Quantitative Evaluation of Distortion

Q E D

http://github.com/QCaudron/QED

Quentin CAUDRON
qcaudron@princeton.edu

Department of Ecology and Evolutionary Biology
Princeton University

December 22, 2013

Contents

1 Introduction **2**
1.1 Disclaimer . . . 2
1.2 About QED . . . 2
1.3 Acknowledgements . . . 2

2 Installation **3**
2.1 Stand-Alone Installation . . . 3
2.2 Source . . . 3

3 Running QED **4**
3.1 Image Preparation . . . 4
3.2 Launching QED . . . 4
3.3 Selecting the Directory to Analyse . . . 4
3.4 Entering Genotypes . . . 5
3.5 Defining the Eye Region . . . 5
3.6 Optimising the Edgeset . . . 5
3.7 Selecting the Image Genotype . . . 7
3.8 Collating Edgesets . . . 7

4 Results **8**
4.1 Generated Files . . . 8
4.2 Repeat Analyses . . . 8

5 Code **9**
5.1 File Breakdown . . . 9

6 Help **10**
6.1 GitHub . . . 10
6.2 Contact Details . . . 10

1 Introduction

Quantitative Evaluation of Distortion (QED) is a software package, written in Matlab, that assesses the extent of structural differences in biomedical images. QED's website is located at `http://github.com/QCaudron/QED`.

1.1 Disclaimer

Please note that the QED software package comes with absolutely no warrantee, and no guarantee to operate and function as specified or advertised, including being error-free. This software is provided as-is. In no event shall the contributors be held responsible for any damages of any type.

1.2 About QED

This package is geared towards analysing phenotypic differences in *Drosophila melanogaster* eyes, caused by genetic manipulations such as the expression of *tau* protein, or in the general use of fruit flies as neurodegenerative disease models. QED uses implementations of edge detection algorithms to extract information from images, namely the Marr-Hildreth and Sobel methods. It accounts, to some extent, for the large-scale curvature of the eye by a perspective transformation and uses morphological operations on the resulting images to obtain connected edgesets - a pixel-thin net that defines the location of all edges (of a specific width) on the original image. From these edgesets, information is extracted and metrics are computed. Comparing these between genotypes with statistical tests leads to a quantitative evaluation of the difference between physical features of the two genotypes.

1.3 Acknowledgements

QED was written by Quentin Caudron, with contributions from John Aston of the Department of Statistics, and from Ceri Lyn-Adams, Bruno Frenguelli and Kevin Moffat of the Department of Life Sciences, all at the University of Warwick. Special thanks to Madi, Jenya, and Ceri for their excellent beta-testing of the code on Windows computers.

2 Installation

QED can be downloaded, in the form of source code, from the QED GitHub Repository. The Windows installer, due to its large size, can be found at `http://www.quentincaudron.com/qed/QED_Windows_Installer.exe`, and contains both the QED executable and the Matlab Component Runtime libraries (MCR). The MCR libraries are essential for the running of QED on your system, as they contain all of the functions required for QED to work if Matlab is not installed on your computer.

2.1 Stand-Alone Installation

If Matlab is not available on the system, QED can be installed in a stand-alone manner. The Windows installer is available for download on QED's website. Whilst the installer itself is less than 50 KB, please note that in order to maintain QED up-to-date, all installation components are downloaded upon installation. You will therefore need an internet connection to install QED. Please note that this is a relatively large download (approximately 400 MB) and that installation of the MCR component may take some time. Once the MCR libraries have been installed, QED will be downloaded and installed into your *Program Files* directory.

2.2 Source

The source code is available in the GitHub repository. All files should be kept in the same directory, but otherwise, no installation is necessary.

3 Running QED

3.1 Image Preparation

QED will analyse the contents of a folder on your computer. This folder should contain the microscopy images you wish to process, in greyscale (single-channel) `.tif` or `.tiff` format. You should copy images from both the control genotype and the genotype of interest into this folder. The more images present, the more accurate the results. It is the directory itself that QED will take as input - there are no "project files", only directories (processed or unprocessed).

3.2 Launching QED

If you installed QED from the stand-alone executable, it can be run from the Start Menu or Desktop shortcut. Because QED is written in Matlab and uses a large number of external runtime components, it may appear to run relatively slowly. Please be patient and only run the program once ! Whilst graphical elements are demanding on system resources, the information processing and analysis, once underway, is fast. A black terminal window will appear. This window is purely used for verbose output, and in fact, the vast majority can be ignored. Only in the case of a program crash should the contents of this window be studied.
If you have Matlab and are using the source code, simply call `qed` from the command line. Any warnings or errors will be directed to Matlab's standard output.

