# QuArray

***Release 0.6***

**Callum Arthurs**

**Mar 05, 2021**

# CONTENTS

**Note:** This documentation is written to support the QuArray application, developed for Tissue Array export and analysis. Download the latest stable binary and an example tissue array from a previously analysed cancer tissue array (Arthurs et al, *Sci Rep 2020*), or read the *installation page* to get started.

Welcome to the official documentation of QuArray.

CHAPTER

ONE

# INSTALLATION

## 1.1 Getting Started

QuArray is a stand-alone, platform independent application (designed for Windows, Mac or Linux operating systems), and requires only a correct binary file to be downloaded. The binary is available from GitHub.

> **Note:** An example Tissue Array is hosted here. The file contains a previously analysed cancer tissue array (Arthurs et al, *Sci Rep 2020*) and mock data file.

The application may take a short while to start when newly installed (as it is compiled with PyQt5 as a single executable file) however, this is not a problem once the GUI is initialised. The start up times vary between operating systems due to differences in how the OS checks the files. A benchmark for each OS is as follows - Ubuntu 20.04LTS 3seconds, Windows 10 42seconds, MacOS Catalina 57seconds.

## 1.2 Operating systems

### 1.2.1 MacOS

The binaries may be compatible on earlier operating systems but it has been tested on MacOS Catalina 10.15.6 and newer.

To get started, download the latest stable version of the `MacOS binary` here or by using the link at the top of this page.

The downloaded application can be moved into your applications folder or run from the downloads folder if necessary.

On the first opening of the file the user may have to certify that the program is from an unknown developer.

The application should then boot as shown in the instructional videos.

### 1.2.2 Windows

The binaries should be compatible with Windows7 and newer.

To get started, download the latest stable version of the `Windows binary` here of by using the link at the top of this page.

The downloaded application can be moved to any location on the computer.

The application should then boot as shown in the instructional videos.

### 1.2.3 Linux

The binaries may be compatible on other distros but it has been tested on Ubuntu 16.04 LTS.

It should be compatible with all versions of Ubuntu newer than this.

Download the `Ubuntu binary` and make sure to make it executable by running -

```
sudo chmod +x QuArray_ubuntu
```

Or you can do this in the Ubuntu desktop.

The program should then start as shown in the instructional videos.

