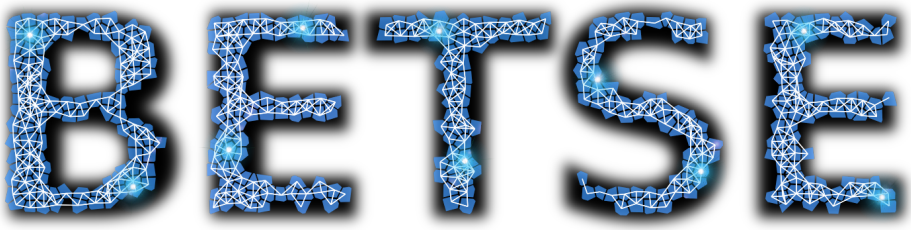
BETSE 1.0

April 10, 2019



Bio**E**lectric **T**issue **S**imulation **E**ngine

Command Line Interface Documentation (Version 1.0)

Alexis Pietak

Contents

1 Preamble

This **B**io**E**lectric **T**issue **S**imulation **E**ngine (BETSE) facilitates bio-realistic modeling of dynamic electrochemical phenomena in gap junction networked cell collectives.

BETSE is intended to help test hypotheses relating to experimentally-accessible electrochemical phenomena important in morphogenesis, regeneration, and disease processes such as cancer. A special emphasis is placed on how electrochemical phenomena can lead to spatio-temporal pattern formation.

2 Getting Started

2.1 Installation

BETSE and additional resources are available from the online repository:

https://gitlab.com/betse/betse

BETSE is installable under Linux, OS X, and Windows, as follows:

1. Windows users only: emulate Ubuntu Linux via the Windows Subsystem for Linux (WSL).

   https://docs.microsoft.com/en-us/windows/wsl/install-win10

Note that the Windows Subsystem for Linux (WSL) and – hence BETSE itself – is only installable under Windows 10. Under older Windows versions, BETSE may be installed from a virtual Linux guest.

2. Install the Python 3.x (e.g., 3.6) variant of Anaconda:

   https://www.anaconda.com/distribution/

Note: **do not** install the Python 2.7 variant of Anaconda. BETSE requires Python 3.x.

If you prefer **not** to install Anaconda, BETSE dependencies are also manually installable via your platform-specific package manager (e.g., Homebrew on macOS, APT on Ubuntu Linux). Doing so is non-trivial and, where performed incorrectly, could produce a performance-crippled single-core installation of BETSE. Anaconda suffers no such issues and is guaranteed to produce a performance-optimized multicore installation of BETSE on all supported platforms. Anaconda is strongly recommend.

3. Open a Bash terminal. To open a POSIX-compatible terminal on your operating system:

→ Under Windows, install Ubuntu Linux via the Windows Subsystem for Linux (WSL). Open an Ubuntu Linux terminal.

→ Under macOS, open the *Finder*. Open the *Applications* folder. Open the *Utilities* folder. Open *Terminal.app*.

→ Under Ubuntu Linux, type Ctrl+Alt+t, or find and open *Terminal* in the *Applications* browser under *Utilities*.

4. Enable conda-forge at the command line:

```
conda config --add channels conda-forge
```

5. Install BETSE at the command line:

```
conda install betse
```

This command installs both the most recent stable release of BETSE and all mandatory and most optional dependencies of this release. Older stable releases are installable in a similar manner (e.g., conda install betse=0.7.0 for BETSE 0.7.0). All Anaconda packages are kindly hosted by the non-profit conda-forge organization.

6. Test the BETSE install at the command line:

```
betse -v try
```

This test command enables verbosity with the -v option, simplifying issue reporting in the event of an unexpected error. Creates a sample_sim/ subdirectory in the current directory, providing the default simulation for this release of BETSE. This includes all configuration files and resources referenced by these files. Runs all simulation phases (e.g., seed, initialization) of this simulation.

When finished, you may safely either remove the subdirectory *sample_sim* created by betse try, or rename the subdirectory (e.g., to my_sim/) to serve as a basis for subsequent simulations. Alternatively, you may preserve the subdirectory as is.

2.2 System Requirements

Supported operating systems:

→ Mac OS X 10.6 (Snow Leopard) or newer.

→ Linux distributions providing at least *glibc* 2.19 or newer. This includes, but is not limited to, the following Linux distributions: Linux Mint 17.1 (Rebecca) or newer. Ubuntu 14.10 (Utopic Unicorn) or newer. BETSE currently runs only on 64-bit systems of Linux. This is principally due to the increasing obsolescence of 32-bit systems, particularly for scientific work.

→ Microsoft Windows XP or newer

We currently recommend your system have at least 4 GB RAM for BETSE simulations.

2.3 Tutorial Materials

BETSE makes a number of sample simulations, including configuration files and all materials required to run the simulation, as well as a Developer's Tutorial, which explains how BETSE models and modules can be worked with from an external Python script. These materials and their documentation are available from:

https://gitlab.com/betse/betse#tutorials

3 What's New in BETSE 1.0?

3.1 New Features for BETSE 1.0

→ New "fast" and "full" BETSE solver types. BETSE can now work from an equivalent circuit model formalism (e.g. Hodgkin-Huxley based model of bioelectricity) using the new "fast" solver, or from its original molecular view of bioelectricity using the "full" solver type. This allows users to tailor their computational perspective to the needs of their biophysical problem. Details are discussed in Section 6.

→ Expression profile read-in. BETSE can now read in expression levels from a user-defined expression configuration file for any custom-defined biomolecule, transporter, or channel. This allows different levels of a biomolecule/transporter or channel to be specified on all of the user-defined tissue profiles of a BETSE model, where expression data can be sourced from experiments or experimental databases. See Section 15, 15.1 and 15.9 for details.

→ Single cell simulation. BETSE can now simulate a single cell, a feature requested for fast prototyping of certain simulations. See Section 10.1 for details.

→ SVG model definition. In addition to the original bitmap method of defining cell cluster and tissue profiles, BETSE also allows use of Scalable Vector Graphics (SVG) files to define exact locations of cells in your cluster, where colors can be used to specify a tissue profile for each individual cell. See Section 10.1 for details.

→ Mesh refinement. Optionally run a refinement algorithm on BETSE meshes to generate more computationally robust model geometries. See Section 10.1 for details.

→ Modulator functions. Use a customizable modulator function (e.g. gradient along the x-axis) to modulate expression of custom biomolecules, transporters or pumps. See Section 13 for details.

3.2 New Features from BETSE 0.4

→ **B**ioelectricity-**I**ntegrated **G**ene and **R**eaction (BIGR) networks: define your own regulatory networks with customized chemical substances that activate/inhibit each other's native growth rates, chemical reactions between substances, transmembrane transporters, and ion channel and gap junction regulation.

→ Wide range of biorealistic dynamic ion channels sourced from experimentally determined Channelpedia models.

→ Deformations of the cell cluster using global electric fields as a force and assuming a galvanotactic mechanism.

→ Modulation functions (e.g. spatial gradients of substance production or membrane gating) to provide an initial symmetry breaking in a simulation.

→ Improved simulation speed with lower memory requirements

→ Automated testing of new features (for developers)

→ Online code repository (for developers)

→ Memory and speed profiling (for developers)

3.3 New Features from BETSE 0.3

BETSE 0.3 includes minor new features:

→ Restructuring of the config file to assist in proper synchronization between the cell cluster (seed), initialization (init), and simulation (sim) runs as well as more comprehensive error reporting.

→ A "Goldman Equation Calculator" to provide a comparison between BETSE's Vmem output and the Goldman Equation for each cell and time step (see section 16.7)

→ Optional implementation of osmotic pressure into simulation with potential induction of fluid flow through gap junction connected cells.

→ A range of new plot types.

3.4 New Features from BETSE 0.2

BETSE 0.2 included the four major new features:

1. Optional use of bitmap images (*.png) to control cell cluster shape and to define tissue profiles within the cluster (Figure 1).

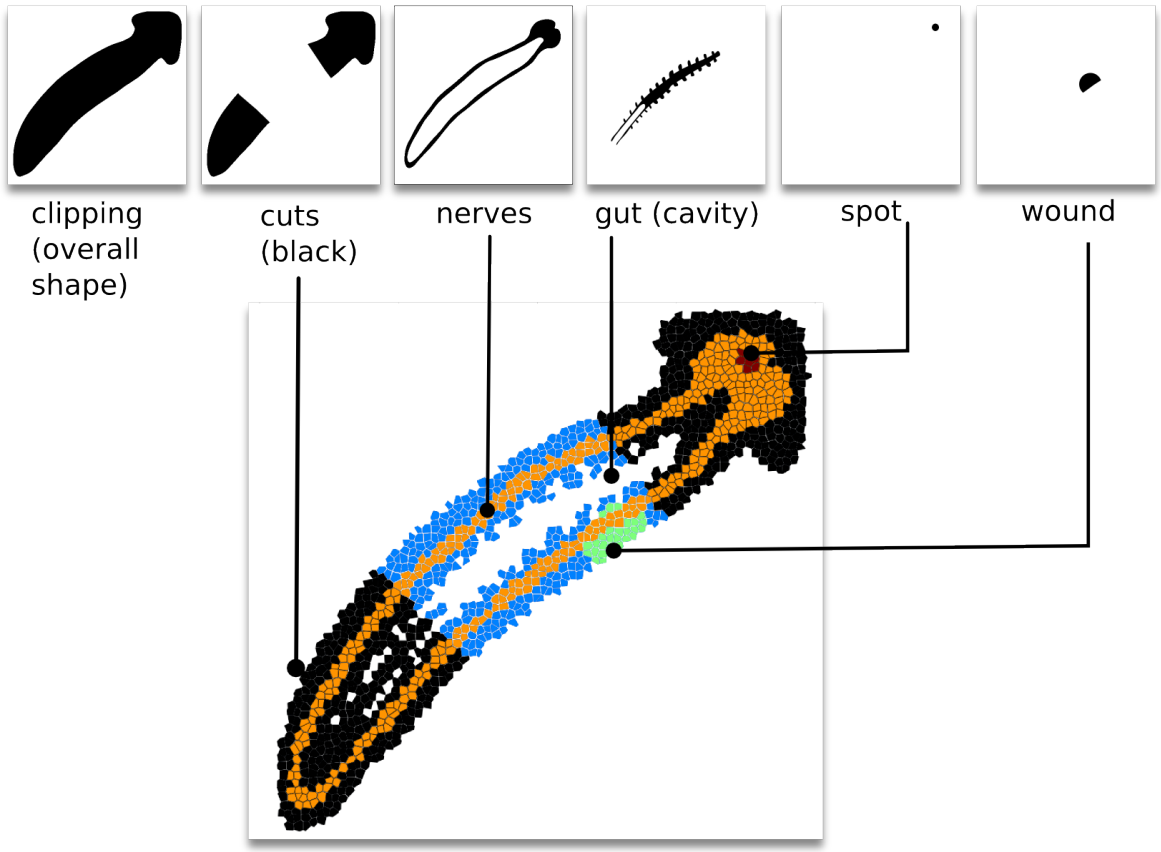


Figure 1: Bitmap images (top row) can now be used to control overall shape of the cell cluster (clipping) as well as to cut a hole into the model upon initialization (cavity), to define cuts for the model at the onset of a simulation (cuts), and to define tissue profiles (nerves, spot, wound) where different scheduled interventions and dynamic channels may be assigned.

