Workflow for Fibrosis Analysis in PSR-Stained Heart Images

This protocol is designed for images captured on the Cytation 5 imager ("Indigo") using Color Brightfield at 4x, and QuPath version 0.4

Citations:

- Fiji
- QuPath
- LiverQuant
- (If used for automation) CellProfiler
- The CSMSR

Automated image preparation in CellProfiler and Fiji

For large datasets, you can avoid the manual image preparation steps by using CellProfiler to quickly merge files by name, and Fiji to rename and set the scale.

- Collect all Cytation Experiment data folders in a single folder.
- Open the **merge_colors.cpproj** project in CellProfiler.
    - Click on the **Images** step, and drag the folder containing the Cytation experiments onto the file list pane. Click **Apply filters to the file list**.
    - Go to the **Metadata** step and click **Update**, and verify that the ROI and well information show up in the table below.
    - Go to the **Names and Types** step and click **Update**, and verify that the red, green, and blue files show up in the table below.
    - Set the output folder for the spreadsheet.
    - Click **Analyze Images**. ***Note*:** The output images will be saved in the original folders, in order to preserve the original file hierarchy. This is essential for the next step, the Fiji macro.
- Run the Fiji macro to rename and scale the images. ***Note*:** The final output images will be saved in the output folder specified when running the macro.
- Process the scaled, renamed output images in QuPath.

Manual image preparation in Fiji

1. Generate an RGB TIFF for each stitched ROI image.
    1.1. Open Red, Green, and Blue stitched images in Fiji.
    1.2. *Image > Color > Merge channels.*
    1.3. *Image > Adjust > Brightness & Contrast > Reset* for each channel.

1.4. a. If you have many images, automate the scaling and RGB conversion as follows:

1.4.1. Save composite image: *File > Save as > TIFF.* Include ROI number, sample, and the xpt date/timestamp in the name.

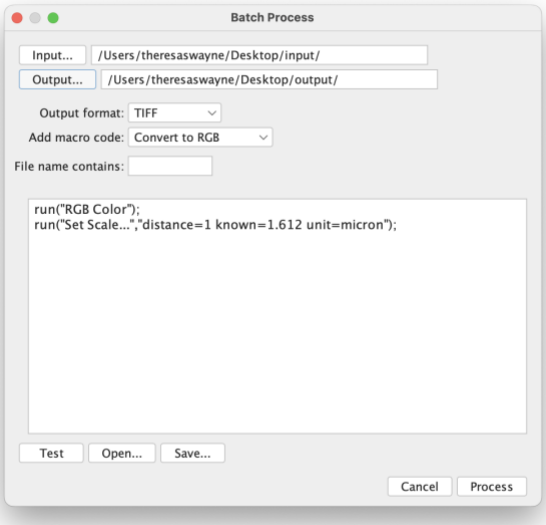
1.4.2. Place all the composite images into a folder.

1.4.3. Create a new folder in a different location for the output.

1.4.4. *Process > Batch > Macro...* and paste in the following text:

```
run("Set Scale...","distance=1 known=1.612 unit=micron");
run("RGB Color");
```

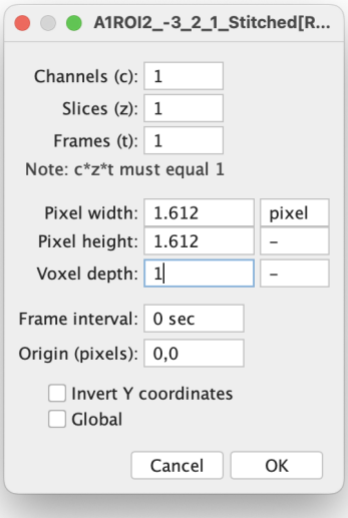
1.4.5. Click **Process.**



1.4. b. If you have just a few images, set scale and convert to RGB manually as follows:

1.4.1. *Image > Type > RGB Color*

1.4.2. On the RGB image, *Image > Properties,* then type in **1.612** for the Pixel Width and Pixel Height.



.

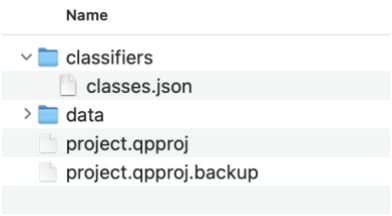
1.5. *File > Save as...* TIFF including ROI number, sample, xpt date/timestamp, and RGB in the name.

Project preparation in QuPath

Use the template scripts and classifiers, included below.