CHAPTER

TWO

# INPUT FILES

## 2.1 Whole Slide Image files

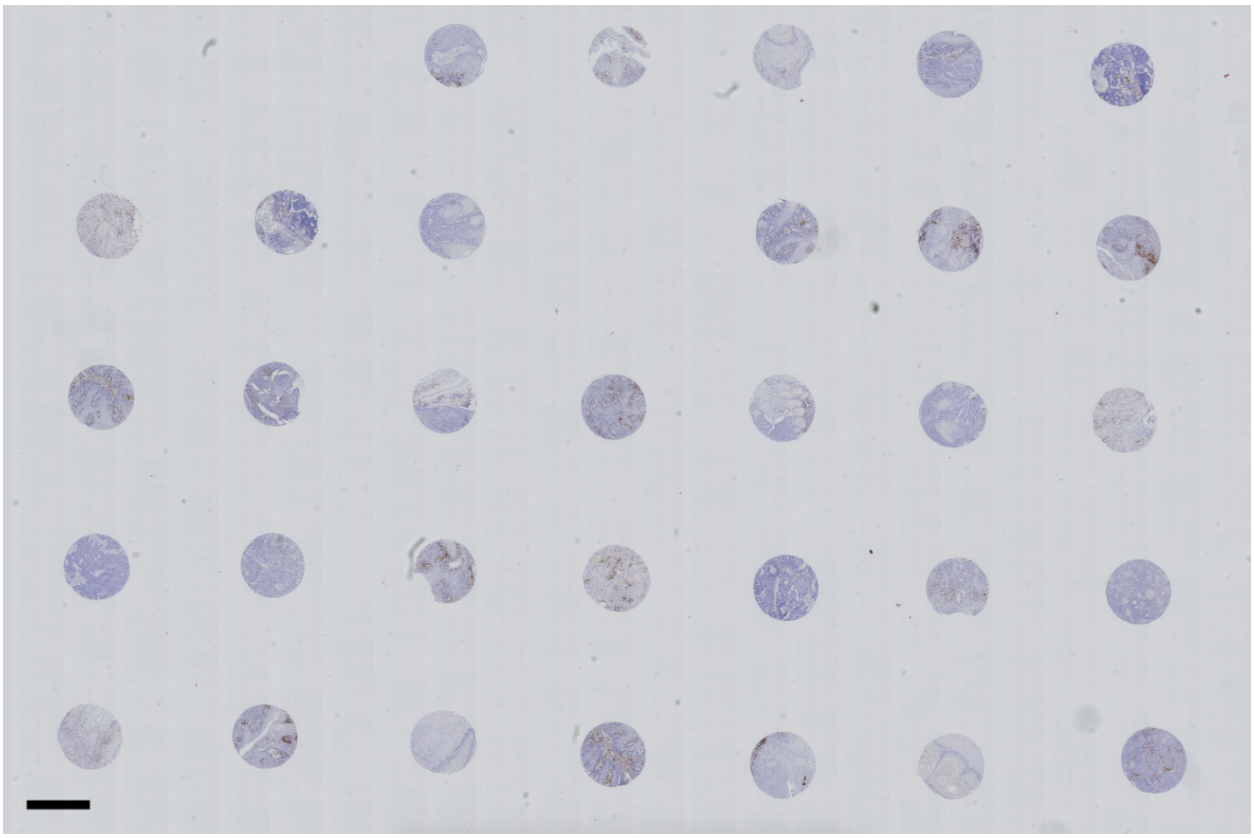
**Note:** An example Tissue Array is hosted here. The file contains a previously analyzed cancer tissue array (Arthurs et al, Sci Rep 2020) and mock data file.

### 2.1.1 Supported Formats

The program now supports whole slide images in the following vendors:

- Hamamatsu (.ndpi, .vms, .vmu)
- Aperio (.svs, .tif)
- Leica (.scn)
- Philips (.tiff)
- Sakura (.svslide)
- Generic tiled TIFF (.tif)
- MIRAX (.mrxs)
- Trestle (.tif)
- Ventana (.bif, .bif)

An example image of a WSI .ndpi file with tissue cores stained with DAB is given below:



Each core in this example is 1mm in diameter (the core diameter or number of cores is not a limitation for QuArray application). Scale bar is 1mm.

## 2.2 Tissue array map file

A spreadsheet (Excel *.xlsx) file acts as a map of the array.

The spreadsheet can also contain indexed information that is used for downstream analysis such as sample ID, sample details (e.g. date of sample collection, type) and categories (e.g. normal or diseased).

## 2.3 File locations

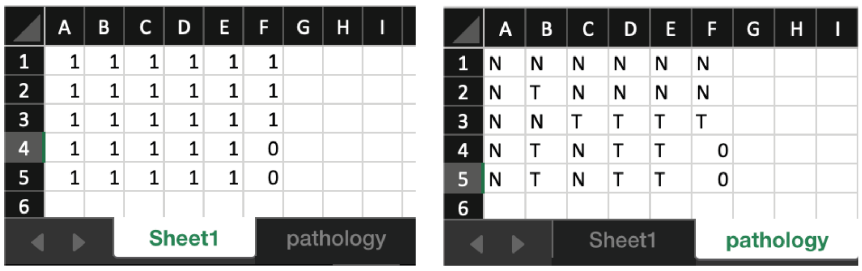
If the WSI file and the tissue array map file are saved in the same directory with the same name then the array map will be loaded automatically when the WSI file is loaded. Otherwise, the user load the map file separately.

Here is an example of a normal tree file structure containing the WSI file and the index file.

```
.
├── WSI.ndpi
└── WSI.xlsx
```

When the images are exported a new directory will be created inside the root to host output files.

