Macros used for quantifying S1P reporter in sections

```
macro "Reporter Regions"
{
/* Instructions
 * Draw ROI's and save as .zip file
 * set thresholds for each channel in the variables at the top of the
macro
 * open the .tif image to be analyzed
 * follow on screen instructions
   to select the folders to pull the ROI.zip files and where to save
the files
Macro functions
-Changes the background pixels in RFP and GFP channels to a value of
zero
-Divides GFP image by RFP image and creates a resulting image where
each pixel’s values are the ratio of GFP to RFP for that pixel.
-Generates a membrane mask based on the CD169 and F480 stains.
-Opens the pre-saved ROI files for white pulp, red pulp, and marginal
zones
-Gives the average GFP:RFP ratio in each ROI
*/
var thresholdGFP = 0;
var thresholdRFP = 25;
var thresholdCD169 = 20;
var thresholdF480 = 32;
//also try RFP at 45 for a stricter gate
//10.02 is the maximum ratio if the RFP threshold is 25.

                 //get image attributes
                 name = getInfo("image.filename");
                 original = getImageID();
                 originalTitle = getTitle();

                 //Set the directory where files will be pulled from
and saved to
                 OpenDir=getDirectory("Choose the folder where data
will be pulled from");
                 DataDir=getDirectory("Choose the folder where data
will be saved");
                 //convert to 32 bit
                 run("32-bit");

                 // split channels and threshold the GFP and RFP
images.
                 //If GFP=0, the ratio image will be 0, and will
become NaN during background-->NaN
                 //If RFP=0, the ratio will be x/0, which should
return NaN, bc I set Misc options
```

```
                 run("Duplicate...", "title=GFPimage duplicate
channels=1");
                 GFPImage = getImageID();
                 GFPImageTitle = getTitle();
                 changeValues(0, thresholdGFP, 0)
                 selectImage(original);
                 run("Duplicate...", "title=RFPimage duplicate
channels=2");
                 RFPImage = getImageID();
                 RFPImageTitle = getTitle();
                 changeValues(0, thresholdRFP, 0)

                 // divide image1 by image2
                 run("Misc...", "divide=NaN");
                 imageCalculator("Divide create 32-bit", GFPImage,
RFPImage);
                 rename("RatioImage");
                 RatioImage = getImageID();
                 RatioImageTitle = getTitle();

                 // generation of membrane mask
                 //Mask the CD169
                 selectImage(original);
                 run("Duplicate...", "title=CD169 duplicate
channels=3");
                 setThreshold(thresholdCD169, 255);
                 run("Convert to Mask", " black");
                 CD169Image = getImageID();
                 CD169ImageTitle = getTitle();
                 //Mask the F480 (but delete MadCam1 staining)
                 selectImage(original);
                 run("Duplicate...", "title=F480 duplicate
channels=4");
                 F480Image = getImageID();
                 F480ImageTitle = getTitle();
                 //ROIs to delete needs to be a separate file
                 roiManager("Reset");
                 open(OpenDir + "RoiSetDUMP.zip");
                 WProiCount=roiManager("count");
                 //selectImage(F480Image);
                 for (i=0; i<WProiCount; i++) {
                 roiManager("Select",i);
                 run("Set...", "value=0");}
                 setThreshold(thresholdF480, 255);
                 run("Convert to Mask", " black");
                 //Create masked image using CD169 and the F480 image
with MadCam1 stain deleted
                 imageCalculator("OR create 32-bit", CD169Image,
F480Image);
                 rename("MemMaskImage")
```

```
                //to make the mask values either 1 or 0.
                run("Divide...", "value=255.000000000");
                MemMaskImage = getImageID();
                MemMaskTitle = getTitle();

                // calculate RFP to GFP ratio on membrane.