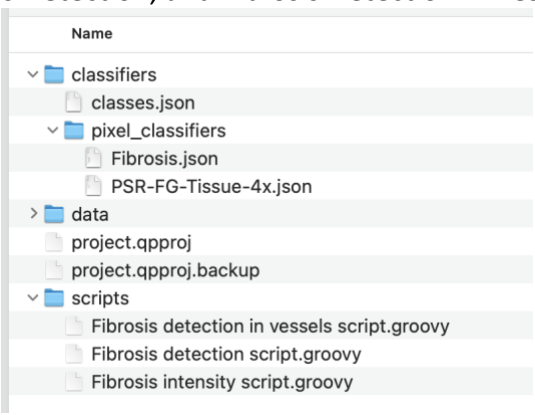
1. Create an empty folder to contain your project.
2. In QuPath, *File > Project > Create Project >* select your empty folder and click Open.
3. Drag your RGB images into the Image List at left. (You can have any number of images in a project, but it is advisable to break them into groups that your computer can easily handle.) Set options as shown below. Click **Import.**

4. *File > Save* the project.
5. Add and update the template scripts and classifiers:
    5.1. Find the project folder on your computer. After you save the project, it should contain some default classes as shown below.



5.2. Inside the **classifiers** folder, create a folder called **pixel_classifiers** and add the Fibrosis.json and PSR-FG-tissue-4x.json files to it. (Do not add FibrosisQuant; it will be created later.)

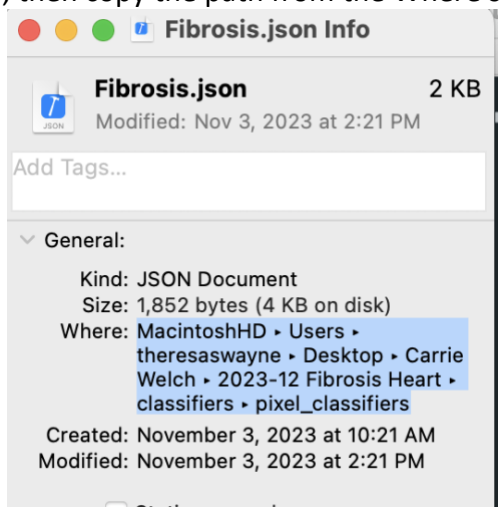
5.3. Inside the project folder, create a folder called **scripts** and add the Fibrosis Intensity, Fibrosis Detection, and Fibrosis Detection in Vessels scripts to it.



5.4. Update the file path in the Fibrosis Intensity script inside your new project folder.

5.4.1. Open the file in a text editor.

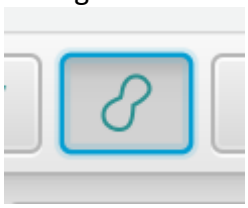
5.4.2. Replace the paths for saving Fibrosis.json and FibrosisQuant.json files to match your new pixel-classifiers folder, approximately lines 29 and 38. (To get the full path, on Mac, select one of the existing pixel classifiers, press Cmd-I (*File > Get Info*), then copy the path from the Where section as shown below.)

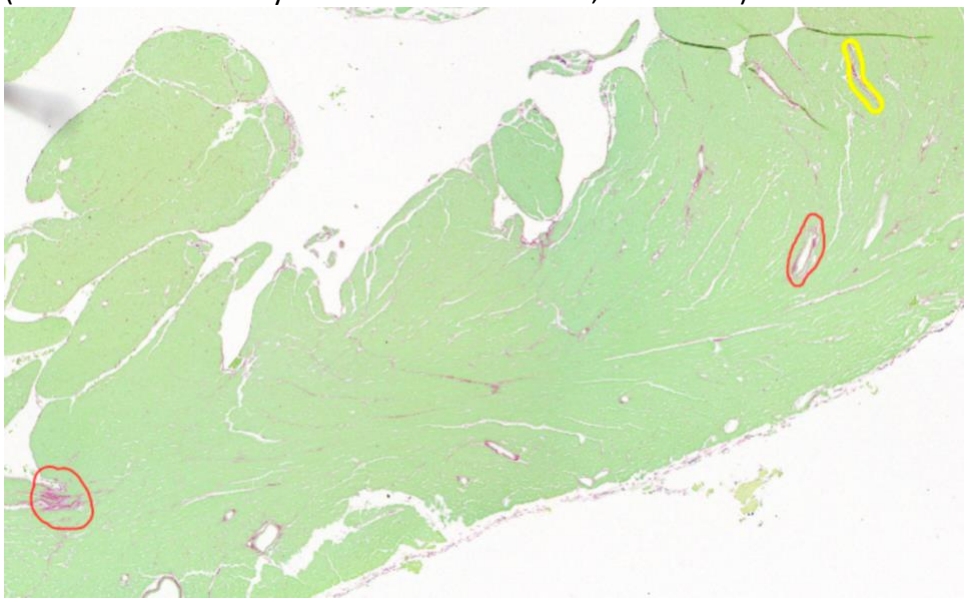


5.4.3. **Save** the updated script.