## 2.4 User interface



| | A | B | C | D | E | F | G | H | I |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | | | |
| 2 | 1 | 1 | 1 | 1 | 1 | 1 | | | |
| 3 | 1 | 1 | 1 | 1 | 1 | 1 | | | |
| 4 | 1 | 1 | 1 | 1 | 1 | 0 | | | |
| 5 | 1 | 1 | 1 | 1 | 1 | 0 | | | |
| 6 | | | | | | | | | |

Sheet1 pathology

| | A | B | C | D | E | F | G | H | I |
|---|---|---|---|---|---|---|---|---|---|
| 1 | N | N | N | N | N | N | | | |
| 2 | N | T | N | N | N | N | | | |
| 3 | N | N | T | T | T | T | | | |
| 4 | N | T | N | T | T | 0 | | | |
| 5 | N | T | N | T | T | 0 | | | |
| 6 | | | | | | | | | |

Sheet1 pathology

Left - Image of the first tab.

Right - Image of the second tab.

The first tab contains 1 where a core is present and 0 where there is no core. There should be nothing in the rest of the document.

The second tab contains pathology info and is used to make a figure if needed. This is not necessary to run the program. The tab does not have to be named *Pathology* but should be in the second tab location. T and N stand for Tumour and Normal respectively.

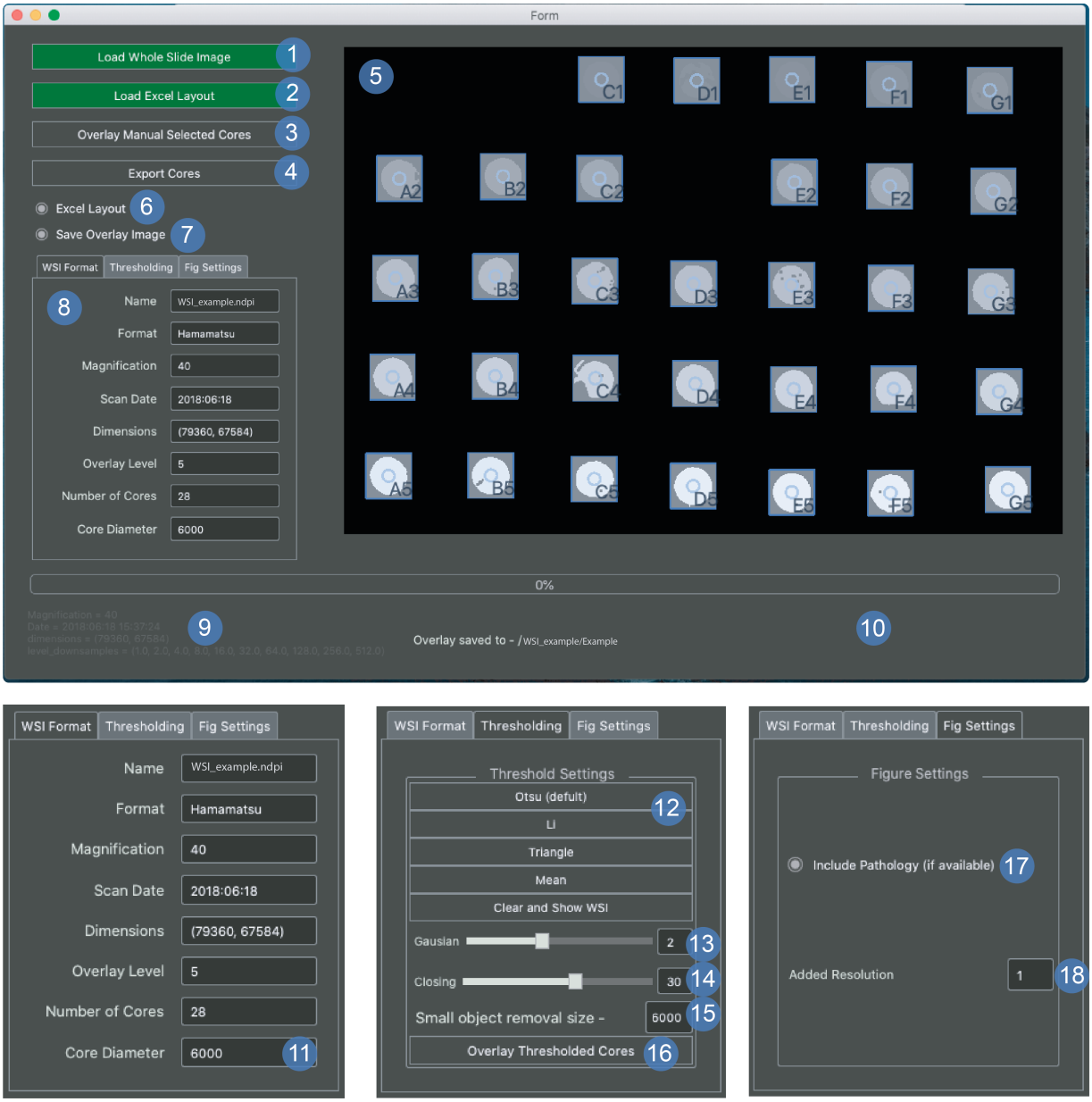
CHAPTER

THREE

# WHOLE SLIDE IMAGE EXPORT

There are videos detailing the whole slide image export process from start to finish at the *end of this page*. Below is a breakdown of each panel and button used in the examples.

## 3.1 Tissue export window




The tissue export window is for exporting png images from whole slide files using a map of the array that is provided by the user - see *array map example*.

1. Load WSI prompts a dialog box for the user to select a NDPI or SVS file.
2. Load Excel this is a prompt to load the *excel map*.
    - If the excel is named the same as the array and in the same directory,it will be loaded automatically. Otherwise this button can be used to load a different location.
    - If this has happened then the button will turn green.
    - This can also be used to reload the excel map at any point if the map has been changed by the user.

3. Overlay of manually selected cores.
    - This is applied after a user selection has been applied in panel number 5.
    - It can also be used after the cores have been moved to update the window.
4. Export cores should only be executed the user is satisfied that the cores are in the correct configuration.