2. Ability to define tissue profiles as spatial regions within the cluster having optional unique base membrane properties, isolated network connectivity, and dynamic elements such as gated ion channels.

3. Optional ability to simulate the extracellular spaces and environment surrounding the cell culture, with calculation of variables including: voltage, electric fields, currents, and electroosmotic flow.

4. Upgraded model to full Finite Volume Method solver for increased accuracy and stability.

4 Basic Commands

4.1 Your First Simulation

The following describes how BETSE can be run as a command-line-interface (CLI) program, meaning it is run by issuing written commands at the Terminal. A graphical user interface (GUI), BETSEE (BioElectric Tissue Simulation Engine Environment) is under development and can be trialed at:

https://gitlab.com/betse/betsee

Program settings are specified by a *configuration file* (*config file*), which is a special kind of text file that you can edit with a program such as *Gedit* (Linux), *Textwrangler* (Mac), or *Notepad* (Windows). We will get into thorough descriptions of the config file settings in subsequent sections of this documentation, but for now let's go through some CLI basics.

To get to the Terminal (command line) on **Mac**: open *Finder*, select *Applications*, choose *Utilities* and double click on *Terminal*. On **Linux**, simply open the *Menu*, select *Accessories*, and choose the *Terminal*. On **Windows**, first install Ubuntu Linux via the Windows Subsystem for Linux (WSL), and then open an Ubuntu Linux terminal.

Please note, all of the following commands will be structured for a Linux and Mac operating system and may need slight adjustment for Windows. The most common difference is that Linux and Mac systems use "/" to specify directory structure (e.g. "Documents/Sims/my_file.png") while Windows uses the "\" (e.g. "Documents\Sims\my_file.png"), so if you're working on Windows the following commands will need to be adjusted accordingly.

Once in the terminal, change the directory to one where you'd like to save your BETSE simulations and results. For example, let's assume that under your 'Home' directory, in 'Documents', you've made a new folder called 'BETSE_Sims' . To get BETSE to save output in this folder, in the command line type:

```
cd ~/Applications/BETSE_Sims
```

This will set the working directory to "BETSE_Sims".

To see if BETSE works at all, once you've changed the directory, run BETSE from that directory by simply typing:

```
betse
```

A help message should appear, which goes over the top-level commands for BETSE (we will cover these in the following).

In order to test that the program works on your system and get a hassle-free look at what a basic simulation looks like, next type:

```
betse try
```

This 'try' command does everything necessary to run default initialization and simulation runs, and saves the resulting simulation files and visualized results on your computer in a new folder called 'sample_sim', appearing wherever you have copied and run the BETSE executable.

Once the initializations and simulations are running, you should see a continuously updating figure appear, which is plotting the V_{mem} for the cell collective as the simulation is computing. Once this has completed, messages will appear in the terminal reporting the progress of the simulation.

After running the 'betse try' command, if you go back to the directory where you placed and ran the BETSE executable, you should see a new directory called 'sample_sim' inside it. Inside of this 'sample_sim' directory, you should find a default configuration file called 'sample_sim.yaml', as well as an 'init' directory. Inside the 'init' directory is a 'results_a' directory, where you should see the saved 'plot while solving' animation as a set of png images.

The default simulation run using the 'betse try' command studies a 100 μm cluster of cells (approximately 66 cells) on an irregular hexagonal lattice. An initialization is run for 1 second, which is followed by a simulation run for 1 seconds. At the beginning of the simulation run, cells are removed from the bottom right region of the cluster by a wounding event.

If at any time you wish to prematurely end a BETSE event, press the keys ***Ctrl c*** to cancel (sometimes ***Ctrl c*** must be pressed repeatedly until the event cancels).

4.2 What to do when BETSE Crashes

When BETSE crashes, please send the config file you were attempting to use, as well as the log file for the crash to: alexis.pietak@gmail.com.

When BETSE crashes it will give you a message directing you to where it writes BETSE log files on your computer.

It is important to send both the config and log file to facilitate debugging.

You are also welcome to post issues (and make feature requests) on BETSE's gitlab code repository:

https://gitlab.com/betse/betse/issues

4.3 The BETSE Command Line Interface

The CLI features 10 main commands that you can issue BETSE at the Terminal:

1. try *seed, init, and sim a sample tissue simulation*
2. info *display instructive information about BETSE and the current system*
3. config *create a new tissue simulation configuration file that also supplies folders with default bitmaps and sample network config files*
4. seed *make the cell cluster from parameters specified in a configuration file of your choice*
5. init *initialize a previously created cell cluster with initialization-specific parameters specified in a configuration file of your choice*
6. sim *run the simulation for an initialized cluster using simulation-specific parameters specified in a configuration file of your choice*
7. plot *plot previously created cell clusters, initialization or simulation runs (uses sub-commands)*
8. sim-grn run a gene regulatory network (GRN) without bioelectrical simulation (for speed and comparison)
9. repl drop into a “read-eval-print-loop” interactive coding environment (to manipulate data your own way)

To determine if a BETSE executable is active in a directory and see a high-level help message, simply type:

```
betse
```

To check if BETSE will simulate properly on your computer and obtain a straightforwards taste of how BETSE works, simply type the ’try’ command:

```
betse try
```

To determine where various files (such as log files) live on your system, and to obtain all sorts of meta-data information about BETSE, type the ’info’ command:

```
betse info
```

When you start running simulations in BETSE, it's easiest to make a new directory on your system (e.g. "Documents/BETSE_Sims"). Switch into this directory:

```
cd ~/Documents/BETSE_Sims
```

Then create a default configuration file in that directory by using BETSE's ***config*** command along with the name of the new configuration file you'd like created:

```
betse config test_1.yaml
```

This ***config*** command will create a file called 'test_1.yaml' in the directory "Home/Documents/BETSE_Sims". Of course, you can specify whatever directory and filename you wish for the config file, but we'll remain consistent with the location and filename of a configuration file at the "~/Documents/BETSE_Sims/test_1.yaml" path for this example.

To change settings of the cell cluster, initialization, and simulation, you'll need to locate the 'test_1.yaml' on your system, open it in a text editor, and make the desired changes. Please see the remainder of this documentation for information on how to change features of the config file.

To continue to run a simulation, the cell cluster and (optionally) environmental spaces grids and other basic information must be set up for computation. To do this, while still in the directory that you want to save simulation in, and with the outcome determined by settings in the configuration file that is specified, use the BETSE ***seed*** command:

```
betse seed test_1.yaml
```

After a "seed" has been successfully run, BETSE will write a file to the relative directory specified under 'init file saving', 'worldfile' of your config file (in this case the test_1.yaml file). This is referred to as the *world file* and has the suffix *.betse.

To run an initialization using a config file of your choice and a cell cluster previously seeded, use the ***init*** command, specifying the filename of the config file controlling the scenario you want to run:

```
betse init test_1.yaml
```

After an 'init' has been successfully run, BETSE will write a file to the relative directory specified under 'init file saving' of your config file. This is referred to as the *init file* and has the suffix *.betse. The purpose of the init is to generate a cluster that is at steady state.

As such, none of the dynamic features, such as user-specified interventions, will be active during the init.

After creating the cell cluster and running the initialization, use the ***sim*** command, specifying filename of the config file controlling the scenario you want to simulate. If a world or initialization file aren't found under the path and filename listed under 'Initialization Settings' section of the config file, BETSE will automatically seed a cell cluster and/or do an initialization before getting to the simulation run:

```
betse sim test_1.yaml
```

After a 'sim' has been successfully run, BETSE will write a file to the relative directory specified under 'sim file saving' of your config file. This is referred to as the sim file and has the suffix *.betse. If you are using a compression algorithm, the suffix may have an additional term indicating the type of compression (e.g. *.betse.gz)

After creating a cell cluster (seed), running an initialization (init), or completing a full simulation (sim), you can load, plot, and save the stored data from the world, init and sim files that BETSE created. It's possible to change settings in the *Results* section of the configuration file to alter the appearance and type of plots that are created with the plot command. To plot a previously created cell clusters, initializations, or simulations use the 'plot' command with one of three subcommands:

To load and plot the cell cluster:

```
betse plot seed test_1.yaml
```

To load and plot an existing initialization:

```
betse plot init test_1.yaml
```

And to load and plot a previously run simulation:

```
betse plot sim test_1.yaml
```

The config file tells the computer where to save all files, and these same paths in the config file tell the computer where to find files to load and plot. Plotting results are saved to folders specified under "results file saving" of the config file.

Help messages at the Terminal can be accessed for commands by adding '-h' or '–help' after the command name, for example:

```
betse sim --help
```

Note, at any time if you wish to abort an initialization or simulation, at the Terminal simply press **ctrl c** (on Mac you may need to press **ctrl c** many times to get it to kill the program).

Additional commands

The command:

```
betse sim-grn test_1.yaml
```

Lets you test-drive a user-defined gene regulatory network (GRN) using an additional configuration file specified under the "Gene Regulatory Network" banner of the main config file, without involving bioelectricity (see Section 15 for details). This is intended for speed, and to compare the output to the traditional non-bioelectricity influenced scenario for a particular GRN.

The results of the GRN simulation can be plotted using:

```
betse plot sim-grn test_1.yaml
```

The command:

```
betse repl
```

Allows you to use an interactive code environment at the command line (similar to matlab), which enables you to work with BETSE data structures.

5 The Configuration File

The configuration file allows you to have control over what happens during BETSE cell cluster creation, initializations, simulations, and results export and visualization.

In order to change settings and variables in BETSE, you simply open up a configuration file in a text editor and set the values to what you wish. You then run the BETSE program with that configuration file's path and name specified at the command line (see Section 4).

5.1 How to Make Default Config Files

To create a default configuration file that you can edit further, you have two command-line choices:

After switching into the directory where you want to save BETSE output, for example:

```
cd ~/Documents/BETSE_output
```

you could run the ***try*** command:

```
betse try
```

...in which case you'll find a 'sample_sim.yaml' file inside a folder called 'sample_sim' in the directory where you're running BETSE (e.g. "Documents/BETSE_output/sample_sim/sample_sim.yaml"). You can change the name of the .yaml file and edit it as you wish, but the config filename should always end with ".yaml".