                imageCalculator("Multiply create 32-bit", RatioImage, 
MemMaskImage);
                setThreshold(0.0001, 255.0000);
                run("NaN Background");
                rename(originalTitle + "MemRatio");
                MemRatioImage = getImageID();
                MemRatioTitle = getTitle();
                run("Fire");
                saveAs("Tiff", DataDir+MemRatioTitle+".tif");
                setMinAndMax(0.0000, 4.000);
                selectImage(RFPImage); close();
                selectImage(GFPImage);  close();
        //Ratio measurements
        run("Set Measurements...", "area mean display redirect=None 
decimal=3");
        roiManager("Reset");
        open(OpenDir + "RoiSetWP.zip");
        ROICountWP=roiManager("Count");//=3
        for (i=0; i<ROICountWP; i++) {
                roiManager("Select",i);
                roiManager("Rename", "WP");
                }
                open(OpenDir + "RoiSetRP.zip");
                ROICountRP=roiManager("Count");
        for (i=ROICountWP; i<ROICountRP; i++) {
                roiManager("Select",i);
                roiManager("Rename", "RP");
                }
                open(OpenDir + "RoiSetMZ.zip");
                ROICountMZ=roiManager("Count")
        for (i=ROICountRP; i<ROICountMZ; i++) {
                roiManager("Select",i);
                roiManager("Rename", "MZ");
                }
                roiManager("Deselect");
                roiManager("Measure");
                //Adding the settings to the results table:
                i = nResults;    // variable for counting, 
initialising with 0
                setResult("Image name", i, originalTitle);
                setResult("thresholdGFP", i, 
thresholdGFP);            // add a "Label" column to the results table 
and name the entry "point1"
                setResult("thresholdRFP", i, thresholdRFP);
```

```
                 setResult("thresholdCD169", i, thresholdCD169);
                 setResult("thresholdF480", i, thresholdF480);
                 selectWindow("Results");
                 saveAs("Results", DataDir + originalTitle + ".xls");
                 }
```

```
macro "GFP to RFP density"
{
/* Instructions
 * Draw ROI's and save as .zip file
 * set GFP and RFP thresholds by negative image as thresholdGFP and
thresholdRFP variables
 * open the .tif image to be analyzed
 * follow on screen instructions
   to select the folders to pull the ROI.zip files and where to save
the files
Macro functions
-Take each MZ ROI and each RP ROI and
-Calculate the density of GFP within that entire ROI (after setting a
threshold to eliminate background GFP pixels)
-Calculate the density of RFP within that entire ROI (after setting an
RFP threshold)
-Divide the GFP density by the RFP density to get a reporting density
ratio for that ROI.
*/
var thresholdGFP = 24;
var thresholdRFP = 33;

                 //get image attributes
                 name = getInfo("image.filename");
                 original = getImageID();
                 originalTitle = getTitle();
                 //Set the directory where files will be pulled from
and saved to
                 OpenDir=getDirectory("Choose the folder where data
will be pulled from");
                 DataDir=getDirectory("Choose the folder where data
will be saved");
                 //convert to 32 bit
                 run("32-bit");
                 // split channels and threshold the GFP and RFP
images.
                 run("Duplicate...", "title=GFPimage duplicate
channels=1");
                 GFPImage = getImageID();
                 GFPImageTitle = getTitle();
                 changeValues(0, thresholdGFP, 0)
                 selectImage(original);
```

```
                run("Duplicate...", "title=RFPimage duplicate
channels=2");
                RFPImage = getImageID();
                RFPImageTitle = getTitle();
                changeValues(0, thresholdRFP, 0)
                //Measure the area and intensities of the image
                selectImage(GFPImage);
                run("Set Measurements...", "area mean integrated
display redirect=None decimal=2");
        roiManager("Reset");
        open(OpenDir + "RoiSetWP.zip");
        ROICountWP=roiManager("Count");//=3
        for (i=0; i<ROICountWP; i++) {
                roiManager("Select",i);
                roiManager("Rename", "WP");
                }
                open(OpenDir + "RoiSetRP.zip");
                ROICountRP=roiManager("Count");
        for (i=ROICountWP; i<ROICountRP; i++) {
                roiManager("Select",i);
                roiManager("Rename", "RP");
                }

                open(OpenDir + "RoiSetMZ.zip");
                ROICountMZ=roiManager("Count")
        for (i=ROICountRP; i<ROICountMZ; i++) {
                roiManager("Select",i);
                roiManager("Rename", "MZ");
                }
                roiManager("Deselect");
                roiManager("Measure");

        //RFP measurements
                selectImage(RFPImage);
        roiManager("Reset");
        open(OpenDir + "RoiSetWP.zip");
        ROICountWP=roiManager("Count");//=3
        for (i=0; i<ROICountWP; i++) {
                roiManager("Select",i);
                roiManager("Rename", "WP");
                }
                open(OpenDir + "RoiSetRP.zip");
                ROICountRP=roiManager("Count");
        for (i=ROICountWP; i<ROICountRP; i++) {
                roiManager("Select",i);
                roiManager("Rename", "RP");
                }
                open(OpenDir + "RoiSetMZ.zip");
                ROICountMZ=roiManager("Count")
        for (i=ROICountRP; i<ROICountMZ; i++) {
```

```
                 roiManager("Select",i);
                 roiManager("Rename", "MZ");
                 }
                 roiManager("Deselect");
                 roiManager("Measure");

                 //Adding the settings to the results table:
                 i = nResults;    // variable for counting,
initialising with 0
                 setResult("Image name", i, originalTitle);
                 setResult("thresholdGFP", i, thresholdGFP);
                 setResult("thresholdRFP", i, thresholdRFP);
                 // save the results table as an excel file
                 selectWindow("Results");
                 saveAs("Results", DataDir + originalTitle +"GFP RFP
dens" + ".xls");
                 selectWindow("Results"); close();
                 }
```