    - Export core command will disable user input and export the core png images to the WSI directory.
5. Display window.
    - Double click to add core.
    - Spacebar to remove last point.
    - You can drag pre-applied cores before or after applying the bounding boxes.
6. Excel layout the recommended way to index cores is with the row names as the number and the column names as the letter (A6 col A row 6) if you want to reverse this (A6 row A col 6) then uncheck the box when the window opens.
7. *Save overlay image* determines whether or not to automatically save the image in the viewer (5).
8. *Tab viewer window* is further explained in points 11-18.
9. Image metadata for the WSI file.
10. *Progress update panel* is where progress updates will appear.
11. *Core diameter* is currently set to 6000 pixels which is similar bounding box size as a 1mm diameter tissue core.
    - This value can be chosen before image export and you can see what the new selection looks like by selecting the overlay cores button (3).
12. Thresholding options.
    - Choose the best threshold then proceed to 13.
13. Gaussian blur shows the sigma value applied to the gaussian which must be applied after 12.
14. Closing should be applied after 13 and stands for binary morphological closing.
    - higher slider values will lead to a more closed mask.
15. Removal of small objects from the mask gives the value of the minimum size to be removed.
    - If you have very small tissue cores then this will need to be reduced.
16. This applies the changes added in steps 12-15 and overlays the core images and labels.
17. These are the settings for the first figure that is exported.
    - This option will add a bounding box to the figure to denote the pathology of the core as either red or green.
    - This will only work if the first tab of the xlsx file with the array contains a map indexed with N and T for normal and tumour, respectively.
    - Please see the *array map example* for more indexing pathology details.
18. If you increase this number, then it will increase the level that the images are taken from in the WSI.
    - Increasing this will exponentially slow the program down so use sparingly.