Tissue Analysis in QuPath

Measure fibrosis intensity on a representative area

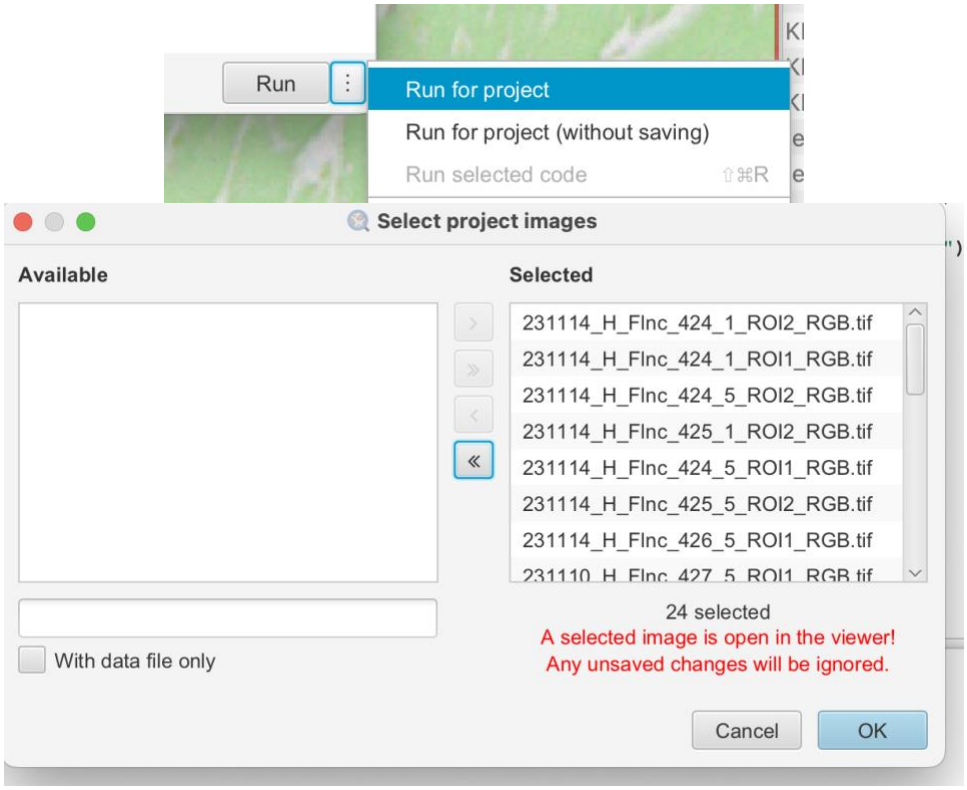
1. Determine the intensity of fibrotic regions and background.
    1.1. Pick one representative image and annotate 3 fibrotic regions with the brush (don't have to exactly match the fibrotic area; see below). 



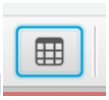
    1.2. *Automate > Scripts > Project Scripts > Fibrosis Intensity Script* > click Run. This will update the Z value in Fibrosis and FibrosisQuant.json.

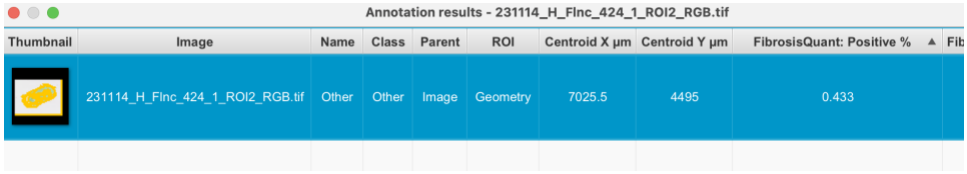
Detect tissue and % fibrotic areas for all images in the project

2. Delete any annotations remaining from the intensity determination.
3. *Automate > Scripts > Project Scripts > Fibrosis Detection Script* > click the 3 dots next to Run and select Run for Project. When prompted, add all the project images to the list on the right. Click OK.

