QuimP11b Guide

Richard Tyson and Till Bretschneider

Systems Biology DTC, Warwick University

Correspondence to Till Bretschneider. Email: Till.Bretschneider@warwick.ac.uk

June 3, 2014

Web site: `http://go.warwick.ac.uk/quimp`

Contents

1 QuimP11b - Whats new?

1. Fixed several minor bugs.
2. Updated and tested with ImageJ 1.49a and MATLAB 2014a
3. The ECMM and Q Analysis plugins will search for other .paQP in a directory and prompt to batch process files.
4. BOA prompts to check image scale.
5. BOA can read in a previous segmentation using the LOAD button
6. When editing segmentations the user can scroll between frames without leaving edit mode

If you spot a bug you can email richard.tyson@warwick.ac.uk.

2 Installation

QuimP11 can be downloaded from `http://go.warwick.ac.uk/quimp` and is written as a set of ImageJ plugins. ImageJ can be downloaded here: `http://imagej.nih.gov/ij/download.html`. Install the latest version of ImageJ (QuimP11 requires 1.43 or later). Unzip *QuimP11b_WSB.zip* and copy the file *QuimP11b.jar* into the *plugins* folder located in the installation directory of ImageJ.

QuimP11 will also function with Fiji, an extended version of ImageJ (`http://pacific.mpi-cbg.de/wiki/index.php/Fiji`).

3 QuimP11 Workflow

There are four stages (and four plugins) to completing an analysis using QuimP11.

1. Cell segmentation - Performed by the **BOA** plugin. Cell outlines, represented as a chain of nodes, are extracted from each frame of an image sequence using an active contour. BOA outputs global measures, such as cell centroid displacement, or cell area.
2. Membrane tracking - Performed by the **ECMM Mapping** plugin. Outlines are mapped between frames to extract local membrane velocities.
3. Measuring Fluorescence - Performed by the **ANA** plugin. Pixel intensities are sampled from the cell's cortex, its width defined by the user.
4. Data analysis - Performed by the **Q Analysis** plugin. Statistics are generated and data are visualised in the form of spatial-temporal maps.

Once complete, data can be analysed in Excel or MATLAB.

A plugin can be launched from either the [*Plugins* → *QuimP11b*] menu, or from the QuimP bar (opened via [*Plugins* → *QuimP11b* → *QuimPBar*]). Running a plugin will either require an image and/or a QuimP parameter file (with extension *.paQP*) generated by BOA. Plugins must be executed left to right, although steps can be repeated without re-running previous plugins and fluorescence measuring with ANA can be skipped entirely.

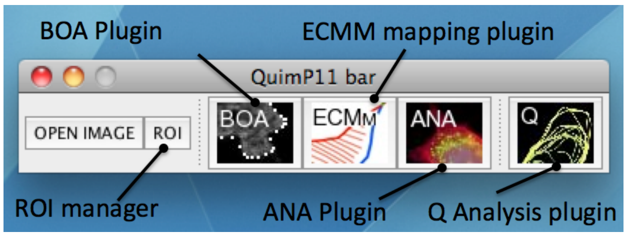


Figure 1: The QuimP bar

4 Cell Segmentation - BOA Plugin

QuimP utilises an active contour to segment cells from the background of an image. Typically, this will be a time lapse movie containing a fluorescence channel, captured from a confocal microscope, but the BOA plugin will segment any image which has bright objects on a dark background (note that phase contrast images can not be segmented with this method). Attaining the best image for segmentation may require some pre-processing, such as the combining of available fluorescence channels (the ideal case is that of an evenly white cell on a black background). Figure 2 describes the active contour method and provides insight as to BOA's parameters.

Step 1: Open an image and ensure the correct scale. QuimP11 can segment from a single image, or from an image stack (but not from a hyperstack). Only 8-bit images can be used for segmentation and so you will be prompted to convert to 8-bit if required. QuimP11 uses the scale properties of the image to scale all output to microns (pixel size) and seconds (frame interval). **ENSURE YOUR SCALES ARE CORRECT** within [*Image* → *Properties...*]. On launching, BOA will prompt to check your image scale. The scale is recorded by BOA and stored in the parameter file for use in the rest of the analysis.

Step 2: Launch the BOA plugin

Unlike previous versions, BOA no longer requires selection of cells prior to launching, but you can do so. Launch BOA from the QuimP bar or menu. The BOA window is now a fully functional ImageJ window that allows the user to perform any ImageJ function, such as drawing, creating ROI's, or zooming while the BOA plugin is in use. Figure 3 shows the BOA window. The image scale is displayed at the top right. To adjust the scale click *Set Scale.*

BOA can segment one, or more cells, each beginning and ending at any frame you wish. In addition, you can manually edit any segmentation at any frame, add/delete cells, truncate segmentations, and adjust the segmentation parameters for individual frames.