Or, you could run simply the ***config*** command:

```
betse config my_new_config.yaml
```

...in which case you'll find a 'my_config_file.yaml' file created inside a folder called 'bio_sims' in directory where you're running BETSE (e.g. "Documents/BETSE_output/my_new_config.yaml"). You can obviously change the directory and filename to whatever you wish, but the config filename should end with ".yaml".

You can also just copy, paste, and rename any BETSE config file into a directory of your choice using the graphical system navigator of your operating system (e.g. *Nautilus* on Linux or *Finder* on Mac).

5.2 Basic Structure of Config Files

Each BETSE config file has 20 sections:

1. **Solver Settings** *(specify the type of BETSE simulator to run)*
2. **File Handling** *(specify paths for saving/loading cell cluster, initialization runs, simulation runs, and results output)*

3. **Initialization Settings** *(settings specific to the initialization run)*
4. **Simulation Settings** *(settings specific to the simulation run)*
5. **General Options** *(specify main options like what ions you want to include – these parameters are fixed to the initialization run and propagated through subsequent simulations loading that initialization and can only be modified at the time of running betse* **seed***.)*
6. **World Options** *(specify aspects of the cell, cluster, and gap junction network – these parameters are fixed to the initialization run and propagated through subsequent simulations loading that initialization; they can only be modified at the time of running betse* **seed***.)*
7. **Tissue Profile Designation** *(tell BETSE what bitmaps define the cluster and tissue regions, specify properties of tissue profiles, specify global boundary profiles)*
8. **Targeted Interventions** *(schedule events, such as changing membrane permeabilities, that apply to specify tissue profiles)*
9. **Modulator Function Properties** *(edit properties of functions that can be used to modulate biomaterial, transporter, or channel expression levels)*
10. **Global Interventions** *(schedule events applying to the whole environment or cluster, such as the addition or pharmacological agents or blocking gap junctions)*
11. **General Network** *(define a biomolecular regulatory network in the main configuration file; this section turns it on/off and allows you to run an optimization)*
12. **Custom Biomolecules** (define your own chemical substances that appear in your regulatory network)
13. **Custom Chemical Reactions** (define chemical reactions between substances of your regulatory network, which may have rates that are co-regulated by substances of your regulatory network)
14. **Custom Channels** (define your own ion channels, which may adopt voltage gating properties and be co-regulated by substances of your regulatory network)
15. **Custom Transporters** (define your own transmembrane transporters, which may be co-regulated by substances of your regulatory network and be accompanied by a chemical reaction occurring only in the cytoplasm)
16. **Custom Modulators** (allow substances of your general regulatory network to modulate the behavior of core BETSE entities, such as gap junctions)
17. **Gene Regulatory Network** *(run a user-defined regulatory network that is read from a separate config file specified under this main config file banner)*
18. **Variable Settings** (general and world options that can be changed between seed, init and sim runs without causing synchronization issues)

19. **Results** *(specify what plots, animations and exports to include, as well as their features)*
20. **Internal Use Only** *(change variables relating to the inner-workings of the simulation)*

5.3 Keeping BETSE Output Files Synchronized

BETSE creates three kinds of program-related output files: 1) a world file created after the ***seed*** command is successfully run, which stores data needed to set up the cell cluster and its environment; 2) an init file created after the ***init*** command is successfully run, which stores all the data relevant to the initialization run on the cell cluster; and 3) a sim file created when the ***sim*** command is successfully run, which stores all of the data relating to the simulation run output.

A potential trouble spot in BETSE exists because *only* some of the parameters in the config file can be changed without destroying important relationships between the seed, init and sim files. For example, when settings under the *Tissue Profile Definition*, *Targeted Interventions*, *Dynamic Ion Channels*, *Global Interventions*, *Variable Settings*, *Results*, and *Internal Use Only* sections of the config file are changed when trying to run a sim after already running a seed and an init, the sim can update the settings without issue and incorporate them into the simulation or plotting request. There are no concerns about changing parameter values for these sections of the config file.

However, if settings under *General Options* and *World Options* are changed after running a seed or an init, it's necessary to run a fresh ***seed*** and a fresh ***init*** before moving on to do the ***sim***. This is because the settings under these sections define key data architectures, and if the config file changes, it breaks the synchronization between BETSE's three output file types. To ensure that these important sections are synchronized, BETSE will throw an error and tell you to go run a fresh seed and init if it detects desynchronization between parameters in a config file you're trying to run and the original config file used to do the seed.

The ability to change parameters is listed in the config file under the header of each section as a "File sync warning". For reference, here is a summary of the config file sections listing whether parameters must remain fixed after running a seed and/or an init (headings in boldface type), or if parameters in these sections can be changed at any time:

1. Solver Settings *(can be changed at any point, though instability may result)*
2. File Handling *(can be carefully changed at any point as long as the correct load files and directories are maintained)*
3. Initialization Settings *(can be changed at any point; relevant only when running an init)*
4. Simulation Settings *(can be changed at any point; relevant only when running an init)*

5. ***General Options*** *(can be specified only when running a seed and must stay fixed after that or a new seed run if changed)*
6. **World Options** *(can be specified only when running a seed and must stay fixed after that or a new seed run if changed)*
7. Tissue Profile Designation *(can be changed at any time)*
8. Targeted Interventions *(can be changed at any point; relevant only when running a sim)*
9. Modulator Function Properties *(can be changed at any point)*
10. Global Interventions *(can be changed at any point; relevant only when running a sim)*
11. General Network *(can be changed at any time)*
12. Custom Biomolecules *(can be changed at any time)*
13. Custom Chemical Reactions *(can be changed at any time)*
14. Custom Channels *(can be changed at any time)*
15. Custom Transporters *(can be changed at any time)*
16. Custom Modulators *(can be changed at any time)*
17. Gene Regulatory Network *(can be changed at any time)*
18. Variable Settings (*can be changed at any point*)
19. Results *(can be changed at any point; relevant only when plotting)*
20. Internal Use Only *(can be changed at any point)*

5.4 Config Files and YAML

The configuration file is written in a computer language called *YAML*, which is fairly easy for both humans and the Python BETSE program to read. However, because the configuration file is a little computer script, when editing it you have to be careful not change things that will make it non-readable by the BETSE program. For example, it is essential that the line indentations (tabs) are maintained in their current order, and that only variable assignments are changed. Variable names must not be changed. In the file, comments intended only for human users, and that are not read by the BETSE program, are prefixed by a "#" and may be edited as you wish. Likewise, feel free to add in new # prefixed comment lines to any BETSE config file.

For more information on the *YAML* language, please see:

https://en.wikipedia.org/wiki/YAML

```
# ------------------------------------------------------------------------------
# SOLVER SETTINGS
# ------------------------------------------------------------------------------
# Settings specific to simulation solvers (i.e., numerical techniques internally employed by
# BETSE to iteratively compute each time step of this simulation).

solver options:
  type: full # Type of solver to enable for this simulation, as any following string:
             # * "full", a complete but slower solver producing publication-quality results
             #   by comprehensively simulating all possible bioelectrical phenomena.
             # * "fast", an incomplete but faster solver producing draft-quality results
             #   by the well-known equivalent circuit formalism analogizing biological
             #   systems to electronic circuits. While integrated with gene regulatory
             #   networks (GRNs), this solver *CANNOT* simulate:
             #   * Bioelectric fields or currents.
             #   * Extracellular voltages or voltage polarities.
             #   * Ion concentrations.
             #   These phenomena are silently ignored when this solver is enabled.

```

Figure 2: The *Solver Settings* section of a BETSE configuration file.

6 Solver Settings

BETSE 1.0 includes the option to use entirely different computational methods to simulate bioelectricity, which can be employed using the 'type' field of the new 'solver options' panel of the main config (Figure 2). The "full" solver type simulates bioelectricity from a molecular perspective using the GHK-flux equation to describe transmembrane ion fluxes and updates concentrations inside (and optionally outside) of cells. The "fast" solver uses the equivalent circuit model of bioelectricity (e.g. Hodgkin-Huxley formalism) to calculate ion fluxes using reversal potentials for each ion, and does not update concentrations inside and out of cells.

7 File Handling

7.1 Specifying File Save/Load Directories and Filenames

Figure 3 shows the *File Handling* section of the configuration file, where you can specify the directory and file names for the init (init file saving), cell cluster (worldfile), sim (sim file saving) and any automatically saved and exported results (results file saving). Initializations and simulations in BETSE automatically save all of their data as a '.betse' binary file that

```
# ------------------------------------------------------------------------------
# FILE HANDLING
# ------------------------------------------------------------------------------
# Settings specific to saving simulations and results output.

init file saving:                # Initialization paths to save and load.
  directory: INITS               # Directory containing the following files.
  worldfile: world_1.betse.gz    # File with cell cluster created by "betse seed".
                                 # Supported filetypes include (ordered by decreasing filesize):
                                 # ".betse" (no compression),
                                 # ".betse.gz" (very fast minimal compression),
                                 # ".betse.bz2" (slow medium compression), and
                                 # ".betse.xz" (fast maximal compression).
  file: init_1.betse.gz          # File of initialization results created by "betse init".
                                 # Supported filetypes are as above.

sim file saving:                 # Simulation paths to save and load.
  directory: SIMS                # Directory containing the following file.
  file: sim_1.betse.gz           # File of simulation results created by "betse sim".
                                 # Supported filetypes are as above.

results file saving:             # Initialization and simulation exports to save.
  init directory: RESULTS/init_1 # Directory of initialization exports created by
                                 # "betse plot init" (e.g., plots, animations).
  sim directory: RESULTS/sim_1   # Directory of simulation exports created by
                                 # "betse plot sim" (e.g., plots, animations).

```

Figure 3: The *File Handling* section of a BETSE configuration file.

can be re-read and plotted afterwards. These files can be optionally compressed in size by adding a ".gz", ".bz2", or ".x2' suffix to the file save/load name (see Figure 3).

There is also the option to automatically save all the plots, animations, and raw data exports for a simulation when a plot command is run. The places where these auto-saves (and corresponding auto-loads) happen are also defined in the BETSE configuration file. Figure 3 shows the "results file saving" tab, where exported results will be saved to the init and sim directories for init and sim phases, respectively.

For portability, files and directories are configured as paths that are *relative* to the location of the config file, rather than as *absolute* pathnames. For example, instead of defining the save/load path of the simulation as:

"/Documents/bio_sims/my_sim/sim_52.betse"

If the config file is located within a folder called "bio_sims" , then under "sim file saving" all you need to specify is a directory name of "my_sim" and a file of: "sim_init.betse" to achieve the desired result of saving sim_52.betse under the above directory structure. Note, if "my_sim" folder does not exist, BETSE will automatically create this for you.