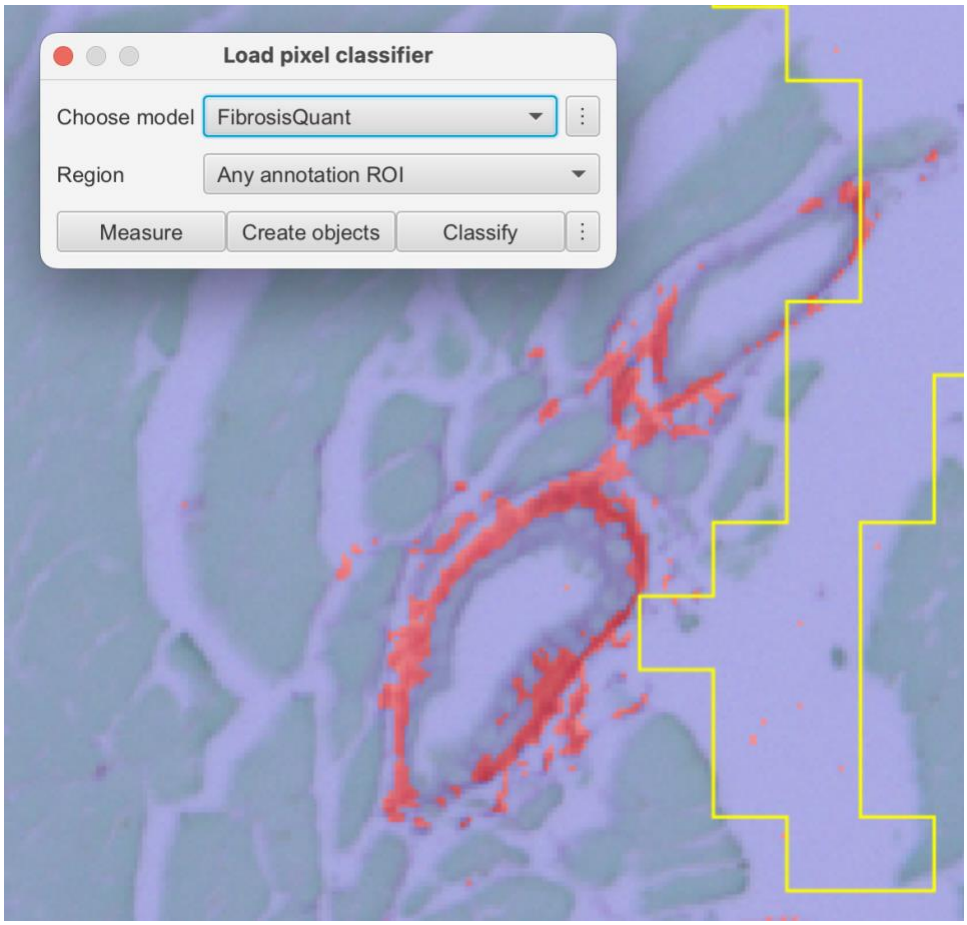


Check the results

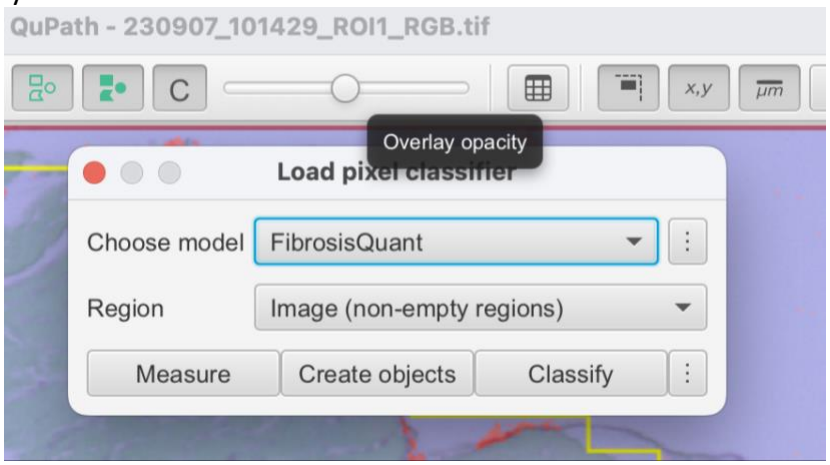
1. Select an image, click the Measurements tool  > *Show Annotation Measurements* and look at **FibrosisQuant: Positive%**



2. Select an image, Classify > Pixel classification > Load PixelClassifier > select FibrosisQuant. **Do not click any buttons on this dialog!**