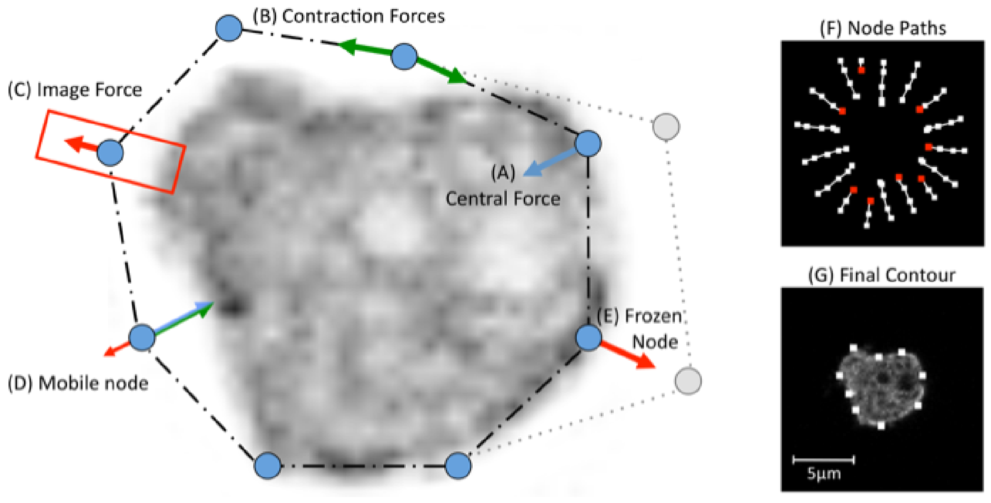


Figure 2: Caricature of outline detection using active contours. QuimP's active contour consists of a chain of connected nodes (a reduced number are shown here for simplicity) initialised to encircle a cell of interest. Three types of forces act on each node; (A) Central forces contribute to chain shrinking; (B) Contraction forces between neighbouring nodes shrink the chain and maintain chain integrity; (C) Image forces oppose the shrinking of the chain by a magnitude determined by the local image intensity gradient (red box). When a node experiences inward forces greater than the opposing image force it moves inwards (D). On reaching the boundary the image force becomes sufficiently large to cancel out the other forces, halting and freezing the node (E). (F) shows the paths of nodes during contraction. Note that nodes can be added or removed (red nodes) to maintain an average distance between neighbours. (G) shows the final position of nodes. QuimP usually makes use of 100 or more nodes to extract high resolution cell outlines. [Tyson, R A *et al*, High resolution tracking of cell membrane dynamics, Math. Model. Nat. Phenom. Vol. 5, No. 1, 2010, pp. 34-55]

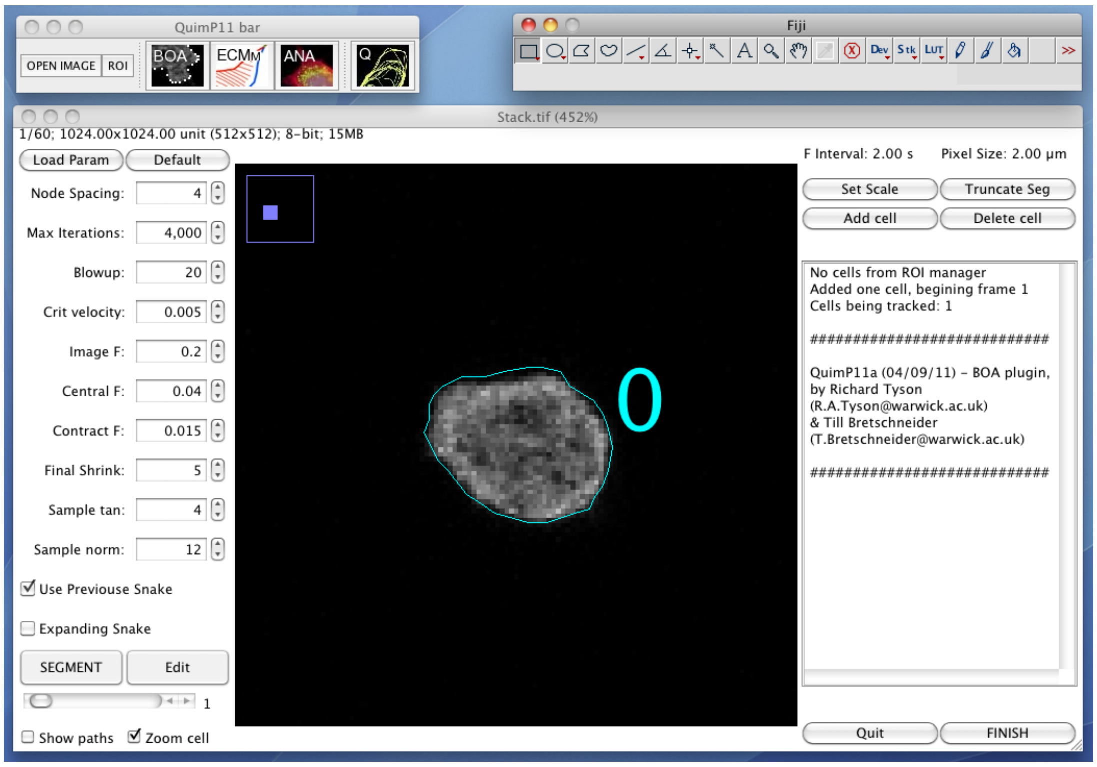
Step 3: Adding and deleting cells.

To add a cell first scroll to the frame that you want to begin the segmentation at (either with the mouse wheel, or slider). Using either the circular, rectangular or polygon selection tool, draw an ROI (region of interest) around a cell, and click *Add cell*. BOA will immediately attempt to segment the cell at the current frame. If convergence is successful, the result is displayed as an image overlay. The cell is also given an identification number. If the segmentation fails for some reason, the parameters can be adjusted (see below). You do not need to re-add the cell.

Additional cells can be added in the same way. The active contours surrounding each cell will interact with one another and prevent contours crossing over to other cells. This capability is particularly useful in situations where cells come into close contact, which often disrupts the segmentation process.

To delete a cell click *Delete cell*, and then click the centre marker of a cell, on any frame where a segmentation is visible. Hit the same button again to leave delete mode.

Step 4: Adjusting segmentation parameters.. When a parameter is altered, BOA immediately re-computes the segmentation for all cells on the current frame with the new parameters (all cells share the same parameters). Each has a minimum/maximum value that can not be exceeded. To alter a parameter either enter a value into the text box, or use the arrows to increase/decrease in



1/60; 1024.00x1024.00 unit (512x512); 8-bit; 15MB

F Interval: 2.00 s   Pixel Size: 2.00 µm

Node Spacing:	4
Max Iterations:	4,000
Blowup:	20
Crit velocity:	0.005
Image F:	0.2
Central F:	0.04
Contract F:	0.015
Final Shrink:	5
Sample tan:	4
Sample norm:	12

☑ Use Previouse Snake

☐ Expanding Snake

☐ Show paths ☑ Zoom cell

```
No cells from ROI manager
Added one cell, begining frame 1
Cells being tracked: 1

############################

QuimP11a (04/09/11) - BOA plugin,
by Richard Tyson
(R.A.Tyson@warwick.ac.uk)
& Till Bretschneider
(T.Bretschneider@warwick.ac.uk)

############################
```

Figure 3: The BOA Window

steps. First, try adjusting the *Image F* parameter to improve the result.