This usage of relative paths allows any folders containing your simulations and results to be safely moved to other locations or computers, without breaking the functionality of existing simulations.

7.2 Using a Simulation as an Initialization

The difference between an init and sim is that an init does not implement any of the global or targeted interventions or dynamic ion channels, yet sometimes it's desirable to set the scene of a simulation with dynamics present. While the typical course of events is to do an initialization (init) first, which saves a *.betse binary to the init directory and filename, it's possible to seed a world, do an init, do a sim (sim a), and to then load the sim as the initialization to another sim (sim b). To do this, after running the first sim (sim a), simply use the sim file created as an initialization by specifying the path and filename of sim a in the "init file saving" fields of the config file, and supply a new directory and filename to save the upcoming sim (sim b) in the "sim file saving" fields. Make sure you move the sim_a betse file to the right place.

8 Init and Sim Settings

```
77 # ------------------------------------------------------------------------------
78 # INITIALIZATION SETTINGS
79 # ------------------------------------------------------------------------------
80 # Settings specific to the initialization.
81
82 automatically run initialization: True      # If init file is not found, do you wish to
83                                               # automatically run an init?
84
85 init time settings:
86   time step: 1.0e-2            # time step-size [s] (recommended at least 1.0e-2 or smaller)
87   total time: 5.0              # end time [s]
88   sampling rate: 1.0           # time interval to sample data [s] (at least value of time-step or larger)
89
90 # ------------------------------------------------------------------------------
91 # SIMULATION SETTINGS
92 # ------------------------------------------------------------------------------
93
94 sim time settings:
95   time step: 1.0e-4         # time step-size [s] (recommended 1.0e-2 or smaller)
96   total time: 0.035         # time to end sim run [s]
97   sampling rate: 1.0e-3     # period to sample data [s] (at least time step or larger)
98
```

Figure 4: The *Initialization Settings and Simulation Settings* sections of a BETSE configuration file.

Each simulation that is run in BETSE requires an initialization run. The initialization run creates the cell cluster and performs a simulation without any of the features listed in the *Global Interventions, Targeted Interventions* or *Dynamic Channels* sections being active, which allows the modeled bioelectric system to be in, or closer to, cell resting $V_{mem}$and steady-state. Whenever you try to run a simulation in BETSE using the betse ***sim*** command, it will check to see if you have a *.betse initialization file saved to the directory and filename specified under the “init file saving” region of the config file. If not, it will go ahead and run an initialization first (unless you change the *Simulation Settings* field ’automatically run initialization’ to False).

Otherwise, you can also run an initialization without running a sim using the ***init*** command (preferred):

```
betse init my_config_file.yaml
```

After running an init, run the sim phase:

```
betse sim my_config_file.yaml
```

The *Initialization Settings* of the config file allow you to specify settings for the initialization run, likewise, the *Simulation Settings* portion allows you to do the same thing for the sim phase.

8.1 Init and Sim Time Profile

The time-step features for the initialization are found under the “init time settings” heading, and for simulation under “sim time settings” (see Figure 4) The first setting is the “time step”, which is the amount by which the simulation increases world-time in each calculation loop. Smaller time steps will give you more stability and more accuracy, at the expense of increased simulation time and memory as each time step represents a computational loop. Simulation step-size is defined in seconds [s].

The recommended range for “time step” is 1e-2 s or smaller for init runs that don’t use voltage gated ion channels, and 1e-4 s for simulations that use voltage gated ion channels.

Next, use the “total time” field to decide how long you want to run the initialization, defined in seconds [s]. The recommended range is from 0.1 to 300 s, but it depends on your time-step. The “sampling rate” allows you to specify a time period in seconds [s] to re-sample data from the init/sim run, so that you are not sampling data at each time step. The recommended re-sampling range is from 1.0e-3 to 10 s, but depends on how large your time-step is, and how long your simulation runs for.

```
init time settings:
    time step:  1e-2
    total time:  20
    sampling rate:  0.1
```

9 General Options

```
# ------------------------------------------------------------------------------------------
# GENERAL OPTIONS
# ------------------------------------------------------------------------------------------
# Specify general options to be used in the init and sim runs.
# File sync warning: you can't change any of these general options after running 'betse seed'.

general options:
  comp grid size: 25                    # grid used in computation of environmental
                                        # parameters (min 10; keep smaller than 50)
  simulate extracellular spaces: True   # include extracellular spaces and true environment
                                        # simulation?

  ion profile: basic    # ion profile options:
                        # 'basic' (Na+, K+, M-, and proteins-)
                        # 'basic_Ca' (Na+, K+, Ca2+, M- and proteins-)
                        # 'mammal' (Na+, K+, Cl-, Ca2+, M- and proteins-) typical mammal
                        # 'amphibian' (Na+, K+, Cl-, Ca2+, M- and proteins-) Xenopus
                        # 'custom' (user-specified settings, see below)
                        # (all profiles contain an unidentified charge-balance anion, M-)

  customized ion profile:       # set user-defined initial values of ion concentrations
                                # for the initialization run
                                # note both extracellular and cytosolic levels of all
                                # ions must sum to zero charge.
    extracellular Na+ concentration: 145.0          # extracellular sodium [mmol/L]
    extracellular K+ concentration: 5.0             # extracellular potassium [mmol/L]
    extracellular Cl- concentration: 115.0          # extracellular chloride [mmol/L]
    extracellular Ca2+ concentration: 2.0           # extracellular calcium [mmol/L]
    extracellular protein- concentration: 10.0      # extracellular protein [mmol/L]

    cytosolic Na+ concentration: 12.0               # intracellular sodium [mmol/L]
    cytosolic K+ concentration: 139.00              # intracellular potassium [mmol/L]
    cytosolic Cl- concentration: 4.0                # intracellular chloride [mmol/L]
    cytosolic Ca2+ concentration: 2.0e-5            # intracellular calcium [mmol/L]
    cytosolic protein- concentration: 135.0         # intracellular protein [mmol/L]

```

Figure 5: The *General Options* section of a BETSE configuration file.

The *General Options* section of the configuration file is where you specify the kinds of ions you wish to include in the simulation and their initial cell and environmental concentrations,

as well as whether or not you want to run a full environmental simulation with extracellular spaces.

9.1 Grid

BETSE creates a square grid that is used to interpolate cell-specific data such as ionic currents to prepare streamline plots, as well as to compute properties in environmental spaces (when this option is implemented). You can specify the resolution of the square grid (as the number of ticks of the horizontal and vertical axes) using the 'grid size' option under the *General Options* section:

```
grid size: 40
```

Grid sizes are recommended to not be lower than 10 and to not exceed 60.

9.2 Simulate a Full Environment

A single line of the *General Options* section controls the major feature of whether the cell cluster, initialization and simulations model a full environment with extracellular spaces around each cell and optional tight and adherens cell-cell junction settings:

```
simulate extracellular spaces: True
```

If 'simulate extracellular spaces' is 'False', the cell cluster will be created without environmental spaces, and initializations and simulations will assume cells flux out to a global environment with instantaneous mixing to the larger whole.

9.3 Ion Profiles

The 'ion profile' field is used to specify the type of ions to be used in the simulation and their initial values inside and out of cells (with all cells having homogeneous values for ions inside and out). The values specified in the ion profile are the initial condition values for ion concentrations inside and out of the cell at the beginning of the initialization simulation:

```
ion profile:  basic
```

To make things a bit easier, there are three pre-made default profiles that you can choose for this field:

→ 'basic' *(Na^+, K^+, negatively charged proteins P^-, and a charge balance anion M^-)*

→ 'basic_Ca' *(Na^+, K^+, Ca^{2+}, negatively charged proteins P^-, and a charge balance anion M^-)*

→ 'mammal' *(Na^+, K^+, Cl^-, Ca^{2+}, negatively charged proteins P^-, and a charge balance anion M^-)*

Note as ion number increases, time to run an initialization or simulation increases proportionally (*i.e.* it will take about 1.75 times longer to run a simulation with the 'mammal' profile compared to the 'basic' profile), and memory demands increase proportionally.

In addition to these four default profiles, you can specify 'custom' for the 'ion profile' field, which allows to you use your own settings. If the 'custom' profile is selected, initial ion levels in the cell and environment must be defined under the 'customized ion profile' header at the end of the *General Options* section (described in sub-section 9.4).

9.4 Customized Ion Settings

If 'customized' is specified in the 'ion profile' setting, you will include all of the major ions in your simulation by default (Na^+, K^+, Cl^-, Ca^{2+}, and anionic proteins/macromolecules P^-, and a charge compensation anion M^-) and you must specify all of the initial values for these ions in the cytosolic and environmental space in the config file.

The units of all concentrations are [mmol/L] or equivalently [mol/m^3]. The values specified in the ion profile are the initial condition values at the beginning of the initialization simulation.

Note that if the "Internal Use Only" parameter (see Section 18) is set to:

```
substances affect Vmem:  True
```

Then charged "custom molecules" from any BIGR network you are running will be included as concentrations in BETSE's assessment of initial charge.

When using BETSE's customized ion settings, BETSE will automatically balance the net charge inside and outside of cells to **zero** as an initial condition. In order to accomplish this, BETSE defines a value of the charge compensation anion, M^-, inside and out of the cell. As BETSE uses an anion to charge compensate, the net charges defined in the '*custom ion profile*' panel, as well as all initial concentrations of charged molecules in your BIGR regulatory networks, must sum to a net positive charge.

```
customized ion profile:
    extracellular Na+ concentration:  145.0
    extracellular K+ concentration:  5.0
    extracellular Ca2+ concentration:  1.0
    extracellular protein- concentration:  10.0
    cytosolic Na+ concentration:  8.0
    cytosolic K+ concentration:  125.0
    cytosolic Cl- concentration:  20.0
    cytosolic Ca2+ concentration:  1.0e-4
    cytosolic protein- concentration:  100.0
```

10 World Options

```
# ------------------------------------------------------------------------------------------
# WORLD OPTIONS
# ------------------------------------------------------------------------------------------
# The 'world' refers to cell geometric properties, the size of the cluster, and the properties of
# the environment, such as size and temperature. These variables are common to both init and sim
# runs. All of these world variables will be created in the init and apply to all sims running from
# the init.
#
# SVG files should have a square page area (e.g. 500x500 mm) and be prepared in Inkscape. The
# centres of cells are defined by 'circle' objects in the 'cells from svg' file. The fill color of
# the circular cell centerpoints can be used to assign cells to a specific tissue profile using the
# hex "color" type in the "Tissue Profiles" section below. When preparing the cell svg file, do not:
#
# * Use transforms (e.g. flip, rotate).
# * Group cells.
#
# File sync warning: you can't change any of these 'world options' after running 'betse seed'.

world options:

  world size: 150e-6     # dimension of the square world space [m]
                           # the world is square with x = y.
                           # recommended range 80e-6 to 1000e-6 m

  cell radius: 5.0e-6      # radius of single cell [m]
                           # recommended ~ 5.0e-6

  cell height: 10.0e-6      # the height of a cell in the z-direction [m]
                           # used in cell volume and surface area calculations
                           # recommended 10.0e-6

  cell spacing: 26.0e-9        # the spacing between cells [m]
                             # recommended 26.0e-9 m for animal cells

  simulate single cell: False  # ignore all geometry and simulate a single hexagonal cell

```

Figure 6: Initial portion of the *World Options* section of a BETSE configuration file.