3. Adjust opacity with the slider next to the **C** on the toolbar.

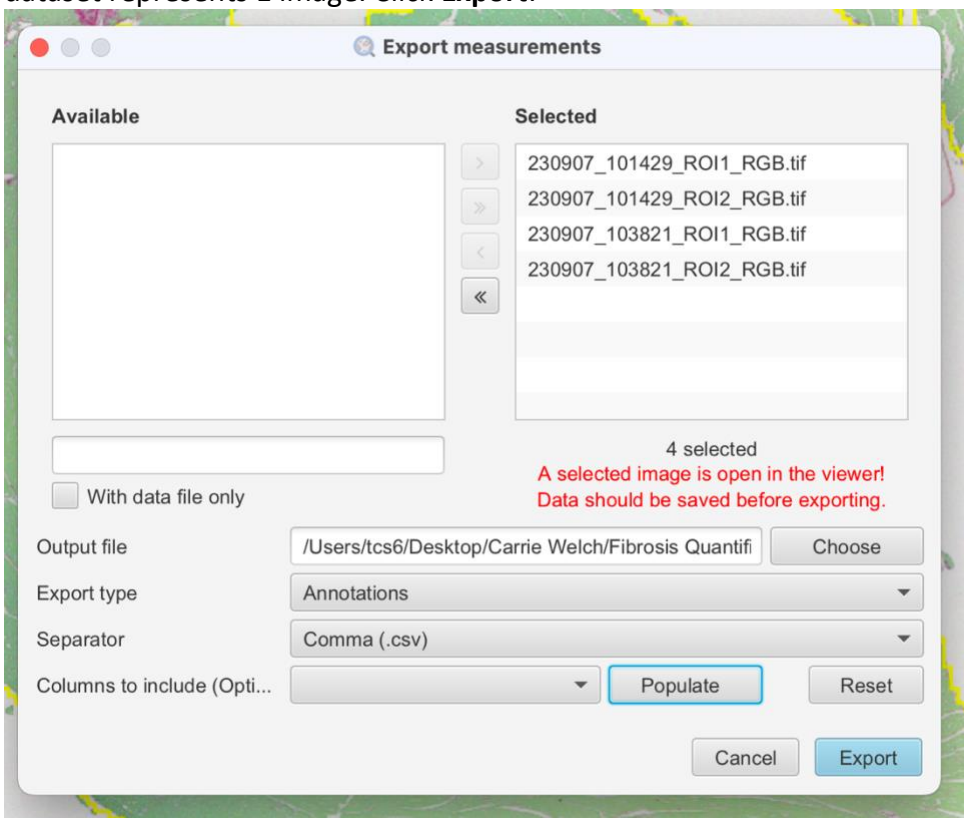


Export measurements and snapshots

1. Export the measurements.
    1.1. *Measure > Export Measurements.*
    1.2. Select all images (move them to the right hand side) to get data from all images in the project.
    1.3. Set the Export type to **Annotations** (it's important to do this first)

1.4. Set format to **.CSV**

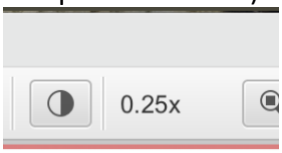
1.5. Set the output file name and folder (ideally in the project folder). Each row in the dataset represents 1 image. Click **Export**.



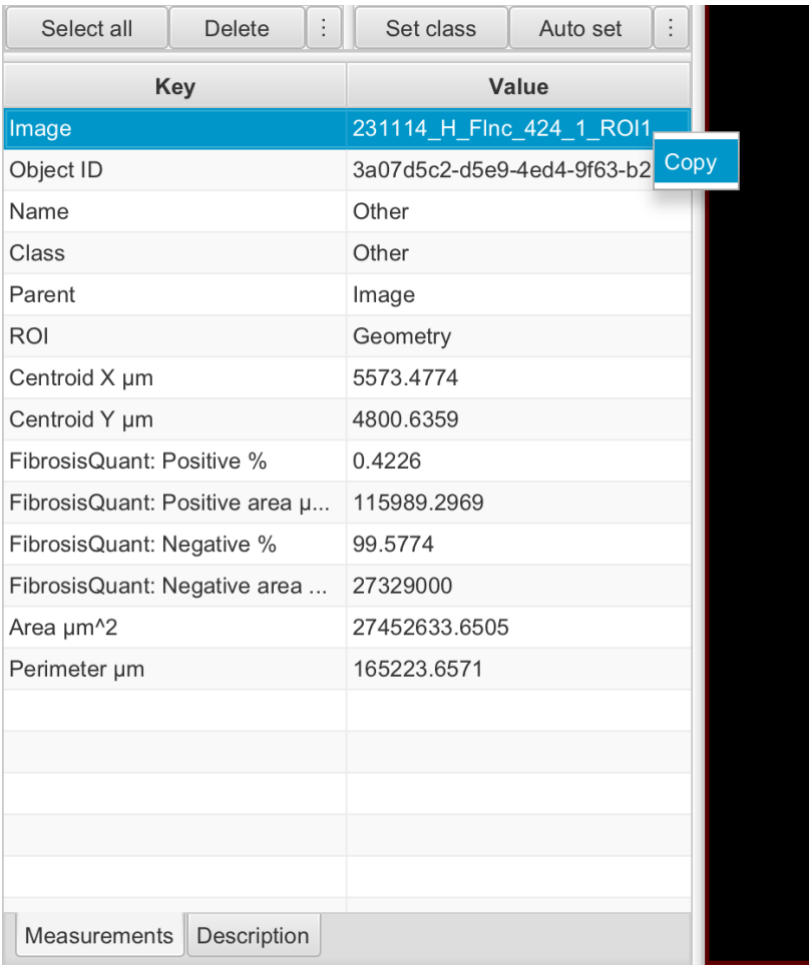
2. Export a screenshot from each image.