The parameters are as follows:

- **Node spacing** - Roughly equates to the number of pixels between nodes, and hence the resolution of the segmentation.
- **Max iterations** - The maximum number of iterations applied to shrink the contour.
- **Blowup** - The number of pixels to expand the solution to the previous frame, for beginning calculation at the current frame (utilised when 'Use previous snake' is active). Increase if your target cell is moving by large amounts between frames. Decrease if other cells are coming into close proximity and interfering with the snake. Lower values reduce computation time.
- **Critical velocity** - The cut-off speed at which nodes are frozen. Lower the value to allow the snake to continue to contract for longer, which may aid in letting nodes enter concavities.
- **Image F** - Controls the magnitude of the image force pushing nodes outwards. This is the most effective parameter and should be the first port of call for adjustment. Highly intense cells will produce high image forces, and you may need to lower this value to allow nodes to approach the cell outline. With weakly intense cells the snake may collapse inwards, past the cell boundary, in which case increase the image force.
- **Central F** - The magnitude of the central force pulling nodes inwards. Increase to shrink the snake further and help nodes enter concavities. Decrease to prevent nodes entering the interior of cells.

- **Contract F** - Magnitude of the force pulling nodes together (i.e. contracting the snake). Increase to prevent nodes 'falling through holes', or to produce a smoother final solution.
- **Final shrink** - The number of pixel to shrink the solution to tighten the snake to the cell boundary. A value of zero does not shrink the solution at all.
- **Sample tan, Sample norm** - Defines (in pixels) the size of the sampling box around each node within which the image is sampled for calculation of the image force. 'tan' describes its width, 'norm' its depth. A larger sampling box will increase the size of the image force. Briefly, the sampling box should be an appropriate size as to give an accurate sample of the local intensity between the cell and the background.
- **Use previous snake** - If selected, the solution to the previous frame will be used as the starting contour for the current frame. Otherwise, the users selection is used as the starting contour for every frame.
- **Expanding snake** - Experimental option. Rather than contract the snake, it expands. May be useful for tracking vesicles.

The key to attaining a good segmentation is the balancing of the 3 forces. Previous segmentations for your image can be loaded back into BOA by hitting load and selecting a QuimP parameter file with the file extension *.paQP*. Parameter values alone can be loaded by declining to load associated snakes.

Enabling the tick box labelled *Show paths* will display a trace of the snake as it contracts. Nodes colour pixels white as they move towards the cell boundary. This view can be useful for diagnosing why a segmentation fails.

Enabling the tick box labelled *Zoom cell*, will zoom the image to the first cell added, and track the zoom to the cell's movements.

Step 5: Segmenting multiple frames. When happy with the segmentation on the first frame, hit *SEGMENT* to segment the cell (or cells) in the following frames. The algorithm may fail at a particular frame and a warning will be printed to the log window, at which point you can alter parameters.

Step 6: Correcting segmentations. In the case of failure, or incorrect results, you have 3 options:

- **1 - Edit a solution.** Scroll to a frame where an error occurred, click *EDIT* (If you have more than one cell click its centre). Drag the points of the ROI to the correct locations (you can hold *alt* to delete nodes or *shift to split nodes*). You can zoom using the imageJ tool (magnifying glass). If you loose your ROI go to [*Edit* $\rightarrow$ *selection* $\rightarrow$ *restoreselection*]. Scroll to edit other frames. When finished, leave edit mode by clicking **STOP EDIT**.
- **2 - Adjust the parameters.** Scroll to a frame where an error occurred and change the parameters to improve the segmentation. Optionally, click *SEGMENT* to apply these new parameters to the following frames.
- **3 - Truncate segmentations.** Scroll to the first frame where an error occurred and click *Truncate Seg'*. Click the centre of a cell to remove all segmentations from the current frame onwards.

Step 7: Save the results. Click *FINISH*. The saved cell outlines are those visible on the blue overlay. A set of files will be outputted for each segmented cell. A cell's ID number is automatically added to filenames.

Enter a name for the analysis (file name extensions are added automatically). BOA outputs files with a *.xxQP* extension.

Files extended with *.paQP* contain file paths and parameters associated with a particular analysis. When running other QuimP11 plugins, it is the *.paQP* file which you must select to continue an analysis (see 8.1).

Files extended *.snQP* contain all data associated with individual nodes of an outline, including pixel coordinates, local cortex intensity, local membrane velocities, and tracking data (see 8.2).

Files extended *.stQP.csv* contain average statistics per frame, and can be opened directly in Excel as a 'comma separated file' (see 8.3).

The BOA plugin alone outputs many useful statistics regarding cell morphology and migration, without the need to run any further analysis. Simply open the *.stQP.csv* file in Excel, or use the accompanying scripts to load data into MATLAB (see section 9).

4.1 A Few Words on Image Segmentation

If you run into difficulties attaining a good segmentation, there are a few tricks that may help. If the width of your cell is around 20 pixels or less try artificially increasing the size of the image with some form of pixel interpolation selected ([$Image \to Adjust \to Size$]). This should aid sampling of the image and allow nodes to enter concavities with greater ease (note, all subsequent analysis must be performed on the image at the new resolution).

If your images are particular noisy, and cell's appear very grainy, you may try applying a weak Gaussian blur to smooth them ([$Process \to Filters \to GaussianBlur$]). This should make for more reliable sampling of the cell's interior.

Remember to recalculate the scale of the image when resizing, and to perform intensity analysis on the rescaled image without any interpolation.

Another common approach to improving image quality is to remove background noise using a 'blank' image, one taken without any cells in the field. This is particularly useful for removing shadows of dirt on the optical equipment, and correcting uneven backgrounds.

5 Cell Tracking - ECMM Plugin

To extract data on local membrane velocity QuimP11 utilises our Electrostatic Contour Migration Method (ECMM). Nodes that form a cell outline at time t are mapped onto the segmented outline at time t+*1*. The distance a node migrates during the mapping process, and the frame interval, determines the local membrane velocity at the node's position. If you only have a single frame there is no need to run this plugin. ECMM edits and adds to data already contained within an *.snQP* file.

Step 1: Launch the ECMM Mapping plugin, and open the desired *.paQP* file when prompted. There are no parameters that require manual setting.

Step 2: Inspect the results. The blue outline represents the cell outline at time t, the green at time $t + 1$. Green circles, overlaid with crosses, denote intersection points that have been validated and used for calculating sectors. Red lines denote paths taken by migrated nodes. Nodes that fail to map correctly are highlighted in yellow. Failed node (FN) warnings are displayed along with a running total (in brackets) of the number of failed nodes in the whole sequence. The log window will display progress and warnings.

