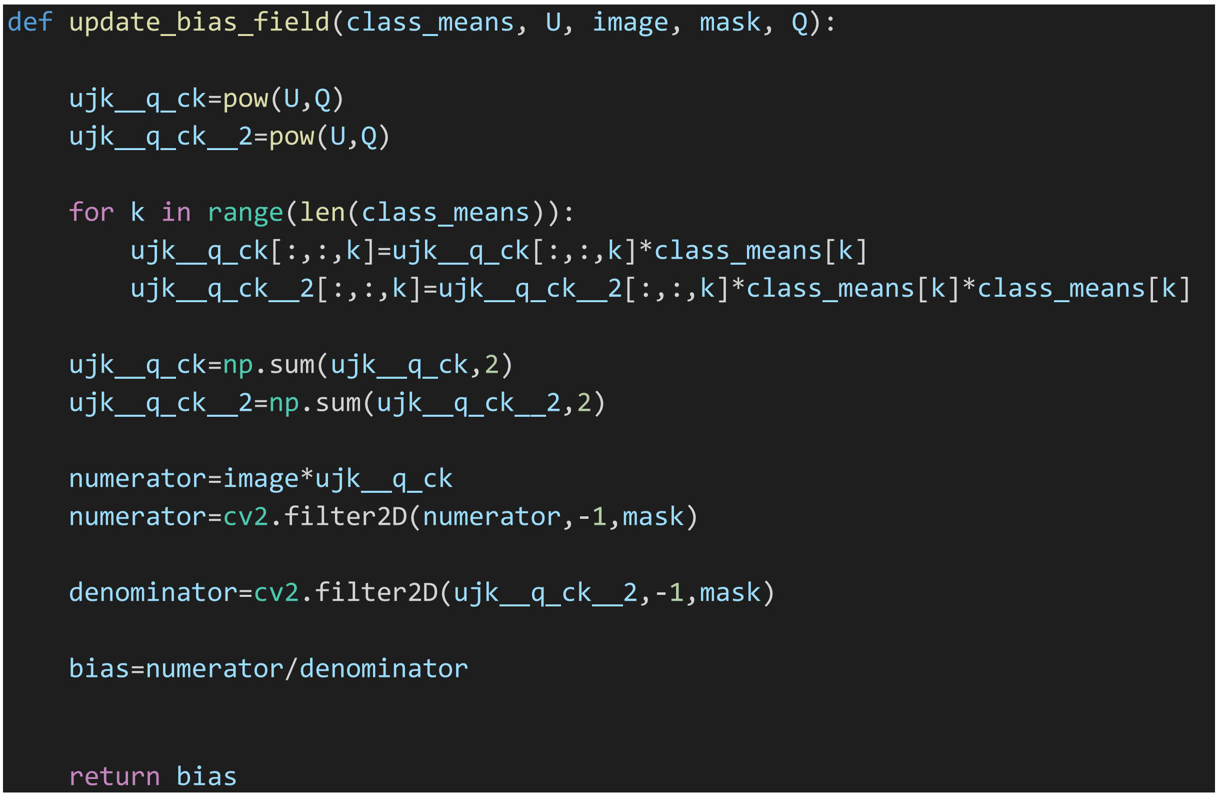
Class means updater:

```
def update_class_means(K, U, image, mask, bias, Q):
    class_means=np.zeros(K)

    wij_bi=cv2.filter2D(bias,-1,mask)
    wij_bi2=cv2.filter2D(bias*bias,-1,mask)

    for k in range(K):
        numerator=(pow(U[:,:,k],Q)*wij_bi)*(image)
        temp=pow(U[:,:,k],Q)

        denominator=pow(U[:,:,k],Q)*wij_bi2

        class_means[k]=np.mean(numerator)/np.mean(denominator)

    return class_means
```

class membership updater:

```
def update_membership(class_means, image, mask, bias, Q):
    I,J=image.shape
    U=np.zeros([I,J,len(class_means)])

    wij_bi=cv2.filter2D(bias,-1,mask)
    wij_bi2=cv2.filter2D(bias*bias,-1,mask)

    for k in range(len(class_means)):
        djk=(class_means[k]*class_means[k]*wij_bi2)+ image*image -
2*class_means[k]*(wij_bi*image)

        djk=djk+(djk==0)*np.mean(djk)
        U[:,:,k]=pow(djk,(1/(1-Q)))/100
        print(djk[177][127])

    print(np.argwhere(np.isnan(U)))
    # print("U",np.any(U==0))
    for k in range(len(class_means)):
        U[:,:,k]=U[:,:,k]/np.sum(U,2)



    return U
```