In BETSE, the 'world' refers to cell geometric properties, the size of the cluster, and the properties of the environment such as size and temperature. These variables are common

to both initialization and simulation runs.

To ensure synchronization between initialization and simulation runs, world options will always default to those of the initialization. In other words, you are not able to change the geometry of cells or the cluster when calling 'betse sim'; only when calling 'betse init'. If world variables are altered after the initialization, they will not be implemented; rather, BETSE will default to whatever world variables existed in the initialization loaded from the Initialization Settings path.

10.1 Cell and Cluster Geometric Features

Currently, the world is square (x=y) and the cell cluster based on a circular shape. Specify the dimension of the world space using the 'world size' field, with the value in meters [m]. The recommended range for world dimension is from 100e-6 to 1000e-6 m.

```
world size:  200e-6
```

Next specify the radius of a single cell in meters [m]. The recommended range is from 3.5e-6 to 20e-6 m.

```
cell radius:  5e-6
```

Use the 'cell height' field to specify the height of a cell in the z-direction [m]. Although BETSE is a 2D simulator, cell height is used to calculate volume and lateral surface area. The recommended range is from 1.0e-6 to 10e-6 m.

```
cell height:  5.0e-6
```

Specify the spacing between cells [m]. The recommended value is 26.0e-9 m for animal cells.

```
cell spacing:  26.0e-9
```

Ignore all tissue profiles and mesh building features of BETSE to simulate a single cell:

```
simulate single cell:  True
```

```
171
172 lattice type: hex            # Type of base cell lattice (i.e., uniform grid to
173                               # which cells are situated before random lattice
174                               # disorder is applied), as any following string:
175                               # * "hex" for hexagonal cells.
176                               # * "square" for square cells.
177
178 lattice disorder: 0.4        # noise level for the lattice
179                               # recommended range 0 to 0.8
180                               # this randomizes lattice points from the rectangular or
181                               # hexagonal base grid. If equal to 0, a perfect
182                               # hexagonal or rectangular grid is created.
183
184 mesh refinement:              # Use Llyod's algorithm to optimize the Voronoi mesh?
185   refine mesh: True           # Turn optimization on? (Only works for Convex model shapes)
186   maximum steps: 10           # Maximum number of itterations)
187   convergence threshold: 1.5  # Threshhold below which optimization is considered complete
188
189 import from svg:  # Import individual cell centres and clipping curve from an svg file
190   svg override: False  # Turn to True to enable imports from SVG files
191   cells from svg: geo/root/root_cells.svg # read in exact cell centre locations from 'circles' in svg file
192   svg size: 500  # size of the svg page area in mm
193
194 alpha shape: 0.01  # alpha shape threshhold (used for working with non-convex shapes) 0.01 to 0.9
195 use centers: False  # use cell centroids instead of circumcentres when building meshes?
196
```

Figure 7: Subsequent portion of the *World Options* section of a BETSE configuration file.

The cell center points can initially be distributed on a hexagonal or rectangular lattice. Specify which one you want using 'hex' for hexagonal and 'rect' for rectangular:

```
lattice type:  hex
```

Specify the noise level for the lattice using the 'lattice disorder' field. This parameter deviates cell center points from a square grid with random perturbations (see Figure 8). If 'lattice disorder' is equal to 0, a perfect square grid arrangement of cells is created; higher values generate irregular (and arguably more bio-similar) grids. The recommended range is from 0 to 0.6.

```
lattice disorder:  0.4
```

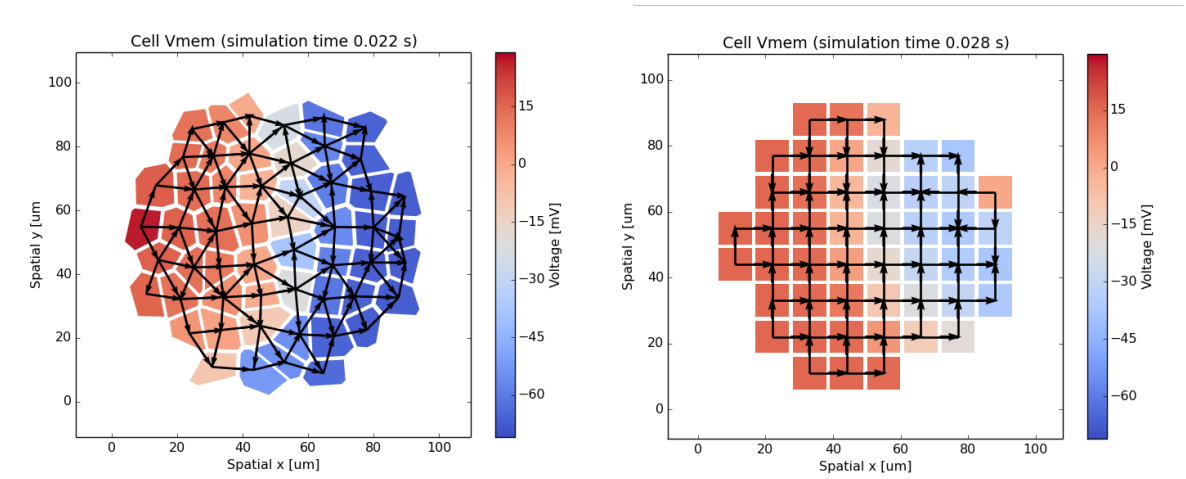


Figure 8: Lattice disorder of 0.8 (left image) and 0.0 (right image).

Mesh refinement

BETSE can now use Lloyds algorithm to refine the Voronoi mesh defining cells of the cluster. This refinement algorithm makes the centroids and circumcenters of the Delaunay triangulation more equivalent, which leads to more accurate discrete math computations. Set the maximum number of times you want the refinement algorithm to run using “maximum steps” and the convergence threshold as required for your model.

```
Refine mesh:  True
maximum steps: 10
convergence threshold: 1.5
```

Specify the threshold for alpha-shape definition of the potentially non-convex boundary of your cell cluster in a range between 0.0 and 1.0 (higher values of alpha shape are required for non-convex model boundaries):

```
alpha shape:  0.2
```

Use the geometric centroid instead of the circumcenters of triangular simplexes of the mesh to calculate Voronoi cell vertices (try using centers if your mesh is messy):

```
use centers:  False
```

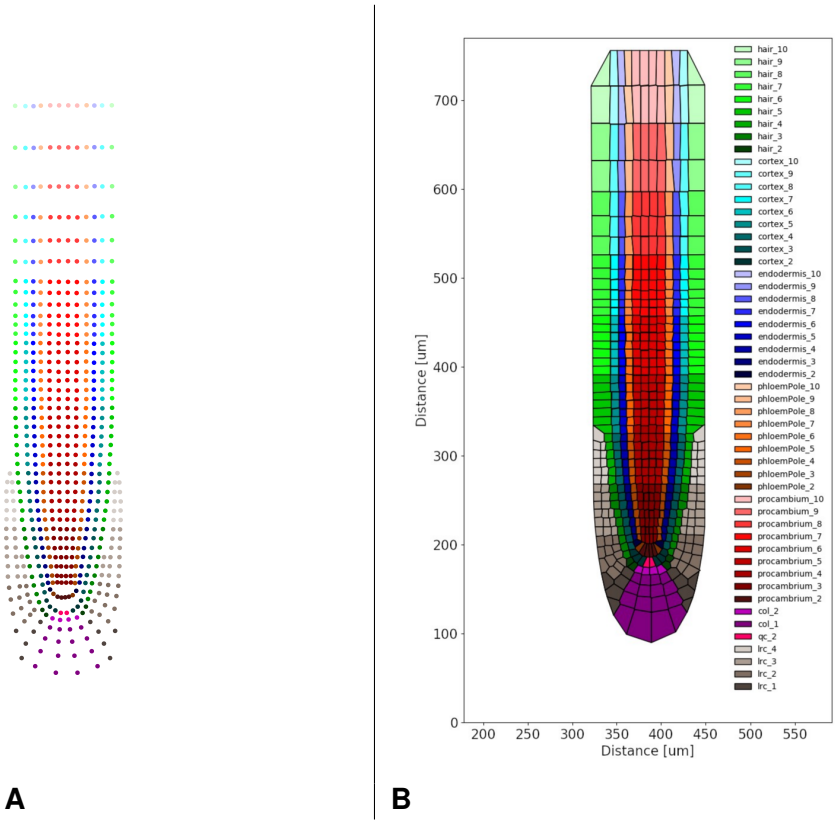


Figure 9: An SVG file can be used to define the BETSE cell cluster and tissue profiles. Each svg circle defines the center of a Voronoi cell of the resulting mesh. Panel A shows an example SVG circle layout, while B shows the resulting BETSE model. The hex color code of each SVG circle fill can be used to assign the cell to a specific tissue profile.

Import from SVG

In addition to defining cluster shape and tissue profiles using bitmaps (see Section 11.1), BETSE now allows for optional use of Scalable Vector Graphics (SVG) images to define exact locations of cells, and assign cells to tissue profiles. The SVG model file must use SVG circles (note they must be circles and not ellipses) to define the location of a cell center in the resulting BETSE mesh. The hexidecimal color value of the fill color of the SVG circle can be used to assign cells with that color to a specific tissue profile (see Section 11.2). The SVG file used to define a model is shown in Figure 9A, while the BETSE-computed cellularized model with cells assigned to color-coded tissue profiles is shown in Figure 9B.

Note that the SVG file used to define the BETSE model must not contain any 'transformations' on the circle objects (i.e. no rotations, translations, etc), as these will not be applied by BETSE and result in circles/cells being located in the wrong place. In the BETSE default simulation package, the file geo/root/root_cells_demo.svg is a working example for a suitable SVG format to define a BETSE model.