If a node fails to map it is simply ignored, the effect of this is a small reduction in mapping resolution at the nodes location. If the process fails to map the majority of the nodes correctly, then a new segmentation will have to be performed (try using a different node resolution or changing the image resolution).

5.1 ECMM Tracking

The details as to the workings of ECMM can be found in our publication; High resolution tracking of cell membrane dynamics. The current implementation includes several significant improvements. It is not necessary to understand the method, only the format of the tracking.

At frame $t = 1$ a node is randomly chosen as being node zero, n_0. Other nodes are then assigned a position according to their distance from n_0 along the cell perimeter. The length of the cell perimeter is normalised to 1, hence the position of a node which is exactly half way around the perimeter will be 0.5 (see figure 4a).

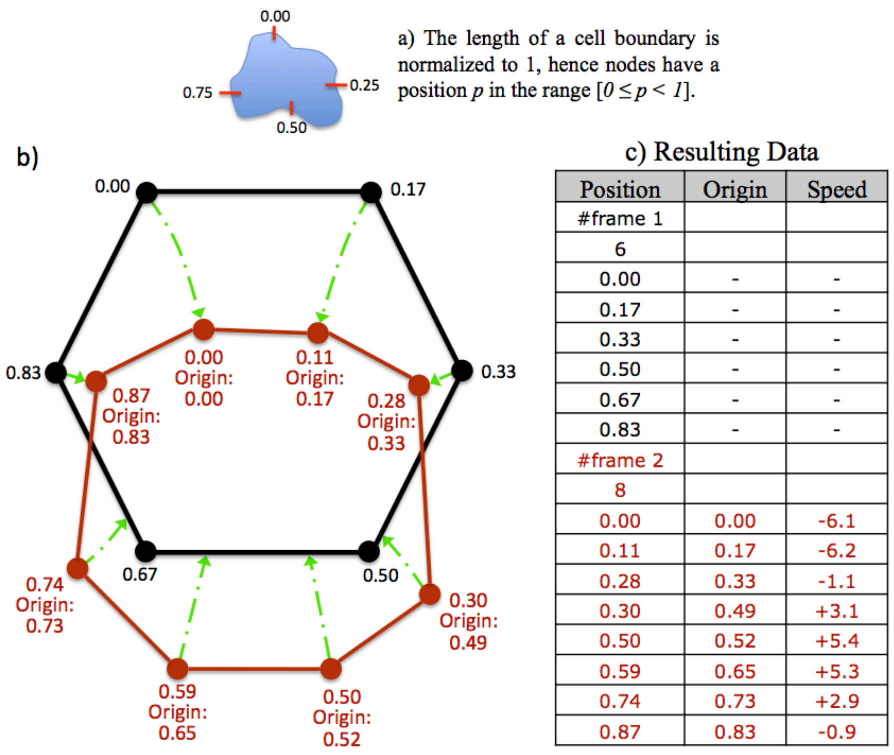
The nodes of the outline at $t = 2$ also have positions assigned, but in addition have an *origin*. This represents where a node originated from on the $t = 1$ outline. For example, a node with an origin of 0.5 originated from position 0.5, exactly half way around the outline at $t = 1$.

It should be realised that this method of tracking is sub-node resolution. By simply interpolating the positions and origins of nodes, any real number position on an outline can be tracked at sub-node resolution, for as many frames as desired. See section 8.2 for the format of data output, and section 9.3 for using ECMM data within MATLAB.

6 Fluorescence Measurements - ANA Plugin

QuimP11 samples the intensity of pixels from within the cortical region of a cell. At each frame, nodes of the cell outline are shrunk inwards to form an inner outline. The amount of shrinkage is specified by the user, and relates to the estimated cortex width. Nodes are migrated inwards towards the inner outline, while simultaneously measuring pixel intensities (a 3x3 average). A nodes fluorescence intensity is the maximum recorded as it migrates across the cortex. ANA can record data for up to 3 separate fluorescence channels.

Step 1: Open an image stack containing the desired fluorescence channel. This may be the same stack used for cell segmentation, or one containing data from another channel. Ensure the



a) The length of a cell boundary is normalized to 1, hence nodes have a position *p* in the range [*0* ≤ *p* < *1*].

c) Resulting Data

Position	Origin	Speed
#frame 1		
6		
0.00	-	-
0.17	-	-
0.33	-	-
0.50	-	-
0.67	-	-
0.83	-	-
#frame 2		
8		
0.00	0.00	-6.1
0.11	0.17	-6.2
0.28	0.33	-1.1
0.30	0.49	+3.1
0.50	0.52	+5.4
0.59	0.65	+5.3
0.74	0.73	+2.9
0.87	0.83	-0.9

Figure 4: **ECMM tracking method. a)** Normalisation of the cell perimeter. **b)** Caricature of the tracking of node positions between two frames (black t, red $t + 1$). Arrows pointing towards the red outline indicate forward mappings and a nodes origin is recorded simply as its position on the black outline. Arrows pointing towards the black outline indicate reverse mapping and node origins are recorded as interpolated values between nodes on the back outline. **c)** Data from *b)* as it would appear in a *.snQP* file. The frame number is followed by the number of nodes in the outline. Speeds are proportional to the lengths of the green mapping arrows.

opened stack has the same resolution and number of frames as that used for segmentation.

Step 2: Launch the ANA plugin and open the desired *.paQP* file when prompted.

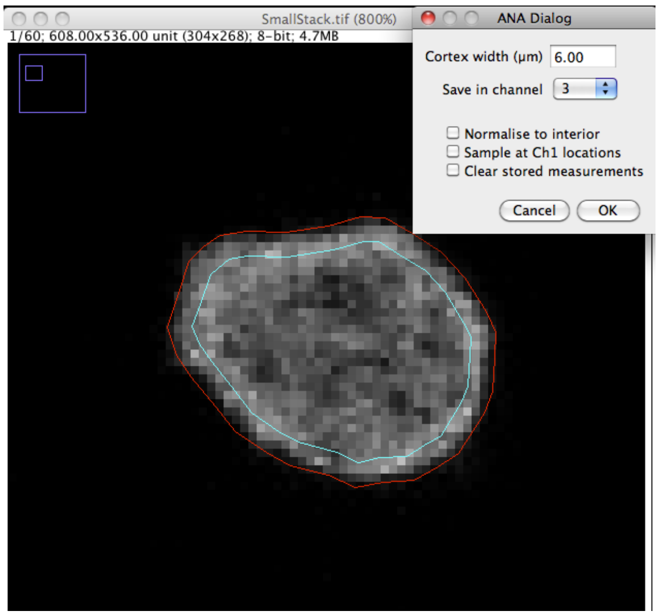
Step 3: Enter a value for the cortex width (μm) and choose a channel. The inner and outer outlines are displayed on the image. Changing the cortex width will update the image overlay. Choose a channel to record data in from the drop down menu.

