Notes for Tyler

The first function that is called is TexturePatches

% this function is the helper function which calls the function which makes

% the masks. The masks are then used to in the patching program which

% generates the patches to be classified, either by hand or automatically

% so this is, in most cases, the first function which needs to be run

%

% the bulk of the function simply is a case-by-case list of the lsms and

% hard coded options which say which channel of the lsm will be used to

% decide on the mask and which will be shown when patching

%

% gets inputs which are

% opt: which sets an option; it's not really used but could be

% lsms has the list of lsm files and inds is which of those to process

%

% in the GUI, I imagine that all these options will be set by the user so

% this is somewhat defunct but I include it for reference

function TexturePatches(opt,lsms,inds)

TexturePatches calls two useful files using vecadCh and vessCh, and a filename fn

% this bit calls the function to get the masks

TextureSliceMasks(fn,vessCh,vecadCh)

% this goes through the lsm to find minimum and maximum values across

% the lsm which are used when displaying things

GetDataStdThresh(fn,vecadCh);

I have re-written these functions so they work with tiffs and not lsms so you’ll be able to see the difference in the functions in the file GetSliceMasksIms:

% this function takes a list of filenames fnlist and then generates the masks

% it first asks the user to input the mask channel and the classification

% channel, which it retyrns as outputs

%

% In the original functions it replaces the functions TexturePatches and TextureSliceMasks

function[maskchan,claschan]=GetSliceMasksIms(fnlist)

I’ve written a couple of helper functions which are meant to be first attempts at having something that will work if we pass in an lsm or an image file but I haven’t gone further than the idea as I’m not really sure it’s that easy (you can look at the help text I’ve written for the issues) but I’m sure you’ll come up with something clever ;)

% this function could be generalised to read either an lsm or a tiff or

% whatever based on the file extension

%

% the idea is that it reads an image from the val'th member of fnlist

%

% it then returns the image in im, the number of channels in nChan and

% some flag filetype which could be used later.

%

% However, I'm still not entirely sure we can do something general because

% the lsms contain multiple slices, while the images only have one

function[im,nChan,filetype]=SliceReadIm(fnlist,val)