2.1. If you do not see the fibrosis classifications and data, follow the steps above to load the classifier, adjust opacity, and show annotation measurements.

2.2. Zoom in as much as possible while retaining the whole section in the window. To standardize the zoom level, double-click on the zoom level and type the desired downsample factor (4.0 corresponds to 0.25x)



2.3. To make it easier to type the name, right-click and copy the image name from the info at bottom left. ***Note:*** To paste the name, use right-click if Cmd-V does not work.



2.4. To save **just the image** with overlay, *File > Export snapshot > Current viewer content*

2.5. To save **the image and the sidebar** including the associated measurements, be sure the Annotations tab is selected at left, and the whole tissue annotation is selected. You should see the Fibrosis Positive % in the lower sidebar. Then *File > Export snapshot > Main window content*

Vessel Fibrosis

1. Do this after measuring total fibrosis and screenshotting, because the previous annotations may be deleted.
2. Use the brush to make annotations on vessels of approximately 50-200 microns (diameter). The minimum diameter is more significant. Include the fibrotic area but don't go beyond it if possible. Do this for all images in the project.
    2.1. Suggestion: Use keyboard shortcuts (B for brush, M for move; or with any tool active, hold down space bar while dragging to move)
    2.2. Measure diameters of objects: *Analyze > Calculate features > Add shape features*
3. Export screenshots : Ensure that the annotated vessels are showing. File > Export Images > Rendered RGB > choose JPG and downsample factor of 5.

4. Run Vessel Fibrosis script (on project if desired). The values of interest are the total area and the vessel diameters.
5. Export measurements.

Scripts

Fibrosis intensity script.groovy

```
//set image type
setImageType('BRIGHTFIELD_OTHER');
//select annotations
selectAnnotations();
//merge annotations
mergeSelectedAnnotations()

//average intensity quantification for merged annotations
runPlugin('qupath.lib.algorithms.IntensityFeaturesPlugin',
'{"pixelSizeMicrons":1.0,"region":"ROI","tileSizeMicrons":25.0,"colorOD":false,"color
Stain1":false,"colorStain2":false,"colorStain3":true,"colorRed":false,"colorGreen":fa
lse,"colorBlue":false,"colorHue":false,"colorSaturation":false,"colorBrightness":fals
e,"doMean":true,"doStdDev":false,"doMinMax":false,"doMedian":false,"doHaralick":false
,"haralickDistance":1,"haralickBins":32}')

//get key and values for annotations
def annotations = getAnnotationObjects()

//get individual key for intensity quantification
for (annotation in getAnnotationObjects()){
x = measurement(annotation, "ROI: 1.00 µm per pixel: Residual: Mean")
}

//optimization for intensity quantification
//for fibrosis with acetic acid counterstain:
//z = (x*10)
//for fibrosis with fast green counterstain:
z = -(x*0.1)

//get pre-saved or pre-generated Fibrosis file and write to it
//**IMPORTANT** You must include your own path directory below for the script to
function properly
//def file = new File('YOUR PATHNAME GOES HERE/Fibrosis.json')
//def file = new File('/Users/tcs6/Desktop/Carrie Welch/LiverQuant
practice/classifiers/pixel_classifiers/Fibrosis.json')
def file = new File('/Users/theresaswayne/Desktop/Fibrosis
2/classifiers/pixel_classifiers/Fibrosis.json')


def newConfig = file.text.replaceAll(' 0.0', z.toString())

//Write new file to existing Config
```

```
//**IMPORTANT** You must include your own path directory below for the script to
function properly
//new File('YOUR PATHNAME GOES HERE/LiverQuantF.json').write(newConfig, "utf-8")
//new File('/Users/tcs6/Desktop/Carrie Welch/LiverQuant
practice/classifiers/pixel_classifiers/LiverQuantF.json').write(newConfig, "utf-8")
new File('/Users/theresaswayne/Desktop/Fibrosis
2/classifiers/pixel_classifiers/FibrosisQuant.json').write(newConfig, "utf-8")
```