ANA provides 3 further options

- Normalise to interior - Toggle normalisation of intensity sampling to the cells interior (recommended).
- Sample at Ch1 locations - Sample at the same locations as determined for channel 1.
- Clear stored measurements - Removal all fluorescence measurements from QuimP's files.

Click *OK*.

Fluorescence data associated with specific nodes is added to the *.snQP* file, while whole cell data (e.g. total cortex fluorescence) is written to the *.stQP.csv* file.



Cortex width (µm) 6.00
Save in channel 3
☐ Normalise to interior
☐ Sample at Ch1 locations
☐ Clear stored measurements

Figure 5: ANA. The area between the red and blue contours is the user defined cortex, and is where intensity samples are taken. If *Normalise to interior* is ticked, sampled intensities will be normalised to the average intensity of pixels within the blue contour.

Step 4: For further channels, open the next channel image and re-run ANA.

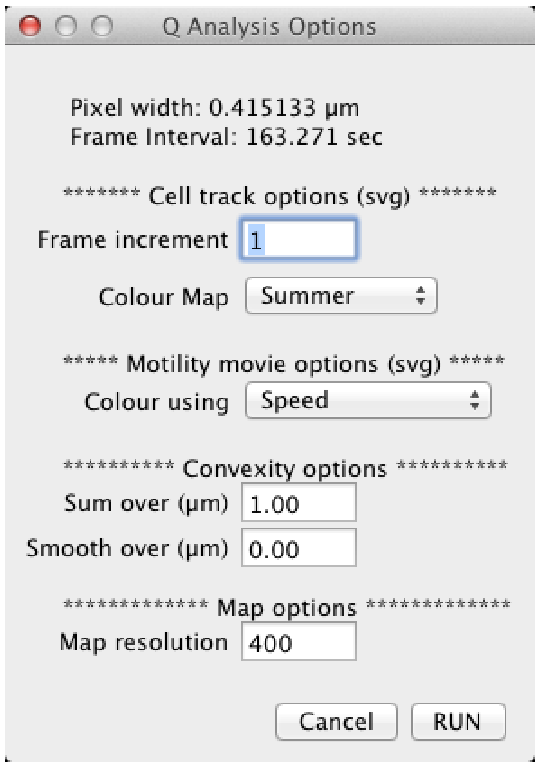
7 Compiling Data - Q Analysis Plugin

Data within *.snQP* and *.stQP* files can be read into Matlab at this point, but motility maps have not yet been generated. The Q Analysis plugin will build spatial-temporal maps of motility, fluorescence, and convexity, and also several other maps to aid analysis. In addition, vector graphics are produced using the *Scalable Vector Graphics* format. These include a cell track in which all cell outlines are overlaid and coloured according to frame number, and a motility movie in which nodes are coloured with a user specified data type.

Step 1: Launch the Q Analysis plugin and open the desired *.paQP* file when prompted.

Step 2: Enter parameters to the options dialog. The parameters are as follows:

- **Cell track option - Frame increment** [positive integer]. The frame increment for drawing the cell track (*_track.svg*). Frames are only draw at the specified increment. Set to 1 for all frames.
- **Cell track option - Colour Map**. Select a colour scheme for the cell track.
- **Motility movie option - Colour using.**. Select the data type for use in colouring nodes of the motility movie.
- **Convexity option - Sum over (microns)** [positive real number]. Distance around the cell perimeter (μm) to sum curvature. Set to zero for no summation.



Q Analysis Options

Pixel width: 0.415133 µm
Frame Interval: 163.271 sec

******* Cell track options (svg) *******

Frame increment: 1

Colour Map: Summer

***** Motility movie options (svg) *****

Colour using: Speed

********** Convexity options **********

Sum over (µm): 1.00

Smooth over (µm): 0.00

************* Map options *************

Map resolution: 400

Figure 6: Q Analysis options dialog

- **Convexity option - Smooth over (microns)** [positive real number]. Distance around the cell perimeter (μm) to over which to average curvature for smoothing. Set to zero for no smoothing.
- **Map option - Map resolution** [positive integer]. A value of x will produce images that are X pixels in width. The height of the image will be equal to the number of frames of the analysed sequence, or a multiple of number of frames (if less then x). Data is interpolated to form maps of any size.

The images produced are for visualisation only, as they have been scaled to fill the RGB colour spectrum. Vertical image scale is set to frames (frame zero at the top), horizontal to the normalised length of the cell perimeter (0,1]. Import *.maQP* files as text images for further analysis in ImageJ. See Section 8.4 for details and analysis of maps.

8 Data Files Explained

8.1 Parameter Files - *paQP* Files

A parameter file contains text and has the extension *.paQP*. Both QuimP11's plugins and MATLAB functions use the parameter file to locate files associated with an analysis and also to store data such as the pixel size, frame interval, and start frame. In addition, it stores the parameters used for segmentation (the last used parameters).

The first line identifies the file as a QuimP parameter file, and contains the creation date. The second is random identifier. The third is the absolute path to the image used for segmentation.

In general, the contents of a parameter file should not be changed, but you may wish to alter the paths to image files if you later move them.