Bias field updater:

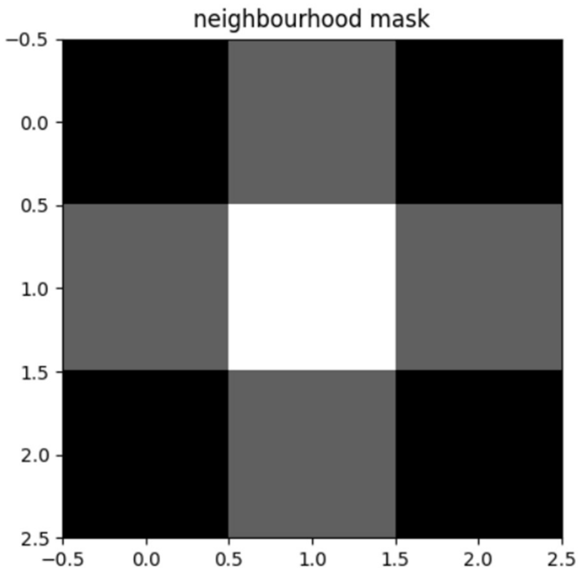


```
def update_bias_field(class_means, U, image, mask, Q):

    ujk__q_ck=pow(U,Q)
    ujk__q_ck__2=pow(U,Q)

    for k in range(len(class_means)):
        ujk__q_ck[:,:,k]=ujk__q_ck[:,:,k]*class_means[k]
        ujk__q_ck__2[:,:,k]=ujk__q_ck__2[:,:,k]*class_means[k]*class_means[k]

    ujk__q_ck=np.sum(ujk__q_ck,2)
    ujk__q_ck__2=np.sum(ujk__q_ck__2,2)

    numerator=image*ujk__q_ck
    numerator=cv2.filter2D(numerator,-1,mask)

    denominator=cv2.filter2D(ujk__q_ck__2,-1,mask)

    bias=numerator/denominator


    return bias
```

chosen Q-value

Q = 1.55

Neighbourhood mask:



Initial membership estimates:

The initial estimate images are all white as each voxel is giventhe same probability to belong to either of the 3 classes so 1/3,1/3,1/3, so its not visible.

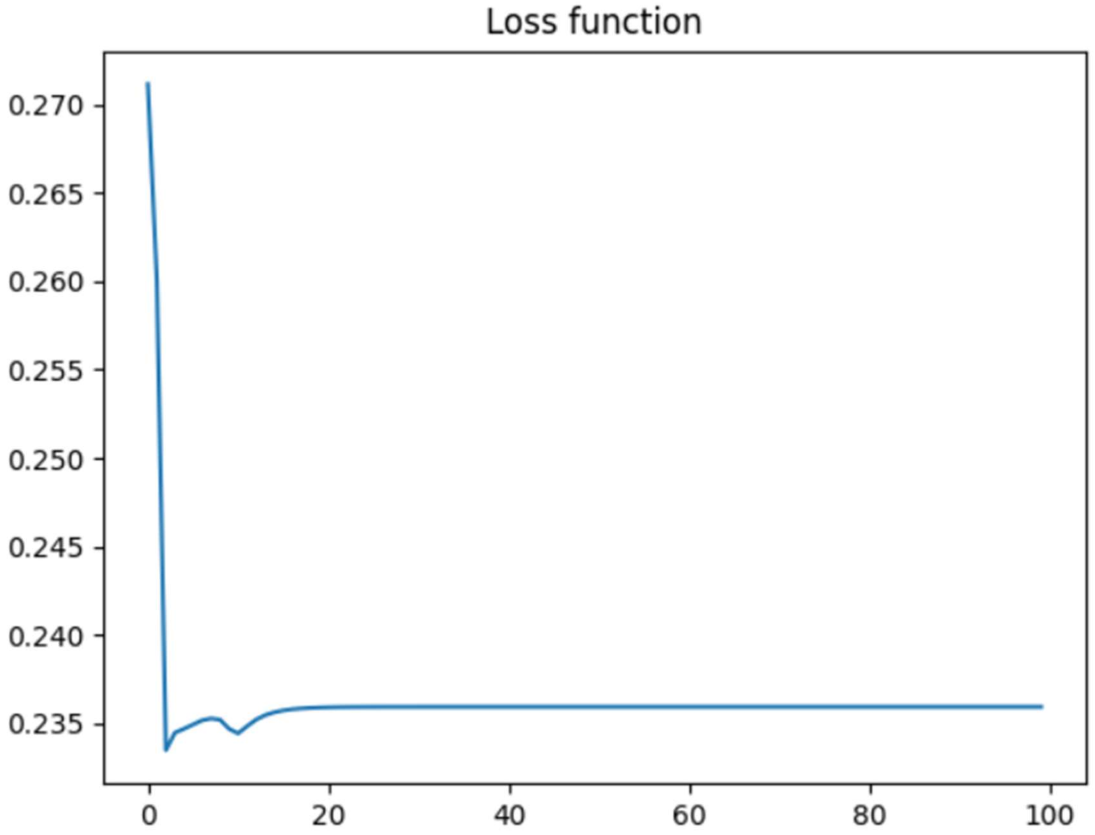
Initial class mean estimates:

C=[0.456,0.635,0.0006]

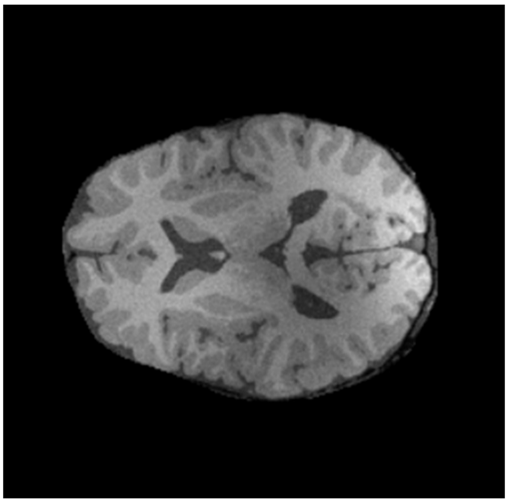
These values were chosen through performing a run of k-mean algorithm with 3 centers and using these centers as the starting point to run the FCM. This was done hoping to give the algorithm ahead start with a good and well-spaced initial estimate.

Objective function values:

```
Iteration: 1 ====> Log loss: 0.2774423959943755

Iteration: 2 ====> Log loss: 0.26005332770490386

Iteration: 3 ====> Log loss: 0.23347315072605132

Iteration: 4 ====> Log loss: 0.2344653057892684

Iteration: 5 ====> Log loss: 0.2346848001018944

Iteration: 6 ====> Log loss: 0.23492282279345456

Iteration: 7 ====> Log loss: 0.23516313609141104

Iteration: 8 ====> Log loss: 0.23526890442192233

Iteration: 9 ====> Log loss: 0.23519998333564476

Iteration: 10 ====> Log loss: 0.23468629584946366

Iteration: 11 ====> Log loss: 0.23442630043236978

Iteration: 12 ====> Log loss: 0.23483319145674855

Iteration: 13 ====> Log loss: 0.23520598352940691

Iteration: 14 ====> Log loss: 0.23545662605259804

Iteration: 15 ====> Log loss: 0.23561845107793397

Iteration: 16 ====> Log loss: 0.23572344178162782

Iteration: 17 ====> Log loss: 0.23579225364718018

Iteration: 18 ====> Log loss: 0.23583770362580567

Iteration: 19 ====> Log loss: 0.2358678749200642

Iteration: 20 ====> Log loss: 0.23588796590589653
```

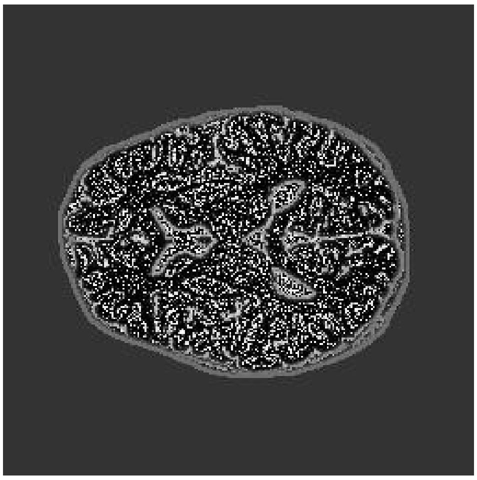


Corrupted image:

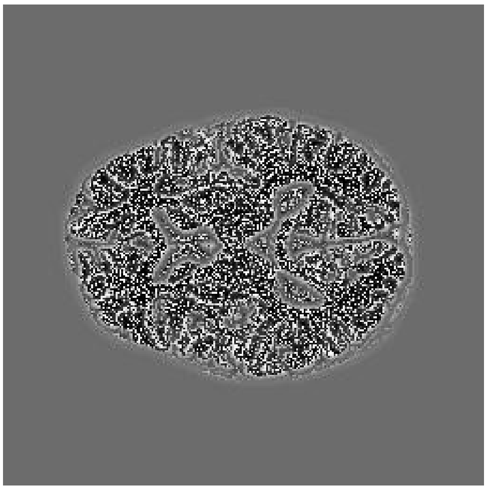


Optimal class membership-estimate image

Class1:

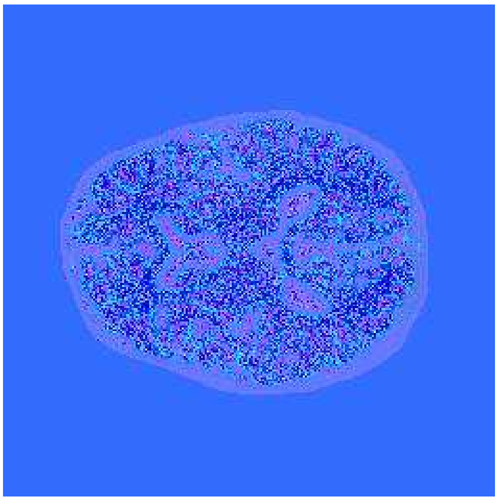


Class 2:

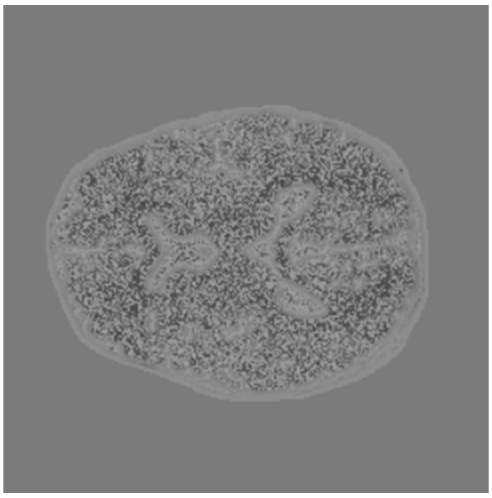


Class 3:

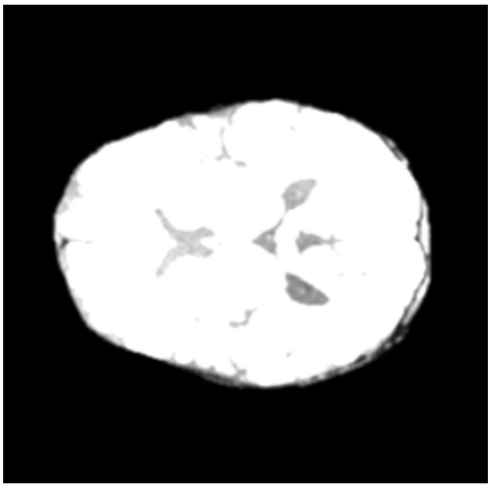
Heatmap:



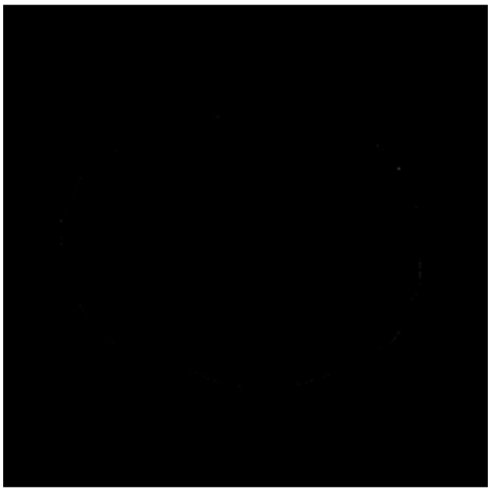
Optimal bias field estimate



Bias removed image



Residual image



Final estimates of class means:

C=[0.27024234, 0.31108601, 0.29653829]

The algorithm taught in class performed well, except for over generalizing the bias. That being the case, the algorithm did give a unique soln.