11 Tissue Profile Designation

The *Tissue Profile Designation* section of BETSE config files is where you define specific geometric aspects of your cell cluster. BETSE currently supports three different types of geometric profiles: a “clipping” profile that defines the overall shape of the cell cluster, a “tissue” profile that defines different properties on regions within the main cell cluster, and a “wound” profile that allows you to specify a region that will be removed from the cell cluster in a cutting event. BETSE allows you to specify bitmap images (i.e. *.pngs) that BETSE will read in to define your tissue profiles.

11.1 Using Bitmap Images to Define Profiles

BETSE can read in an unlimited number of bitmap images that can be used to define the overall shape of your cell cluster, as well as to define spatial regions within the model that have different properties and dynamics (tissue profiles), or regions which are cut out of the model before an initialization or during a simulation.

These bitmap image sets can be created in any drawing program that supports the export of *.png images.

Note that when you create a default configuration file using the “betse config ~/path_to_my_config/config_file.ya
command, a folder called “geo” is also created. Geo includes different sub-folders with default bitmap image sets suitable for use with BETSE.

Bitmap images you create yourself must have the following features:

- → square images with equal number of pixels on each side (recommended 500 x 500 pixels per image)
- → pure white backgrounds (not transparent!)
- → area to define cluster or tissue region must be a solid black area
- → *.png file format
- → saved with file names and .png extension

Figure 11 shows an example of four bitmap images that were used to define a cell cluster and three tissue profiles in Figure 12.

```
198 #-------------------------------------------------------------------------------------------
199 # TISSUE PROFILE DESIGNATION
200 #-------------------------------------------------------------------------------------------
201 # Tissue profiles are user-defined regions within the cluster to which specific base membrane
202 # diffusion profiles, interventions, and individualized dynamics (e.g., voltage gated channels) can
203 # be applied. Tissue profiles can be gap-junction connected or separated (isolated) from other
204 # tissue profiles. Cluster shape and individual tissues can be defined by images, or specified via
205 # various functional regimes. When multiple profiles apply to the same cell, options defined by
206 # later profiles override options defined by earlier profiles.
207 #
208 # Raster image masks defining tissue and cut profiles *MUST*:
209 #
210 # * Be square (i.e., have equal width and height).
211 # * Contain no alpha transparency layer.
212 # * Contain pure-black pixels (i.e., pixels whose RGB color components are all 0) defining
213 #   the interior shape of that profile. All other pixels are ignored.
214 #
215 # No File synch issues: Tissue profiles can be altered after the time of the 'betse seed' command.
216
217 tissue profile definition:
218
219   profiles enabled: True # Apply (rather than ignore) tissue and cut profiles listed below?
220
221   tissue: # Tissue profiles (i.e., regions whose cells all share the same initial conditions).
222     default: # Default tissue profile applied to all cells *NOT* specifically targeted by a
223               # tissue profile defined below. Never ignored, even if "profiles enabled" is False.
224       name: "Base" # Arbitrary unique profile name.
225       image: geo/circle/circle_base.png # Absolute or relative filename of the image whose
226                                         # pure-black pixels define the cell cluster shape.
227       diffusion constants: # Membrane diffusion constants applied to all unprofiled cells.
228         Dm_Na: 2.0e-18      # Na+ membrane diffusion constant [m2/s]
229         Dm_K:  1.0e-18      # K+ membrane diffusion constant [m2/s]
230         Dm_Cl: 1.0e-18      # Cl- membrane diffusion constant [m2/s]
231         Dm_Ca: 1.0e-18      # Ca2+ membrane diffusion constant [m2/s]
232         Dm_M:  1.0e-18      # M- membrane diffusion constant [m2/s]
233         Dm_P:  0.0          # proteins- membrane diffusion constant [m2/s]
234
```

Figure 10: The beginning of the *Tissue Profile Designation* section of the configuration file, where you specify the clipping bitmap that shapes the whole cell cluster, and tell BETSE whether or not further tissue profiles are enabled .

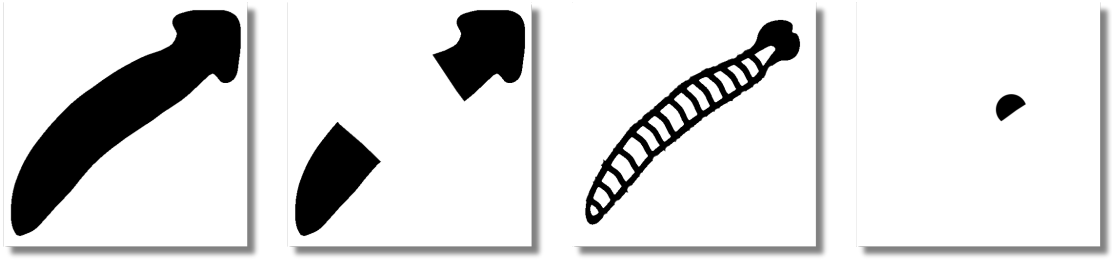


Figure 11: Four example *.png images that can be used to define the overall shape of the cluster (far left image; “clipping” bitmap) and three different tissue profiles within it (“type: tissue” bitmaps).

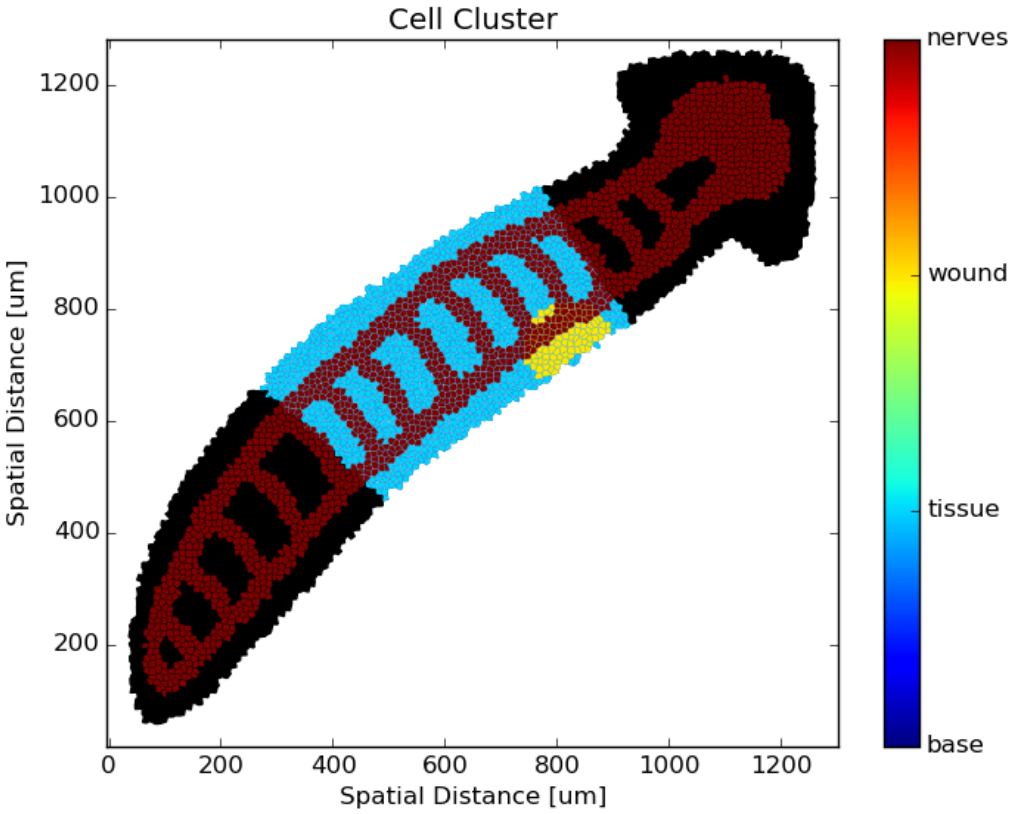


Figure 12: BETSE cell cluster and tissue profiles resulting from bitmap image set shown in Figure 11. Regions of the cluster that are scheduled to be amputated during a simulation (’cuts’) are shown in black.

```
1386      - name: col_1
1387        insular: False
1388        diffusion constants:
1389          Dm_Na: 1.0e-18   # Na+ membrane diffusion constant [m2/s]
1390          Dm_K: 1.0e-18    # K+ membrane diffusion constant [m2/s]
1391          Dm_Cl: 1.0e-18   # Cl- membrane diffusion constant [m2/s]
1392          Dm_Ca: 1.0e-18   # Ca2+ membrane diffusion constant [m2/s]
1393          Dm_M: 1.0e-18    # M- membrane diffusion constant [m2/s]
1394          Dm_P: 0.0        # proteins membrane diffusion constant [m2/s]
1395        cell targets:
1396          type: color    # Cell population type, as any following string:
1397                         # * "all", uniformly applying this tissue profile to all cells.
1398                         # * "image", selecting only cells residing in an image's pure-black area.
1399                         # * "indices", selecting only cells with the indices listed below.
1400                         # * "percent", selecting only a random subset of cells in the cluster.
1401          color: '800080'                 # Color in hexadecimal format of simple circles within this
1402                                          # vector image.
1403          image: geo/circle/spot_3.png # Absolute or relative filename of this image.
1404          indices: [3, 14, 15, 9, 265] # List of all positive integers uniquely identifying each
1405                                       # cell in this cluster to apply this tissue profile to.
1406                                       # Ignored unless "type: indices" specified above.
1407          percent: 50  # Percentage of all cells in this cluster to randomly apply this tissue
1408                       # profile to in the range [0, 100] (e.g., 50 selects 50% of all cells).
1409                       # Ignored unless "type: random" specified above,
1410
```

Figure 13: The hexidecimal fill color value of circles in an SVG model file can be used to assign them to a specific tissue profile. Here SVG circles colored with the hex value '8000080' are assigned to a tissue profile called 'col_1'. This method can be used to specify exact location of cells in a tissue profile. Note that this method of specifying 'cell targets' for a tissue profile is ignored if 'svg override' is not 'True'.

11.2 Using SVG Images to Define Model and Profiles

When using 'svg override' and therefore importing a SVG file to define your BETSE cell cluster, the hexidecimal fill color value of SVG circles defining cell centers can be used to assign cells to a tissue profile. To use the SVG fill color to assign cells to a profile, use 'cell targets' 'type: color' and specify the 6-digit hexidecimal color code for the fill-value of circles in the SVG file representing the cells you want to assign to a particular tissue profile (see Figure 13 and 9).

11.3 The Default Profile

The first section of Tissue Profile Designation (see Figure 10) allows you to define a "clipping" profile that will define the shape of the whole cluster. If you set "profiles enabled" to False, none of the subsequently-defined tissue profiles will be used in your BETSE init or sim.