8.2 Snake Data - snQP files

The *.snQP* file contains data relating to the nodes of cell outlines (*snake* is a reference to active contours being described as snakes). Comment lines begin with a #. Data for frame f begins with the comment line *#frame f*, followed by a single integer value indicating the number of nodes that make up the outline (N). The following N rows hold the data for the nodes. The columns are as follows:

- **Node Position** - The normalised position of the node in relation to node zero, n_0 (see section 5.1).
- **X_coord** - The horizontal pixel coordinate of a node on the image used for segmentation.
- **Y_coord** - The vertical pixel coordinate of a node on the image used for segmentation.
- **Origin** - The position from which a node originated from on the outline at the previous frame (see section 5.1).
- **Global Origin** - The position from which a node originated from on the outline at the first segmented frame (see section 5.1).
- **Speed** [microns per second]- The speed at which a node travelled between frames, as determined by ECMM.
- **Fluor_Ch$** - The intensity value assigned to a node by ANA (see section 6) for the channel number denoted by $. This value is the maximal $3x3$ average intensity measured at the node's local cortical region.
- **Ch$_x** - The horizontal pixel co-ordinate on the fluorescence image at the point where Fluor_Ch$ was measured, i.e. the co-ordinate where the maximum intensity occurred.
- **Ch$_y** - The vertical pixel co-ordinate on the fluorescence image at the point where Fluor_Ch$ was measured, i.e. the co-ordinate where the maximum intensity occurred.

  It is possible to store data for 3 fluorescence channels ($\$ = \{1, 2, 3\}$) within the *.snQP* file. Missing data is denoted by negative values.

8.3 Frame statistics - stQP files

The statistics file, with the extension *.stQP*, contains whole cell statistics for each frame (rather than node associated data). Data is written as *comma separated values* which can be opened easily as a spreadsheet. The file is split into two section. Data in the top section is computed by BOA and relates to cell morphology and movement of the cell centroid. Data in the lower section relates to cell fluorescence and was computed by ANA (data is listed separately for all 3 available channels). Missing data is denoted by -1.

The cell centroid is computed as the weighted centre of the polygon formed by the cell outline. The measures computed by BOA are as follows:

- **X-Centroid** [*pixels*] - Horizontal pixel co-ordinate of the cell centroid.
- **Y-Centroid** [*pixels*] - Vertical pixel co-ordinate of the cell centroid.
- **Displacement** [*microns*] - Distance the cell centroid has moved from its position in the first recorded frame.
- **Distance Travelled** [*microns*] - Sum of the centroid displacements between frame, i.e. the total distance over which the centroid has moved.
- **Directionality** - Persistence in direction, calculated as $Displacement/Dist.Travelled$ (chemotaxis index). A value of 1 reveals that a cell has moved in a straight line. Decreasing values denote a cell moving increasingly erratically.
- **Speed** [*microns per second*] - Speed at which the centroid moved between the current and previous frame.
- **Perimeter** [*microns*] - Length of the cell perimeter (segmented outline).
- **Elongation** - An ellipse is fitted to the cell outline and the major/minor axis used to compute the elongation of the cell's shape. $Elongation = major\ axis/minor\ axis$. Note that a value of 1 does not necessarily represent a perfectly circular cell, only a circular fitted ellipse.
- **Circularity** - A measure of circularity defined by the following equation: $\frac{4*PI*Area}{Perimeter^2}$. A value of 1 reveals the cell's outline to be perfectly circular.
- **Area** [*microns*^{2}] - Cell area.

The measures computed by ANA are as follows:

- **Total fluo.** [*pixelintensity*] - Total fluorescence. Sum of all pixel intensities within the cell outline.
- **Mean fluo.** [*pixelsintensity*] - Mean fluorescence. Average intensity of pixels within the cell outline.
- **Cortex width** [*microns*] - Width of the cortex, as specified by the user (see section 6).
- **Cyto. area** [*microns*^{2}] - Area of the cytoplasm (area of the whole cell minus the cortex area).
- **Total cyto. fluo.** [*pixels intensity*] - Sum of all pixel intensities within the cytoplasm.
- **Mean cyto. fluo.** [*pixels intensity*] - Average pixel intensity within the cytoplasm.

- **Cortex area** [$microns^2$] - Area of the cortex.
- **Total cortex fluo.** [*pixels intensity*] - Sum of all pixel intensities within the cortex.
- **Mean cortex fluo.** [*pixels intensity*] - Average pixel intensity within the cortex.

8.4 QuimP Maps - maQP files

The images produced by Q Analysis are spatial-temporal maps of motility, convexity, and fluorescence. The vertical axis represents time, with frame 1 at the top. The horizontal axis represents positions along the cell outline, the length of which has been normalised to 1 at every frame. Outlines can be thought of as having been 'cut' at position zero, laid along the horizontal axis, and stacked below one another, hence maps are cylindrical (wrap around onto themselves on the horizontal axis).

The node randomly designated as begin node zero (n^0), at frame 1, is mapped to the top left pixel. Data from ECMM is used to track the position of n^0 throughout all the frames, and is always positioned at the left most pixel at each frame. All other nodes are positioned along the horizontal axis according to their distance from n^0 along the cell perimeter. Values between nodes are estimated using linear interpolation.

If the number of frames is below the specified map resolution, then maps are scaled vertically, hence a row of pixels may be an interpolated value between frames.

As well as producing *.tif* images, maps are also saved as text images with the extension *.maPQ*. Unlike the *.tif* maps, these are not scaled vertically, so each line of values contains the data for a single frame. It is these maps that should be used for further analysis.

Similarly, four additional maps are produced to aid further analysis:

- **Motility Map.** Pixels are coloured according to node speed, as calculated by ECMM. Red shades represent expanding regions, blue shades contracting regions. Pixel values within the *tiff* image are scaled to fill the colour spectrum. The map file extended *_motilityMap.maPQ* contains un-scaled values, in *microns per second.*
- **Fluorescence Maps**. A fluorescence map is produced for each channel that data has been recorded into. The fluorescence map images display pixel intensities, as extracted by the ANA plugin. Files extended *_fluoCh$.maPQ* contain the respective data.
- **Convexity Map**. Represents the curvature of the cell, measured in the range $[-1, 1)$, where negative values are concave (blue), and positive convex (red). As with the motility map, values are scaled to fill the colour map. Files extended *_convexityMap.maPQ* contain the respective data (un-scaled).
- **Co-ordinate Map (a.k.a. Position Map)**. As described in section 5.1, each node has an associated position. The co-ordinate map, rather than contain values regarding motility, fluorescence or convexity, instead contains the position values of nodes. The main purpose of the co-ordinate map, along with the origin map, is for tracking positions through time (see section 9.3).

- **Origin Map**. As described in section 5.1, each node has an origin, the position a node originated from on the previous frame. The origin map contains origin values and can be used, along with the co-ordinate map, to track positions through time (see section 9.3).
- **xMap**. Contains horizontal image pixel co-ordinates (those on the image used for segmentation) relating to map pixels.
- **yMap**. Contains vertical image pixel co-ordinates (those on the image used for segmentation) relating to map pixels.