Fibrosis detections script.groovy

```
//This .Groovy script is for automated batch detection of fibrosis within tissue
sections.//
// Modified by Theresa Swayne from LiverQuant script by Dominick Hellen //
// https://github.com/DominickHellen/LiverQuant/ and DOI: 10.21769/BioProtoc.477 //

//This line of code is placed before and after analyzing each slide//
//It is used to pause the current thread for 100 milliseconds//
//It also clears accumulated RAM for batch projects//
Thread.sleep(100)
javafx.application.Platform.runLater {
getCurrentViewer().getImageRegionStore().cache.clear()
System.gc()
}
Thread.sleep(100)

//Allows Qupath to make decisions for positive staining based on staining that is
residual to DAB and hematoxylin//
setImageType('BRIGHTFIELD_OTHER');

// This should be deleted if you want to annotate by hand
//clearSelectedObjects();

// This should be deleted if you want to annotate by hand
//Increase or decrease the 'threshold' number below to improve accuracy of automated
tissue annotation
//runPlugin('qupath.imagej.detect.tissue.SimpleTissueDetection2',
'{"threshold":178,"requestedPixelSizeMicrons":200.0,"minAreaMicrons":10000.0,"maxHole
AreaMicrons":1000000.0,"darkBackground":false,"smoothImage":false,"medianCleanup":tru
e,"dilateBoundaries":false,"smoothCoordinates":true,"excludeOnBoundary":false,"single
Annotation":true}')

// detect the tissue
//createAnnotationsFromPixelClassifier("PSR-FG-tissue", 10000.0, 50.0,
"DELETE_EXISTING", "INCLUDE_IGNORED")
createAnnotationsFromPixelClassifier("PSR-FG-Tissue-4x", 10000.0, 50.0,
"DELETE_EXISTING")

//Selects annotations on each slide//
selectAnnotations();

//Quantifies positive pixel amount (Fibrotic area) for all annotations//
addPixelClassifierMeasurements("FibrosisQuant", "FibrosisQuant")

//Clears RAM and thread after each slide//
```

```
Thread.sleep(100)
javafx.application.Platform.runLater {
getCurrentViewer().getImageRegionStore().cache.clear()
System.gc()
}
Thread.sleep(100)
```

Fibrosis detection in vessels.groovy

```
//This .Groovy script is for automated batch detection of fibrosis within tissue
sections.//
// Modified by Theresa Swayne from LiverQuant script by Dominick Hellen //
// https://github.com/DominickHellen/LiverQuant/ and DOI: 10.21769/BioProtoc.477 //

//This line of code is placed before and after analyzing each slide//
//It is used to pause the current thread for 100 milliseconds//
//It also clears accumulated RAM for batch projects//
Thread.sleep(100)
javafx.application.Platform.runLater {
getCurrentViewer().getImageRegionStore().cache.clear()
System.gc()
}
Thread.sleep(100)

//Allows Qupath to make decisions for positive staining based on staining that is
residual to DAB and hematoxylin//
setImageType('BRIGHTFIELD_OTHER');

// This should be deleted if you want to annotate by hand
//clearSelectedObjects();

// This should be deleted if you want to annotate by hand
//Increase or decrease the 'threshold' number below to improve accuracy of automated
tissue annotation
//runPlugin('qupath.imagej.detect.tissue.SimpleTissueDetection2',
'{"threshold":178,"requestedPixelSizeMicrons":200.0,"minAreaMicrons":10000.0,"maxHole
AreaMicrons":1000000.0,"darkBackground":false,"smoothImage":false,"medianCleanup":tru
e,"dilateBoundaries":false,"smoothCoordinates":true,"excludeOnBoundary":false,"single
Annotation":true}')

// detect the tissue
//createAnnotationsFromPixelClassifier("PSR-FG-tissue", 10000.0, 50.0,
"DELETE_EXISTING", "INCLUDE_IGNORED")
createAnnotationsFromPixelClassifier("PSR-FG-Tissue-4x", 10000.0, 50.0,
"DELETE_EXISTING")

//Selects annotations on each slide//
selectAnnotations();

//Quantifies positive pixel amount (Fibrotic area) for all annotations//
addPixelClassifierMeasurements("FibrosisQuant", "FibrosisQuant")

//Clears RAM and thread after each slide//
Thread.sleep(100)
javafx.application.Platform.runLater {
```

```
        getCurrentViewer().getImageRegionStore().cache.clear()
        System.gc()
        }
        Thread.sleep(100)
```