3.3 Selecting the Directory to Analyse

You will be prompted to select a folder to analyse. Browse through your system, select the relevant folder, and click OK. Please remember, this folder should contain only the image data from the two genotypes that you wish to compare, even though only `.tif` and `.tiff` files are considered by QED. If you wish to redo an older analysis, you may select the same folder, which will now also contain files generated by QED.

3.4 Entering Genotypes

The first time a folder is processed, you will be asked to enter the names of the two genotypes. You should ideally enter plain text, though spaces, dashes and underscores are acceptable. Examples are, `GMR`, `GMR-Tau` or `Control Eyes`. Press *OK* when done.

3.5 Defining the Eye Region

QED will begin processing images alphabetically. You are first told what the filename of the current image is, and asked to cut around the eye. This is done by clicking around the eye. Each time you click, a point is placed on the image. New points automatically connect to the last point, so you should click around the eye to draw a polygon, following the contour as closely as possible. Once you are finished, either click on the first point you placed to close the shape, or double-click while placing your final point.

At this stage, all points can be moved by clicking and dragging them, and the entire shape can be moved by clicking in the middle of the polygon and dragging. You can delete a point by right-clicking on it and selecting *Delete*. If you wish to add a point, hover the mouse over the edge of the polygon and press the *A* key on your keyboard, and then click to insert a new point. Once you are satisfied with the cutout, double-click inside the shape to confirm your selection.

3.6 Optimising the Edgeset

Once you have defined the eye region, you will be shown the original image with the edgeset superimposed, alongside an edgeset, for clarity. You can then modify two parameters and one setting in order to obtain a better edgeset.

Gaussian Kernel Width

The first parameter is the width of the Gaussian kernel, which defines the width of edges that are extracted from the image. As an example, a 650×650-pixel image, close-cropped onto the eye of a wild-type fly, may have edge widths approximately between four and six pixels. The default value in QED is set to five pixels, because the images used during testing were of a similar resolution - however, you should experiment with the Gaussian

kernel width in order to find one that suits the resolution of your images, and also that of the particular image. The selected value must be in the correct edge width range, or the algorithm will begin to pick up features that are not true edges, and miss those that are.

Sobel Threshold

The second parameter is the Sobel threshold. This defines how sharp changes in luminosity should be in order to be considered edges. At this stage in the processing, the image is nearly black and white, so this threshold applies to how suddenly changes in the image occur, not necessarily how large those changes are. The default, for the above image dimensions, is four. Lowering this tends to bring out more edges, but also more noise, whereas increasing it will break up edges. Because it is very important to have connected edgesets, this parameter should be set to remove as much noise as possible, but not be set so high as to begin breaking up edges. The lower bound for this threshold is zero, and it does not necessarily have to take integer values. If extreme sensitivity is required, the Sobel threshold can be dropped to 0.001 or even below, for example. On the other end of the scale, if the edges are very well-defined, a higher threshold may be better, and can be increased to values in the tens or above.

Contrast Enhancement

You can apply Contrast-Limited Adaptive Histogram Equalisation to the image, by using the checkbox below the parameter modification area. You will see the original image on the left change, such that contrast is enhanced. This is good if the original image contains a shadow, for example, but is not appropriate for all images. If you select this contrast enhancement, the luminosity values of the image will change, and the above parameters will thus need to be altered. The important parameter to modify, in the case of a contrast-adapted image, is the Sobel threshold, which should generally be increased to account for the increased contrast in the image.

Image Rejection

Modifying these parameters is possibly equal parts science and art, and will require a large amount of trial and error. For the example image dimensions

listed above, it could be found that decreasing the Gaussian kernel width to four, and increasing the Sobel threshold to approximately fifteen, can provide an improvement on the default values.

If, however, an adequate edgeset cannot be found, the option to reject the image exists. The image will simply be moved to another folder (and not deleted), an edgeset will not be generated, and QED will ignore its existence during the analysis. Images may contain artifacts from the microscopy, which may lead to edgesets always containing a non-negligible number of open ommatidia due to shadows or the detection of structures created in the sample's preparation for imaging. In the prior case, images can sometimes be cleaned up using image editing software. In case an image cannot be salvaged to generate an image set, it should be rejected.