8.5 Scalable vector graphics output - svg files

Files extended with *.svg* can be viewed natively in most internet browsers (Windows Explorer requires the Adobe SVG viewer), or opened and edited in graphics software (e.g. Inkscape, free software for mac). The Scalable Vector Graphics (svg) file format can be viewed at any desired resolution.

- **Cell track**. An overlay of all cell outlines coloured according to frame number. Files extension *_track.svg.*
- **Cell motility movie**. Data contained within the motility/fluorescence/convexity map is displayed on the cell track. Extension *_motility.svg.*

9 MATLAB Functions

QuimP11 outputs a large amount of data for each cell analysed. You may want to write custom scripts for analysis, but we provide a set of MATLAB functions to remove the hassle of loading/organising data, and performing simple operations, such as plotting maps. We hope to make data as accessible and as explorable as possible to allow adaptation to a users specific goals.

Help documentation is included with each function which can be accessed in the usual way by typing *help functionName* at the MATLAB command prompt. At the end of this document is a walkthrough analysis using the provided test images (see section 10).

9.1 Loading data - *readQanalysis.m*

The first step is to launch the script *readQanalysis.m* which handles loading of data from multiple cells. Given a directory, it will search that directory, and all sub-directories, for *.paQP* files. For each parameter file that is found an analysis structure is built. As data is read from sub-directories, you may organise your separate analyses into sub-directories, but placing all files into the same directory is still possible. Missing files are skipped, hence it is not necessary to have run all the QuimP11 plugins.

Loading data relies on the maintenance of file names imposed by the QuimP11 plugins, and requires associated files to be present in the same directory as the parameter file. The exceptions to this are the images used for segmentation and fluorescence measurements. Paths to these images are stored in full to allow separation of analysis files from image data. If images are moved, and subsequently cannot be located, the directory where the parameter file is stored will be searched. You may edit the *.paQP* file to alter paths to the images if you wish (note that images are not loaded into memory by *readQanalysis.m*, instead the function checks only that they exist and sets a path to them).

The output of *readQanalysis.m* is an array of structures, C, each element holding data for a single analysed cell. The contents of an analysis structure is as follows (F = number of frames, N^f = number of nodes at frame f):

- **name** [*string*] - File name element common to all associated files (name of the parameter file).
- **index** [*integer*] - Cell ID, in order read in.
- **PATH** [*string*] - Directory of the parameter file.
- **$FILE** [*string*] - Filenames of QuimP output.
- **$TIFF** [*string*] - Filenames of an image sequences.
- **outlines** [*cell array* $(F \times 1)$] - Holds node associated data. Each element contains a matrix of size $N^f \times 6$, each row being a node. Column headers are [Position , X-coord , Y-coord , Origin , Global origin , Speed].
- **outlineHeaders** [*cell array* (1×6)] - Descriptions of column measures within *outlines*, as stated above.
- **fluo** [*cell array* $(F \times 1)$] - Holds node fluorescence data for all three channels. Each element contains a 3D matrix of size $N^f \times 3 \times 3$. The first dimension represents nodes, second the channel number, and third the measure. The measures are [Intensity , X-coord , Y-coord], where the x and y co-ordinates provide the location of sampling (see section 6). For example, the intensity sampled at node 23, on channel 2 is located at (23,2,1).
- **fluoHeaders** [*cell array* (1×3)] - Descriptions of measures within *fluo*, as stated above.
- **nbFrames** [*integer*] - The number of frames.
- **startFrame** [*integer*] - The first frame segmented relative to the the image used for segmentation.
- **endFrame** [*integer*] - The last frame segmented relative to the the image used for segmentation.
- **frames** [*vector* $(F \times 1)$] - Vector of the segmented frames relative to the the image used for segmentation.
- **PixelSize** [*double*] - Pixel size in microns (image scale).
- **frameInterval** [*double*] - Frame interval in seconds.
- **R** [*vector* (1×4)] - Pixel co-ordinates of a bounding box encompassing all movement of the cell. R = [x min, x max, y min, y max].

- **maxSpeed** [*vector* $(F \times 1)$] - For each frame the maximum node speed is calculated. This is useful for scaling plots.
- **stats** [*matrix* $(F \times 11)$] - Holds whole cell statistics relating to the cell centroid and cell shape (see section 8.3). Rows contain data for frames. The column headers are ['Frame' 'x-Centroid' 'Y-Centroid' 'Displacement' 'Distance Travelled' 'Directionality' 'Speed' 'Perimeter' 'Elongation' 'Circularity' 'Area']
- **statsHeaders** [*cell array* (1×11)] - Descriptions of column measure within *stats*, as stated above.
- **fluoStats** [*matrix* $(F \times 3 \times 11)$] - Holds global cell statistics relating to the cell fluorescence (as described in section 8.3). The first dimension is frame number, second the channel number, and third the measure. The measure headers are ['Frame' 'Total Fluo.' 'Mean Fluo.' 'Cortex Width' 'Cyto. Area' 'Total Cyto. Fluo.' 'Mean Cyto. Fluo.' 'Cortex Area' 'Total Cortex Fluo.' 'Mean Cortex Fluo.' '%age Cortex Fluo.']. Missing data is represented by negative values, typically -1.
- **fluoStatHeaders** [*cell array* (1×3)] - Descriptions of the measures in the third dimensional within *fluo*, as stated above.
- **cortexWidth** [*vector* (3×1)] - Holds for each channel the cortex width specified when running ANA.
- $*$**Map** [*matrix* $(F \times MapRes)$] - Maps read from *.maQP* files (see section 8.4). A QuimP map in MATLAB is a 2D matrix, rows being frames, columns being discrete points along the cell outline.
- **forwardMap** - see section 9.3.
- **backwardMap** - see section 9.3.

9.2 Function Overview

Functions prefixed with ‘read’ are used by *readQanalysis.m* and so generally are not used independently, but can be if desired. For details of usage please refer to the functions help by typing *help functionName* at the MATLAB command prompt.