### 3.1.1 Example tissue array export workflow 1 - Auto threshold

### 3.1.2 Example tissue array export workflow 2 - Manual selection

CHAPTER

FOUR

# DAB STAIN ANALYSIS

QuArray provides the option to analyse DAB staining in exported TA images. Below is a video detailing the workflow to be applied to images that have been exported from the *TA export window*. There is a detailed breakdown of each window and button *below*.

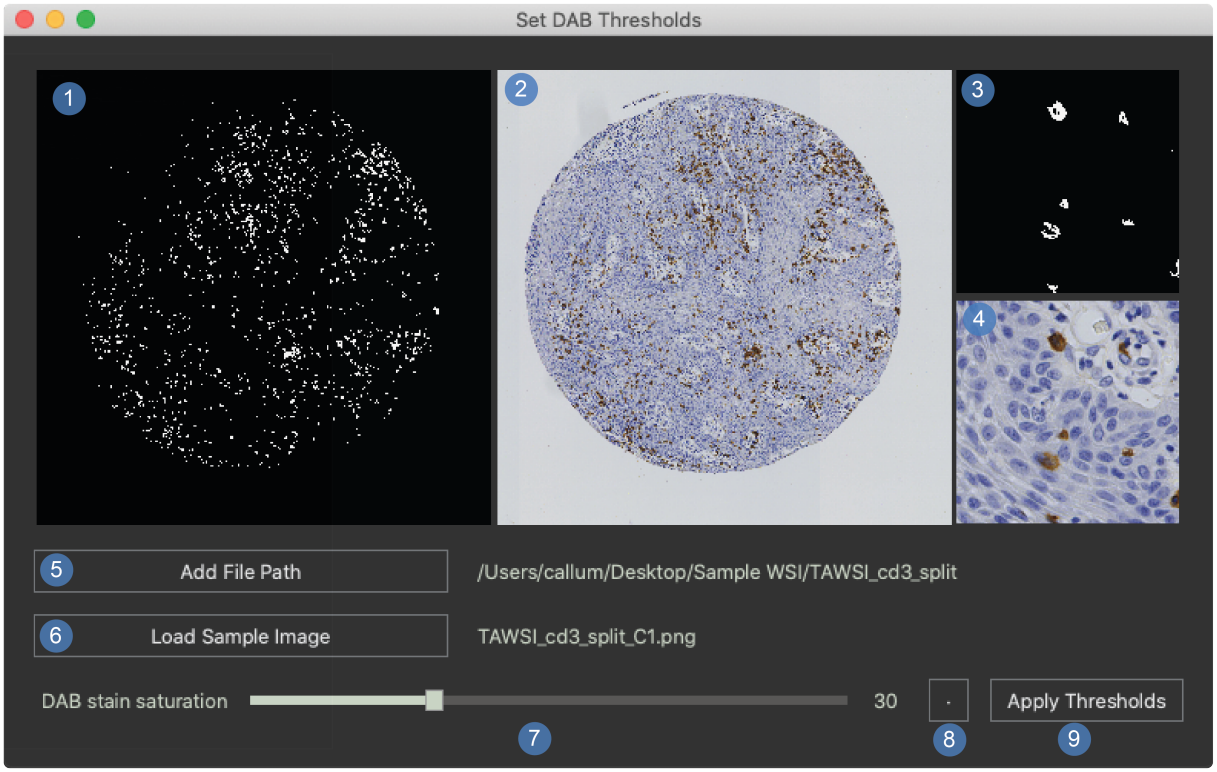
## 4.1 Input

The user selects a threshold using the *threshold selector window*. The analysis is then applied to any PNG file in a given directory.

## 4.2 Selecting a threshold window

To view the *threshold selector window* select the threshold button from the main window. The window shown below should appear.

There is also an option to select the file path of the PNG files. Please see *measurements page*.