3.7 Selecting the Image Genotype

After accepting the edgeset for an image, you will be asked to specify the genotype of that image, and will be presented with a list of the two genotypes you entered earlier. Make your selection, and click *OK*.

3.8 Collating Edgesets

Once all images in the directory have been processed, QED will gather all edgeset data into memory and begin extracting structural information from them. This information is used in the metrics that are used to compare between genotypes : roundness of ommatidia, distances between ommatidia and angles between ommatidia around the centre of the eye. You are then presented with a results graph window displaying the cumulative distributions of the roundness dispersion coefficients, which is automatically saved in both `.eps` and `.tif` formats in the *Results* directory, created in the folder being analysed.

4 Results

4.1 Generated Files

The graphs that appear on your screen show the dispersion coefficients for each metric. A dispersion coefficient is a measure of how wide a distribution is, and thus, the lower it is, the more regular and ordered a measure is. Dispersion coefficients are calculated for ommatidial roundness and for interommatidial distances and angles. For a control genotype, dispersion coefficients are expected to be small, as we expect hexagonal packing (60° and equal distances between ommatidia) of ommatidia, which should all be round. When ommatidia begin to fuse, they lose their roundness (they are now oval or more "blob"-shaped) and the regularity of the hexagonal lattice is reduced, leading to higher dispersion coefficients.

Each graph shows two sets of points. Blue points are the first genotype, which should be your control. Points in red are from the second genotype. Each point represents the dispersion coefficient for a particular image, and the three graphs show these for the three different measures.

Numerical results are also saved in the *Results* folder, as `.csv` files. Dispersion coefficients are saved in three files - one per metric - with the left column containing the first (control) genotype. Full results for metrics are also saved in the *Results* directory, under *Raw Data*.

Images showing the progress of edge detection are also generated, in folders created for each image.

4.2 Repeat Analyses

If you wish to add or remove images, or simply attempt to improve an edgeset, you should be aware of the files created by QED. The folder you have just analysed will now contain the following :

- The `.tif` or `.tiff` images you placed there for analysis
- A `.qed` file for each image, containing genotype and edgeset information
- `genotypes.qed`, a file containing information regarding the genotypes being analysed

- A `Rejected` folder, containing any images which had their edgesets rejected
- A `Results` folder, containing numerical and graphical results from the analysis

The second item in this list determines whether or not an image will be analysed. Any image in the directory without an associated `.qed` file will be analysed, and you will be prompted to create an edgeset for it. Any image with an associated `.qed` file will be analysed, but its edgeset will have been already created, allowing repeat analyses of a directory to be done much more rapidly. If you wish to recreate the edgeset for an image, simply delete its `.qed` file and rerun the program. Likewise, to add images to be analysed, simply copy the image into the directory along with the other images, and an edgeset will be created for it when you run QED. If you believe an image should not be being analysed, delete its `.qed` and move the `.tif` or `.tiff` into the *Rejected* folder.

5 Code

The code for QED is available for download on its website.

5.1 File Breakdown

- `qed.m` is the central file. It prompts for a directory and genotype information, collates edgeset results, and calculates the metrics from the extracted information. It also computes the p-values from the Mann-Whitney test and generates the results plot as well as saving all results to file.
- `qextract.m` opens an image, prompts the user to cut out the eye region and opens the GUI for generating the edgeset. It applies the perspective transform to the edgeset and searches for boundaries.
- `QGUI.m` is the functionality behind the edgeset creation. It defines the behaviour of buttons on the GUI.
- `qedgeset.m` returns the edgesets from an image, given the Gaussian kernel width and Sobel threshold values.

- `qdispersion.m` calculates a dispersion coefficient given a distribution.
- `qgauss2D.m` computes the 2D derivates of a Gaussian.
- `qsurface.m` generates a semiellipsoidal surface based on x and y dimensions, with an empirically-defined curvature.
- `qresult.m` generates CSV files containing results.
- `QGUI.fig` contains information on what the GUI looks like.
- `itriu.m` extracts half of a diagonal matrix.

6 Help

Please open issues on GitHub if you find any bugs or problems. Otherwise, feel free to contact the author with any questions or queries.

6.1 GitHub

6.2 Contact Details

Email :	`qcaudron@princeton.edu`
Address :	Quentin CAUDRON
	Department of Ecology and Evolutionary Biology
	106A, Guyot Hall
	Princeton University
	Princeton, NJ
	08648, USA