The default profile allows you to specify a bitmap image to define the overall model shape (note that this is ignored if 'svg override' is 'True'), and to specify default transmembrane diffusion constants for the ions of your simulation.

```
198 #--------------------------------------------------------------------------------
199 # TISSUE PROFILE DESIGNATION
200 #--------------------------------------------------------------------------------
201 # Tissue profiles are user-defined regions within the cluster to which specific base membrane
202 # diffusion profiles, interventions, and individualized dynamics (e.g., voltage gated channels) can
203 # be applied. Tissue profiles can be gap-junction connected or separated (isolated) from other
204 # tissue profiles. Cluster shape and individual tissues can be defined by images, or specified via
205 # various functional regimes. When multiple profiles apply to the same cell, options defined by
206 # later profiles override options defined by earlier profiles.
207 #
208 # Raster image masks defining tissue and cut profiles *MUST*:
209 #
210 # * Be square (i.e., have equal width and height).
211 # * Contain no alpha transparency layer.
212 # * Contain pure-black pixels (i.e., pixels whose RGB color components are all 0) defining
213 #   the interior shape of that profile. All other pixels are ignored.
214 #
215 # No File synch issues: Tissue profiles can be altered after the time of the 'betse seed' command.
216
217 tissue profile definition:
218
219   profiles enabled: True # Apply (rather than ignore) tissue and cut profiles listed below?
220
221   tissue: # Tissue profiles (i.e., regions whose cells all share the same initial conditions).
222     default: # Default tissue profile applied to all cells *NOT* specifically targeted by a
223               # tissue profile defined below. Never ignored, even if "profiles enabled" is False.
224       name: "Base" # Arbitrary unique profile name.
225       image: geo/circle/circle_base.png # Absolute or relative filename of the image whose
226                                         # pure-black pixels define the cell cluster shape.
227       diffusion constants: # Membrane diffusion constants applied to all unprofiled cells.
228         Dm_Na: 2.0e-18      # Na+ membrane diffusion constant [m2/s]
229         Dm_K:  1.0e-18      # K+ membrane diffusion constant [m2/s]
230         Dm_Cl: 1.0e-18      # Cl- membrane diffusion constant [m2/s]
231         Dm_Ca: 1.0e-18      # Ca2+ membrane diffusion constant [m2/s]
232         Dm_M:  1.0e-18      # M- membrane diffusion constant [m2/s]
233         Dm_P:  0.0          # proteins- membrane diffusion constant [m2/s]
234
```

Figure 14: A tissue type block in the configuration file. These blocks can be added or erased to define as many tissue type profiles as you want for your simulation.

11.4 The Tissue Profile Code Blocks

Under the header “profiles”, you can define any number of tissue profile blocks, which are indicated as “- type: tissue” (see Figure 10). These have their own, unique user-defined name (e.g. name: spot), can be gap junction isolated from other tissue profiles (e.g. insular: True), and that will each be able to have different initial membrane diffusion constants for each ion in the simulation (the set of Dm_ values under “diffusion constants”). Tissue profiles are used so that different scheduled user-interventions, ion channel types, etc, can be assigned to specific regions of a cluster (see Sections 12 and 15, which make use of tissue profiles).

To specify the geometry of a tissue profile, change settings under the “cell targets” heading, specifying the “type”. Under “type: ” you may choose “bitmap”, “indices”, or “percent” (and if using ’svg override’ can use ’color’, as mentioned in Section 11.2). If you choose “type: bitmap”, the image file path is supplied under the subsequent “bitmap” header (see Figure 10). If you choose “type: indices”, integers in square brackets correspond to cell indices in your seeded world (see Figure 10). If you choose “type: percent”, the “percent” header specifies the fraction of randomly selected cells to target (e.g. “percent: 50” means 50% of cells will be randomly selected to be included in the tissue profile. Settings for the cell target assignment possibilities that you do not use are ignored, for example, if you choose “type: bitmap”, “indices” and “random” fields are ignored.

A simulation can contain any number of tissue profile code blocks that you require for your initialization and simulation phases. The entire tissue profile code block, which constitutes all indented material following a “- type: tissue” header (see Figure (see Figure 10)) can be deleted if it is no longer required; similarly a tissue profile code block can be copied, pasted, and modified to easily construct a new profile.

11.5 The Cut Profile Block

Another type of profile is the cut profile block, specified by “type: cut” (see Figure 16), which allows you to specify a bitmap that will define an area where cells are removed from the cluster at the beginning of a ***sim*** command phase when using a targeted intervention called a “cutting event” (see Section 12). The cut profile is assigned a name of your choice, and can only be defined using a bitmap.

```
profiles: # List of all non-default tissue profiles. Ignored if "profiles enabled" is False.
  # Example tissue profile for a depolarized patch of cells at the cell cluster centre.
  - name: "Spot"
    insular: False
    diffusion constants:
      Dm_Na: 2.0e-18   # Na+ membrane diffusion constant [m2/s]
      Dm_K:  1.0e-18   # K+ membrane diffusion constant [m2/s]
      Dm_Cl: 1.0e-18   # Cl- membrane diffusion constant [m2/s]
      Dm_Ca: 1.0e-18   # Ca2+ membrane diffusion constant [m2/s]
      Dm_M:  1.0e-18   # M- membrane diffusion constant [m2/s]
      Dm_P:  0.0       # proteins membrane diffusion constant [m2/s]
    cell targets:
      type: image    # Cell population type as any following string:
                     # * "all", uniformly applying this tissue profile to all cells.
                     # * "color", matching only cells whose cell centres are simple circles with
                     #   a fill color (specified by the "color" setting below) of a vector image
                     #   (specified by the "cells from svg" setting above).
                     # * "image", matching only cells whose cell centres are pure-black pixels
                     #   in a raster image (specified by the "image" setting below).
                     # * "indices", matching only cells whose indices are listed below.
                     # * "percent", matching only a random subset of cells.
      color: ff0000                  # Color in hexadecimal format of simple circles within this
                                     # vector image.
      image: geo/circle/spot_3.png # Absolute or relative filename of this raster image.
      indices: [3, 14, 15, 9, 265] # List of all positive integers uniquely identifying each
                                     # cell in this cluster to apply this tissue profile to.
                                     # Ignored unless "type: indices" specified above.
      percent: 50  # Percentage of all cells in this cluster to randomly apply this tissue
                   # profile to in the range [0, 100] (e.g., 50 selects 50% of all cells).
                   # Ignored unless "type: random" specified above.
```

Figure 15: A tissue type block in the configuration file. These blocks can be added or erased to define as many tissue type profiles as you want for your simulation.

```
295
296 cut profiles: # List of all cut profiles (i.e., cell cluster regions to be permanently
297                # removed by a cutting event triggered during the simulation phase).
298                # Ignored if "profiles enabled" is False or cutting event is disabled.
299   # Example cut profile removing a depolarized wedge of cells from a cell cluster boundary.
300   - name: "surgery" # Arbitrary unique profile name.
301     image: geo/circle/wedge.png # Absolute or relative filename of the raster image whose
302                                 # pure-black pixels define the region of cells to remove.
303
```

Figure 16: A tissue type block in the configuration file. These blocks can be added or erased to define as many tissue type profiles as you want for your simulation.

12 Targeted Interventions

```
#-------------------------------------------------------------------------------------------------
# TARGETED INTERVENTIONS
#-------------------------------------------------------------------------------------------------
# Targeted interventions are scheduled changes that can be applied to cells in
# the cluster by linking to a defined tissue profile. Targeted interventions
# only occur during sim runs, not during an init.
#
# No file synch issues: Targeted intervention settings can be altered after seed
# and init.

change Na mem:        # Change the membrane permeability to Na+
    event happens: False         # turn the event on (True) or off (False)
    change start: 1.0              # time to start change [s]
    change finish: 4.0             # time to end change and return to original [s]
    change rate: 1.0               # rate of change [s]
    multiplier: 10.0             # factor to multiply base level
    modulator function: None              # spatial function:'gradient_x','gradient_y', 'gradient_r' or 'None'
    apply to: ["Spot"]          # name(s) of the tissue profile(s) to apply intervention to

change K mem:        # Change the membrane permeability to K+
    event happens: False           # turn the event on (True) or off (False)
    change start: 1.0              # time to start change [s]
    change finish: 2.0             # time to end change and return to original [s]
    change rate: 0.5               # rate of change [s]
    multiplier: 50.0             # factor to multiply base level
    modulator function: None                  # spatial function:'gradient_x','gradient_y', 'gradient_r' or 'None'
    apply to: ["Spot"]      # name(s) of the tissue profile(s) to apply intervention to

change Cl mem:       # Change the membrane permeability to Cl-
    event happens: False           # turn the event on (True) or off (False)
    change start: 1.0              # time to start change [s]
    change finish: 4.0             # time to end change and return to original [s]
    change rate: 0.5               # rate of change [s]
    multiplier: 10               # factor to multiply base level
    modulator function: None              # spatial function:'gradient_x','gradient_y', 'gradient_r' or 'None'
    apply to: ["Spot"]      # name(s) of the tissue profile(s) to apply intervention to
```

Figure 17: The *Targeted Interventions* section of a BETSE configuration file.

Targeted Interventions section of the config file (see Figure 17) is active only during the ***sim*** command phase (i.e. it is ignored during an init), and allows you to alter key simulation variables, such as specific membrane permeability, in a time-dependent manner, where changes are applied to specific tissue profiles. The following types of targeted interventions are available:

→ Temporarily change membrane permeability to Na^+

→ Temporarily change membrane permeability to K^+

→ Temporarily change membrane permeability to Cl^-

→ Temporarily change membrane permeability to Ca^{2+}

→ Apply physical pressure to an area of the cell cluster

→ Temporarily apply an external voltage to global environmental boundaries (requires simulate ECM = True)

→ Break extracellular matrix junctions (requires simulate ECM = True)

→ Cut a piece out of an initialized cluster

In contrast to globally scheduled changes, *targeted interventions* are associated with a tissue profile, and can therefore be applied to specific regions of your model and not others.

Most targeted interventions follow a similar general format:

```
change Na mem:
    event happens:  True
    change start:  5
    change finish:  10
    change rate:  1
    multiplier:  10
    modulator function:  gradient_x
    apply to:  ['nerves','wound']
```

By setting “event happens” to True, you indicate that this targeted intervention will apply in your simulation run. The fields “change start”, “change finish” and “change rate” indicate the simulation time, in seconds, at which the interventions starts, finishes, and the rate at which the influence goes from none to all. The field “multiplier” will multiply the base level of the variable by the specified amount, for instance, if the original membrane permeability of sodium is Dm_Na = 1.0e-18 m/s2, a multiplier of 10 will take the value to 1.0e-17 m/s2, which begins to change at 5 seconds into the sim, is reached at 6 seconds into the sim, begins to return to the original value at 9 seconds, and is at the original value at 10 seconds.