Classifiers

Fibrosis.json


```
{
  "pixel_classifier_type": "OpenCVPixelClassifier",
  "metadata": {
    "inputPadding": 0,
    "inputResolution": {
      "pixelWidth": {
        "value": 1.8117583114412537,
        "unit": "µm"
      },
      "pixelHeight": {
        "value": 1.8117583114412537,
        "unit": "µm"
      },
      "zSpacing": {
        "value": 1.0,
        "unit": "z-slice"
      },
      "timeUnit": "SECONDS",
      "timepoints": []
    },
    "inputWidth": 512,
    "inputHeight": 512,
    "inputNumChannels": 3,
    "outputType": "CLASSIFICATION",
    "outputChannels": [],
    "classificationLabels": {
      "0": {
        "name": "Positive",
        "color": [
```

```
        250,
        62,
        62
      ]
    },
    "1": {
      "name": "Negative",
      "color": [
        112,
        112,
        225
      ]
    }
  }
},
"op": {
  "type": "data.op.channels",
  "colorTransforms": [
    {
      "stains": {
        "name": "H-DAB default",
        "stain1": {
          "r": 0.6511078257574492,
          "g": 0.7011930431234068,
          "b": 0.29049426072255424,
          "name": "Hematoxylin",
          "isResidual": false
        },
        "stain2": {
          "r": 0.26916687204956063,
          "g": 0.5682411743268502,
          "b": 0.777593185920953,
          "name": "DAB",
          "isResidual": false
        },
        "stain3": {
          "r": 0.6330435387995863,
```

```
        "g": -0.7128599030296365,
        "b": 0.3018056272448775,
        "name": "Residual",
        "isResidual": true
      },
      "maxRed": 255.0,
      "maxGreen": 255.0,
      "maxBlue": 255.0
    },
    "stainNumber": 3
   }
  ],
  "op": {
   "type": "op.threshold.constant",
   "thresholds": [
    0.0
   ]
  }
 }
}
```


FibrosisQuant.json


```
{
  "pixel_classifier_type": "OpenCVPixelClassifier",
  "metadata": {
    "inputPadding": 0,
    "inputResolution": {
      "pixelWidth": {
        "value": 1.8117583114412537,
        "unit": "µm"
      },
      "pixelHeight": {
        "value": 1.8117583114412537,
        "unit": "µm"
      },
      "zSpacing": {
        "value": 1.0,
        "unit": "z-slice"
      },
      "timeUnit": "SECONDS",
      "timepoints": []
    },
```

```
  "inputWidth": 512,
  "inputHeight": 512,
  "inputNumChannels": 3,
  "outputType": "CLASSIFICATION",
  "outputChannels": [],
  "classificationLabels": {
    "0": {
      "name": "Positive",
      "color": [
        250,
        62,
        62
      ]
    },
    "1": {
      "name": "Negative",
      "color": [
        112,
        112,
        225
      ]
    }
  }
},
"op": {
  "type": "data.op.channels",
  "colorTransforms": [
    {
      "stains": {
        "name": "H-DAB default",
        "stain1": {
          "r": 0.6511078257574492,
          "g": 0.7011930431234068,
          "b": 0.29049426072255424,
          "name": "Hematoxylin",
          "isResidual": false
        },
        "stain2": {
          "r": 0.26916687204956063,
          "g": 0.5682411743268502,
          "b": 0.777593185920953,
          "name": "DAB",
          "isResidual": false
        },
        "stain3": {
          "r": 0.6330435387995863,
          "g": -0.7128599030296365,
          "b": 0.3018056272448775,
          "name": "Residual",
          "isResidual": true
        },
        "maxRed": 255.0,
        "maxGreen": 255.0,
        "maxBlue": 255.0
      },
```

```
        "stainNumber": 3
      }
    ],
    "op": {
      "type": "op.threshold.constant",
      "thresholds": [
       -0.008617711179801974
      ]
    }
  }
}
```

PSR-FG-Tissue 4x.json


```
{
  "pixel_classifier_type": "OpenCVPixelClassifier",
  "metadata": {
    "inputPadding": 0,
    "inputResolution": {
      "pixelWidth": {
        "value": 25.79201640372243,
        "unit": "µm"
      },
      "pixelHeight": {
        "value": 25.79201640372243,
        "unit": "µm"
      },
      "zSpacing": {
        "value": 1.0,
        "unit": "µm"
      },
      "timeUnit": "SECONDS",
      "timepoints": []
    },
    "inputWidth": 512,
    "inputHeight": 512,
    "inputNumChannels": 3,
    "outputType": "CLASSIFICATION",
    "outputChannels": [],
    "classificationLabels": {
      "0": {
        "name": "Other",
        "color": [
          255,
          200,
          0
        ]
      },
      "1": {
        "name": "Ignore*",
        "color": [
          180,
          180,
          180
        ]
```

```
      }
    }
  },
  "op": {
    "type": "data.op.channels",
    "colorTransforms": [
      {
        "combineType": "MINIMUM"
      }
    ],
    "op": {
      "type": "op.core.sequential",
      "ops": [
        {
          "type": "op.filters.gaussian",
          "sigmaX": 1.0,
          "sigmaY": 1.0
        },
        {
          "type": "op.threshold.constant",
          "thresholds": [
            200.0
          ]
        }
      ]
    }
  }
}
```