**BETSE GitHub Instructions**

The ability to successfully run pipeline requires Matlab, Anaconda, WEKA, and CellProfiler; The repository is comprised of two main sections:

* Quantitative Image Analysis and Pattern Recognition
* Parameter Optimization

The data and code pertaining to the ‘Quantitative Image Analysis and Pattern Recognition’ section is stored in the ‘../QuantitativeImageAnalysis/’ sub-directory and the data and code pertaining to ‘Parameter Optimization’ is stored in the ‘../temp\_synthesis/’ sub-directory. Instructions to run the code are located in the ‘BETSE-repo/BETSE GitHub Instructions.docx’ file.

My version (below):

**[section 1] Deriving the condition-specific pattern classifiers**

1. To analyze the raw BETSE images, go to the ‘../QuantitativeImageAnalysis/TuringPatterns’ sub-directory and open the ‘dataSetBETSEPatternNewcut’ script in Matlab.

2. The ‘arffFile’ variable will ultimately output the directory and file for the output QTS comparison file (line \_); name it is desired.

3. Populate the ‘positiveimagefolder1’ variable (line 60) with the desired positive image set; this will be what the supervised machine learning algorithm utilizes to distinguish ‘pattern (P)’ and ‘non-pattern (NP)’ input images for the ability to distinguish unique spatial features of ‘pattern (P)’ images.

4. Populate the rest of the non-pattern image folders into ‘negativeimagefolder1-5’ variable(s) into lines 88, 116, etc. as desired.

5. Run the script; the arffFile generated will be used to determine the formal language pattern classifier in WEKA/RIPPER.

6. Open WEKA, select the ‘Explorer’ tab from the start menu, then select the ‘Preprocess’ option. Find directory containing output .arff file that was produced in the prior step and select the .arff file (this file contains the QTS data for each input ‘Pattern’ or ‘Non-pattern’ image).

7. Now select the ‘Classify’ tab in the WEKA Explorer and choose the ‘JRip’ classifier, which utilizes the ‘RIPPER’ supervised machine learning algorithm to learn a formula for your input image sets.

8. Export the text under the ‘JRIP rules:’ section to a .txt file. This file will serve as the defining pattern classifier rule set for each desired culture/environmental condition, and further serves as the input ‘target pattern’ in the Particle Swarm Optimization parameter optimization pipeline.

Graphical user interface, text, application, email

Description automatically generated

9. Open ‘getFormula.m’ script and name the ‘fileName’ variable (line 6) the name of the .txt file generated in step8.

10. Run the script and save the output ‘ans’ variable as a .mat file, named as desired.

In vitro Image directories to analyze:

**[section 2] Generating the Robustness/similarity score for patterns of interest**

1. Go to the ‘../QuantitativeImageAnalysis/TuringPatterns/’ sub-directory and open the ‘findRobustness2.m’ Matlab script.

2. Set ‘imageFolder’ variable to input in vitro image set to be compared against the target pattern classifier. The in vitro set directories used in this study are shown below.

* ‘Control’ image set 🡪 ‘../QuantitativeImageAnalysis/2021 in vitro outcomes/Ctrlall\_3’;
* ‘-GA 10 uM’ image set 🡪 ‘../QuantitativeImageAnalysis/2021 in vitro outcomes/bGA10all\_1’;
* ‘-GA 60 uM’ image set 🡪 ‘../QuantitativeImageAnalysis/2021 in vitro outcomes/bGA60all\_1’;
* ‘K+ 10 mM’ image set 🡪 ‘../QuantitativeImageAnalysis/2021 in vitro outcomes/K10all\_1’;
* ‘K+ 20 mM’ image set 🡪 ‘../QuantitativeImageAnalysis/2021 in vitro outcomes/K20all\_1’;
* ‘LBC2/LBC2-GJA1’ image set 🡪 ‘../QuantitativeImageAnalysis/2021 in vitro outcomes/LBC2GJA1set\_2’;

3. Edit ‘outputFile’ variable to desired output directory and filename (line 18).

4. Select the .mat file containing the parsed pattern classifier formula from section 1, step 10.

5. Run the script.

6. Find the generated output file, and run statistical analyses in Excel to determine image set similarity score mean, standard error on the mean, etc.

**[section 3] Running Particle Swarm Optimization**

1. In the ‘ParameterOptimization.m’ file, set desired pattern classifier .txt file to the target pattern condition you are looking to recreate with your optimized parameters.

2. Define numerical bounds for parameter space searching (line 25).

3. Run ‘qsub bioescript109.pbs’ to start the process.

\*\*\*IFFFF you want to sim one parameter you must make the following changes to files A-D in the \_ sub-directory:\*\*\*

*(A) BetseSetup2.py*

A1. Comment out the input arguments code under the ‘read\_args’ object definition to exclude whatever additional parameter that is not being optimized; for example, if the Dmem,K+ parameter is desired comment out lines 44-49.

A2. Edit the input arguments for the ‘configure\_yaml\_model’ object for the ‘output\_folder’ variable; first, modify the syntax code from ‘ "%s\_%s\_%s\_%s\_%s" ’ to ‘ "%s\_%s\_%s" ’; next, delete inputs to the ‘output\_folder’ variable that correspond to the undesired parameter.

A3. Edit the input arguments for ‘change\_yaml’ object definition (lines 98-109) to disregard undesired 2nd parameter configuration file edits; only uncomment: line 103 if only Dmem,K+ desired; line 104 if only Dmem,Na+ desired, line 105 if only Dmem,Cl- desired, and line 106 if only Dmem,GJ desired.

*(B) ParameterOptimization.m*

B1. Edit the ‘parameters’ variable to 1, instead of 2.

B2. edit the ‘bounds’ variable to (a) only include 1 bounded numerical value interval, and (b) include the correct physiologically-relevant boundary values to prevent the simulation from unstable simulation dynamics (boundaries shown below).

🡪 Dmem,K+ bounds: 1e-19 to 1e-16 m2/s

🡪 Dmem,Na+ bounds: 1e-19 to 1e-17 m2/s

🡪 Dmem,Cl- bounds: 1e-19 to 1e-17 m2/s

🡪 Dmem,GJ bounds: 1e-9 to 1e-7

*(C) Particle\_Swarm\_Optimization\_Parallel.m*

Graphical user interface, text, application

Description automatically generated

C1. In ‘Number\_of\_quality\_in\_Bird’ section (lines 190-199), change ‘value1’ and ‘value2’ to desired parameter(s) to optimize and repeat for lines 231-240; a sample edited ‘Number\_of\_quality\_in\_Bird’ section is shown in lines 242-249.

C2. Repeat this procedure for the second finalized ‘Number\_of\_quality\_in\_Bird’ section (lines 231-240), but replacing ‘gBest(i)’ with ‘optimised\_parameters(i)’.

*(D) runBetse.m*

D1. Uncomment the ‘strcat’ command for loop (lines 19-22); reselect desired parameter name that reflects the simulated parameter being optimized if needed.

D2. Comment the ‘folderName{ii}’ variable for two parameters (lines 26-29).

D3. Uncomment the ‘folderName{ii}’ variable for one parameter (lines 38-40); reselect desired parameter name that reflects the simulated parameter being optimized if needed (types 1-4).

D4. Comment the ‘folderName{ii}’ variable for two parameters (lines 42-44).