1. Mask for the whole image.
    - White pixels will be measured as stain.
    - This is a low-resolution version of the WSI.
2. Original whole image at a low resolution.
3. Full resolution magnification mask.
4. Full resolution magnification image.
5. Add a file path.
    - This is used to select the directory containing the PNG images that will be used to optimise the threshold.
    - A random image from this file will be loaded into the window.
6. Load sample image.
    - This will load a new random image into the window.
    - It is good to optimise the thresholds on at least 10 sample images from the dataset.
7. Saturation slider.
    - This will change the minimum saturation value for the DAB stain.
    - The value will be displayed in windows 1 and 3.
8. Toggle button which shows the original image again and it is now depreciated due to the extra windows.
9. Apply thresholds.
    - This sends the new threshold to the main window of the program.
    - This should be selected before pressing the DAB analysis button in the main window.

- The thresholds button of the main window will turn green.

## 4.3 Running the analysis

This window is used for running chromogen stain analysis on exported images. This is also the landing page so the tissue array export window can be launched from here.


1. The button to launch the tissue array export window.
2. The button to launch the threshold selector window shown above. This should be used to apply new thresholds before chromogen analysis takes place. The button will turn green once this has been set.
3. Run dab analysis will launch a file selection window where the directory containing the whole dataset of images should be selected. This will begin the chromogen analysis process.
4. This is currently a way to overlay figures of the threshold over the image in a green overlay.
5. This can be toggled to the off position if the user would rather that mask images were not saved as a part of the chromogen analysis process.
6. Main window that displays images during analysis only. It shows first the core image and then the thresholded image at a low resolution.

CHAPTER

FIVE

# OUTPUT FROM QUARRAY

## 5.1 Tissue core threshold output

**Note:** The outputs of the tissue core export process are as follows. They are all saved into the same location in the whole slide image export file.

### 5.1.1 1. Whole core images - indexed with coordinates in the file name

### 5.1.2 Image of the selection panel

This will be created when overlay cores is pressed.

### 5.1.3 2. Figure containing images of all tissue cores in an array

An example of such a figure is given below. A bounding box appears (e.g. green or red) if a pathology/condition map is included in (see *input files*), otherwise bounding boxes are omitted.

### 5.1.4 3. Image metadata in JSON format

The following fields are included.

**path** path of the whole slide image

**coordinates** list of tuple coordinates in the whole slide image

**cores** list of strings to denote core names

**diameter** int for the number of pixels in the image

**scale_index** float to show the scale index between low and high WSI levels

**lowlevel** int to show the level that the low magnification file was taken from

**arrayshape** tuple of the shape of the tissue array in col row format

## 5.2 Chromogen measurement output

The output of the chromogen (e.g. DAB) measurement window is a single excel file (.xlsx) which contains measurements of multiple parameters from each tissue core in the input folder.

The data is organised in a single row for the measurements for tissue each core image.

The column titles and data description appear as follows:

**CoreName** The file name of the analysed tissue image

**AMTsignal** the total number of pixels that contain 'signal' in the image, Also termed coverage

**Mean_intensity** The mean average of the pixel values in the threshold image

**Standard_Dev_intensity** The standard deviation of the pixel values in the threshold image

**AMTtissue** The total amount of tissue in the image (calculation provided in the *measurements page*

**AFperAMTT** The AMTsignal divided by the amount of tissue

**Mean_Intensity_perAMTT** The Mean_intensity divided by the amount of tissue

**SD_Intensity_perAMTT** The Standard_Dev_intensity divided by the amount of tissue

## 5.3 Output file structure

Below is an example of what the file structure could look like for an end to end project after running image export and chromogen analysis.

```
.
├── wsi_001.ndpi
├── wsi_001.xlsx
└── wsi_001_split
    ├── CoreAnalysis_WSI_001.xlsx
    ├── wsi_001_split_A1.png
    ├── wsi_001_split_B1.png
    ├── wsi_001_split_C1.png
    ├── wsi_001_split_D1.png
    ├── wsi_001_split_layoutfig.tiff
    ├── wsi_001_split_metadata.json
    └── wsi_001_split_overlay.tiff
```