- **buildTrackMaps** - Constructs the *forwardMap* and *backwardMap* based upon the co-ordinate and origin maps. These are used to track through QuimP11 maps. See section 9.3 for details.
- **mapLookup** - Extract values from maps by specifying a window or region. This can be used, for example, to extract data along tracked paths.
- **plotMap** - Plot a QuimP11 map using specified colour map limits.
- **plotMotility** - Plots a cell outline with nodes coloured according to their speed of movement.
- **plotOutline** - Plots a cell outline.

- **trackBackwards** - Utilises the *backwardMap* to track backwards through a map, from a specified point, in accordance with the ECMM tracking data.
- **trackForwards** - Utilises the *forwardMap* to track forwards through a map, from a specified point, in accordance with the ECMM tracking data.
- **trackForwAcc** - Similar to *trackForwards* but functions at sub-pixel resolution. More accurate but slower.
- **xcorrQ** - Perform cross-correlation and auto-correlation of .maPQ maps.

9.3 Tracking Maps

The term *tracking* in this sense refers to being able to pick a position on the perimeter of a cell and identify its corresponding position in the following frames, and hence, track a position over time. The trace of a position, computed by ECMM, can be plotted on top of a QuimP map.

A QuimP map in MATLAB is a 2D matrix, rows being frames (f) and columns being discrete locations along the cell outline (l). An incorrect view may be that a map pixel p at frame f, and location l, (p_l^f), tracks to the pixel directly below (p_l^{f+1}). This is not the case, p_l^{f+1} does not necessarily originate from p_l^f. The origin map must be consulted to identify the correct location.

[Maps that enforce p_l^f tracking to p_l^{f+1} become heavily distorted, and certain regions lose resolution severely.]

To track from p_l^f to p_{l2}^{f+1} we require the correct matrix column index at $f+1$, $l2$. Firstly, consult the co-ordinate map at p_l^f to identify the position on the cell outline. Next, extract the values in row $f+1$ from the origin map. The index $l2$ is simply the index of the closest origin value to p_l^f's position (this can also be done at sub-pixel resolution for more accuracy). The process is repeated to find p_{l3}^{f+2}, p_{l4}^{f+3}, etc. In a similar way, positions can be tracked backwards in time.

This tracking procedure is implemented in the provided MATLAB functions. The function *buildTrackMaps.m* creates two additional maps, *forwardMap* and *backwardMap*, and is called when data is read in. An element in the *forwardMap*, for example at p_l^f, contains the correct column index for $l2$ to locate p_{l2}^{f+1}. Similarly, *backwardMap* contains the correct values for finding p^{f-1}.

Functions *trackForwards.m*, *trackForwAcc.m* and *trackBackwards.m*, given a frame, location, and number of frames to track over, will return complete traces for you. Consult the help for these functions for details of usage and the walkthrough for an example.

10 Walkthrough analysis

The folder *QuimP11_walkthrough_files* contains 3 tiff image sequences which you can analyse (images are courtesy of Evgeny Zatulovskiy, Rob Kay Group, MRC Laboratory of Molecular Biology).

- 'QW_channel_1_actin.tif'- Actin label. Image and has been background corrected and contrast enhanced.

- ‘QW_channel_2_neg.tif’ - Negative stain. The cell appears as a shadow on a bright background, which has then been inverted.
- ‘QW_channel_2_seg.tif’ - For segmentation we shall use the negative stain channel. A 1 pixel Gaussian blur has been applied, the background removed, and contrast enhanced.

Step 1. Open ImageJ and launch the QuimP bar [$Plugins \rightarrow QuimP11 \rightarrow QuimPBar$]. Open the image ‘QW_channel_2_seg.tif’. Launch BOA from the QuimP11 bar.

Step 2. At the prompt check the scale is correct (2 second frame interval, pixel width 0.2 microns).

Step 3. Using the polygon selection tool, create a selection encompassing the cell. This can be very rough. Click *Add cell.*

Step 4. Adjust the parameters to get a good segmentation. Click *SEGMENT* and wait for completion.

Step 5. Scroll through the sequence to check the segmentation result using either the scroll bar or mouse wheel. The segmentation should have completed to the last frame. If not, we will truncate the segmentation. Scroll to the last frame which successfully segmented and click *Truncate Seg.* Click *FINISH*, and provide a name for the analysis (e.g. ‘QTest’).

Step 6. Launch the ECMM plugin from the QuimP11 bar. It does not require an image. When prompted, locate the QuimP parameter file you just created (e.g QTest.paQP). ECMM will run. You can view the result and close the image.

Step 7. Open the image ’QW_channel_1_actin.tif’. With it in focus, launch the ANA plugin, and again select your parameter file. Choose a sensible cortex width (e.g. 1.5 microns). Make sure ‘save in channel’ is set to 1, and normalise to interior is ticked. Click *ok.* When complete, ANA will show the sample locations for the last frame.

Step 8. Close ’QW_channel_1_actin.tif’, and open ‘QW_channel_2_neg.tif’. We will record another channel for good measure. Repeat the last step, but with the channel 2 image, this time storing data in channel 2 (which is the default), and tick ’Sample at Ch1 locations’. ANA will now use the same sample points as computed for channel 1 (useful if you want to compute ratios between channels).

Step 9. Launch the Q Analysis plugin, and again choose your parameter file. Check your scale is what you expect at the top of the parameter window. We will use all the current defaults, so click *RUN.* Inspect your maps, all of which are automatically saved to disk.

Step 10. The displayed images have been scaled to cover the colour space. The raw values have been saved separately as text files, with the extension *.maQP* (e.g. QTest_0_fluoCh1.maQP). You can view them in ImageJ by opening them via [$File- > Import- > TextImage$...].

Step 11 We will now switch to MATLAB to plot some of this data. Open MATALB and open the script file ’walkthrough.m’, which will guide you through loading and plotting data. You can use the output provided in the QuimP package, called *QWalkthrough*, rather than your own.

11 Contact

The QuimP11 software is being developed at the Systems Biology DTC at the University Of Warwick by the Till Bretschneider group (Till.Bretschneider@warwick.ac.uk). Further information, and future releases, can be obtained from `http://go.warwick.ac.uk/quimp`.

QuimP11 is under continuous development and feedback will be much appreciated. If you discover a bug, have suggestions for features, or simply find a spelling mistake, please contact the developer Richard Tyson at richard.ryson@warwick.ac.uk (`http://www2.warwick.ac.uk/fac/sci/systemsbiology/staff/tyson/`). Thank you for supporting our software.