CHAPTER

SIX

# MEASUREMENTS AS PERFORMED IN PYTHON

## 6.1 Amount of Tissue

```
def QuantCore(self, image, filename, save=False):
    image = gaussian(rgb2gray(image), sigma=2)
    thresh = threshold_triangle(image[image > 0])
    binary = np.logical_and(image < thresh, image > 0)
    wholeCore = np.sum(binary)
    if save:
        self.info.emit("Saving - " + filename + '_core.tiff')
        imagesave = Image.fromarray(binary)
        imagesave.save(self.inputpath+os.sep+filename + '_core.tiff')
    return wholeCore
```

## 6.2 Amount of Signal

```
def QuantStain(self, image, filename, save=False):
    img_hsv = rgb2hsv(image)
    img_hue = img_hsv[:, :, 0]
    image_sat = img_hsv[:, :, 1]
    hue = np.logical_and(img_hue > 0.02, img_hue < 0.10)  # BROWN PIXELS BETWEEN 0.02␣
↪and 0.10
    if self.threshold:
        stain = np.logical_and(hue, image_sat > self.threshold)  # USER DEFINED␣
↪MINIMUM SATURATION THRESHOLD
    else:
        print("Defult threshold")
        stain = np.logical_and(hue, image_sat > 0.79)  #  DEFAULT SATURATION␣
↪THRESHOLD APPLIED IF NO USER INPUT
    self.current_image = stain[::10, ::10].astype(float)  # SHOW A LOW RESOLUTION␣
↪MASK AS A FIGURE
    self.figures.emit()
    if save:
        self.info.emit("Saving - " + filename+'_stain.tiff')  # SAVE A FIGURE IF␣
↪REQUIRED
        imagesave = Image.fromarray(stain)
        imagesave.save(self.inputpath+os.sep+filename+'_stain.tiff')
    stint = np.copy(image)
    stint = exposure.rescale_intensity(stint, out_range=(0, 255))
    stint[stain == 0] = 0  # array with only the stained pixels
    stint = color.rgb2gray(stint) # convert to greyscale
```

(continues on next page)

(continued from previous page)

```
    stint = np.ravel(stint) # flatten image array
    stint = stint[stint != 0] # remove all zeros
    stint_mean = stint.mean()
    stint_std = stint.std()
    stained = np.sum(stain)
    return stained, stint_mean, stint_std
```

CHAPTER

SEVEN

# ACKNOWLEDGEMENTS

This program is built on top of some truly amazing python packages that are listed in no particular order below.

**OpenSlide** https://openslide.org

**Scikit Image** https://scikit-image.org

**Numpy** https://numpy.org

**PyQt5** https://pypi.org/project/PyQt5/

**qdarkgraystyle** https://github.com/mstuttgart/qdarkgraystyle

**qimage2ndarray** https://github.com/hmeine/qimage2ndarray

**Pillow** https://pillow.readthedocs.io/en/stable/index.html

We are grateful to all of the pathologists and scientists who provided tissue, staining, or helped to test this program. We are grateful to Rui Henrique, University of Porto for tissue and pathology expertise and Mike Millar, University of Edinburgh, for tissue array staining. We are also thankful to colleagues at the University College London Confocal Imaging Facility. Financial support for this project was provided by the Prostate Cancer Research Centre (registered charity no. 1156027), UK grant to Aamir Ahmed.

CHAPTER

EIGHT

# BEFORE YOU START

This program is optimised to to run on most up to date MacOS, Linux, and Windows operating systems.

If there are any problems getting it working then please submit a issue through GitHub.

We would love to receive any pull requests for new features and bug fixes.

CHAPTER

NINE

# QUICK START GUIDE

1. Download the latest stable binary.
2. Download mock data. Previously analysed (Arthurs et al, Sci Rep 2020).
3. Run the program and either:
    - *Export tissue cores* from a Tissue Array.
    - *Analyse DAB expression* in the exported images.