The field “modulator function” allows you to let the intervention be further modified by a function, where your options are: “None” (use no function), “gradient_x” (to create a gradient in the parameter along the horizontal axis), “gradient_y” (to create a gradient in the vertical direction), and “gradient_r” (to create a radial gradient), or “gradient bitmap” (to use an imported bitmap of your choice to modulate the intervention). Further functions allow the change to be time-dependent (’periodic’, ’f-sweep’), but are applied to the whole cluster. Note, if a modulator function is used, the targeted intervention magnitude will vary smoothly

from zero (no effect) to the value specified by the “multiplier” field along the direction of the gradient. Modulator function properties, including the slope and offset of the gradients, can be specified a little further down in the configuration file under the “modulator function properties” fields (see Section 13 and Figure 19).

Finally, use the “apply to” field to specify how the targeted intervention applies to your cell population, by specifying tissue type profile names you have defined in the “Tissue Profile Designation” section. This must be specified as a tissue profile name, or a sequence of names, using the text formatting shown in the example (square brackets with tissue profile names in quotes and separated by commas if more than one profile name is desired).

12.1 Applying an external voltage

Applying external voltage is a unique kind of targeted intervention that simulates application of an externally derived electric field to the BETSE simulation. This is done by changing the voltage of the global boundary conditions for the environment in a time-dependent manner.

The text-block defining the ’apply external voltage’ intervention is similar to other targeted interventions, however, the ’peak value’ field represents the voltage (units Volts) that will be applied to each of the two boundaries. You then specify which two boundaries of the square, global simulation box hold the positive and negative voltage, with choices being “top”, “bottom”, “left”, and “right”.

```
apply external voltage:
    event happens:  True
    change start:  5
    change finish:  10
    change rate:  1
    peak value:  0.5
    positive voltage boundary:  top
    negative voltage boundary:  bottom
```

12.2 Cutting events

```
386 # Remove all cells selected by one or more cut profiles defined above at an arbitrary
387 # simulation time step. (Ignored unless at least one such profile is defined.)
388 cutting event:
389     event happens: True       # enable (True) or disable (False) this event?
390     apply to: ['surgery']     # list of names of all cut profiles selecting cells to be removed
391     break TJ: True            # break tight junctions? (otherwise assumes instant barrier healing)
392     wound TJ: 1.0e-1          # how leaky are the tight junctions at the wound (max 1.0)
393
```

Figure 18: The *cutting event* section of a BETSE configuration file.

Cutting events are a unique kind of targeted intervention for which it's worth mentioning a few additional points (Figure 18). In order for cutting events to work, you must have a tissue profile with a designation of "cuts" defined for your simulation (see Section 11.5).

The cutting event currently happens at the very beginning of a simulation run, so there are no time-specific settings for this type of targeted intervention.

13 Modulator Functions

Modulator functions are internally calculated simple functions (e.g. a gradient along the x-axis) or user-defined bitmaps (where red is read in as 1.0 and black as 0.0) used to modulate certain targeted interventions, as well as the growth of custom molecules and the presence of ion transporters and channels in your model. The 'gradient_x', 'gradient_y', and 'gradient_r' functions use a Hill function to calculate the modulating function along the chosen axis. Parameters for these functions can be edited in the Modulator Function Properties banner of the main BETSE config (see Figure 19). Alternatively, a bitmap image can be read in from a relative path (see Figure 19), where red is taken to be areas modulated by 1.0 and black to be 0.0. The term 'modulation' means that these functions are used to multiply growth or expression values of a parameter (e.g. the maximum rate of a transporter) over the entire cell cluster.

```
# MODULATOR FUNCTION PROPERTIES
#-------------------------------------------------------------------------------------------
# These are spatial or temporal functions, modifiable at any phase, that let you set a
# spatial gradient of production rate or other effect, or define a time periodic signal for
# production or other effect. Alter their parameters here.

modulator function properties:   # set properties of modulating functions which may be used in above:

  gradient_x:          # generate a spatial gradient of a property in the x-direction
    slope: 1.0         # rate of change of gradient per unit x-dimension
    x-offset: 0        # offset to all gradient values along the horizontal axis
    z-offset: 0
    exponent: 1        # Hill exponent for gradient values

  gradient_y:          # generate a spatial gradient of a property in the y-direction
    slope: 1.0         # rate of change of gradient per unit x-dimension
    x-offset: 0        # offset to all gradient values
    z-offset: 0
    exponent: 1        # Hill exponent for gradient values

  gradient_r:          # generate a radial spatial gradient of a property
    slope: 1.0         # radial gradient rate of change per unit radius
    x-offset: 0
    z-offset: 0
    exponent: 1        # Hill exponent for gradient values

  periodic:            # generate a sinusoidal oscillation of a property
    frequency: 10      # frequency of oscillation [Hz]
    phase: 0           # phase-offset of oscillation

  f_sweep:             # apply a sinusoidal oscillation of a property as a sequence of frequencies
    start frequency: 0.1e3    # start frequency
    end frequency: 1e3        # end frequency

  gradient_bitmap:   # generate a spatial gradient of a property from an imported bitmap image (png)
    file: geo/ellipse/gradient.png
    z-offset: 0.0
```

Figure 19: The *Modulator Function Properties* section of a BETSE configuration file.

14 Global Interventions

```
433 # ------------------------------------------------------------------------------------------------
434 # GLOBAL INTERVENTIONS
435 # ------------------------------------------------------------------------------------------------
436 # Schedule changes to globally-applied simulation variables such as membrane permeabilities,
437 # pump and gap junction function and environmental concentrations of ions.
438 # No file synch issues: Global intervention settings can be changed at any time between seed, init and sim.
439
440 # The following global changes affect the entire cluster during a sim run (not an init):
441
442 change K env:          # Change the environmental concentration of K+
443   event happens: False       # turn the event on (True) or off (False)
444   change start: 1.00           # time to start change [s]
445   change finish: 12.00         # time to end change and return to original [s]
446   change rate: 1.0             # rate of change [s]
447   multiplier: 20             # factor to multiply base level
448
449 change Cl env:         # Change the environmental concentration of K+
450   event happens: False       # turn the event on (True) or off (False)
451   change start: 5.0            # sim time to start change [s]
452   change finish: 25.0          # sim time to end change and return to original [s]
453   change rate: 2.0             # rate of change [s]
454   multiplier: 10             # factor to multiply base level
455
456 change Na env:         # Change the environmental concentration of K+
457   event happens: False       # turn the event on (True) or off (False)
458   change start: 1            # time to start change [s]
459   change finish: 9           # time to end change and return to original [s]
460   change rate: 1             # rate of change [s]
461   multiplier: 5              # factor to multiply base level
462
463 change temperature:
464   event happens: False       # turn the event on (True) or off (False)
465   change start: 1            # time to start change [s]
466   change finish: 9           # time to end change and return to original [s]
467   change rate: 0.5           # rate of change [s]
468   multiplier: 0.5            # factor to multiply base level
```

Figure 20: The *Global Interventions* section of a BETSE configuration file.

Global Interventions are events that affect the state of the entire cell population or environment. At present you can specify a transient change to:

→ Environmental K+ levels

→ Enviromental Cl- levels

→ Envirnomental Na+ levels

→ World Temperature

→ Block all gap junctions (or a randomly targeted sub-population of gap junctions)

→ Block NaK-ATPase pumps

To implement global scheduled changes, see the *Global Interventions* section of the config file, where there is a list of possible global interventions and options under each type of change. Most of the global interventions are defined by text blocks that follow a similar format:

```
change K env:
    event happens:  True
    change start:  5
    change finish:  100
    change rate:  10
    multiplier:  10
```

To make the global intervention apply in your simulation run, set the 'event happens' field to True. Specify when in the simulation you want the property to start changing ('change start'), when you want the change to finish ('change finish'), the rate at which you desire a change to go from baseline to changed and vice versa ('change rate'), and the factor by which you want the change to deviate from baseline values ('multiplier'). The times and rates are specified in seconds, while the multiplier is unit-less.

If 'simulate ECM' (see section 9.2) is True, when applying a global change to a concentration, BETSE introduces the altered concentration from the global boundaries of the environmental bounding box. Also, when a full envirnoment is simulating, changing the concentration of a particular ion must be charge balanced (i.e. you can't just add in pure K+ ions to the environment, it must come in as a salt with equal moles of an anion). Therefore, K^+ is introduced as equal parts K^+& M^-, Cl^- as equal parts Na^+ & Cl^-, and Na^+ as equal parts Na^+& M^-. If 'simulate ECM' is False, the ion is simply introduced as the environmental charge and voltage are not a concern.

Note, the 'block gap junctions', 'block NaKATPase pump' and 'block HKATPase pump' headers do not have a 'multiplier' field as the block currently sets all activity to zero. When blocking gap junctions, you're able to specify the randomly targeted fraction of gap junctions to be blocked by typing an integer from 1 to 100 in the 'random fraction' field:

```
block gap junctions:
    event happens:  True
    change start:  5
    change finish:  100
    change rate:  10
    random fraction:  75
```

15 BIGR Networks

BETSE allows users to define bioelectricity-integrated gene and reaction networks (BIGR networks). BETSE provides two options for simulating BIGR networks: (1) a *General Network* that is defined within the main config file and (2) a GRN that is specified in an 'extra config' file with path specified under the *Gene Regulatory Network* section of the main config file (Figure 22). Both of these networks share the same defining configuration code blocks. Example GRN extra config files are provided automatically whenever the ***config*** command is called and appear in the folder "extra config files". Note that the Gene Regulatory Network can be run, saved and plotted without bioelectrical computations using the 'sim grn' command.

15.1 General Network

```
518 #-------------------------------------------------------------------------------------------
519 #  GENERAL NETWORK
520 #-------------------------------------------------------------------------------------------
521 # The "General Network" allows you to define a bioelectricity-integrated gene and reaction
522 # network with custom biomolecules, chemical reactions, channels, and transporters. Use this
523 # section to define general network main properties.  See "extra_config_template.yaml" in the
524 # "extra_configs" directory for examples.
525 #
526 # NOTE: The "optimization" is *not* required to run a general network; this is a tool that
527 # supplies you with a list of membrane diffusion constants and reaction/transporter maximum
528 # rate constants that are a "best fit" match to the target Vmem and initial concentrations
529 # listed for the "custom biomolecules" section of the network.
530
531 general network:
532
533   implement network: True   # Turn the entire network defined below on (True) or off (False)
534
535   expression data file: extra_configs/expression_data.yaml
536
```

Figure 21: The *General Network Features* section of a BETSE configuration file.

The General Network can be turned on/off using the “implement network” field under the General Network Features section of the main config. Use this first portion of the network config block to specify a relative path to an ’expression data file’ (details discussed in Section 15.9).

15.2 Custom Biomolecules

Under the “custom biomolecules” header, any number of code blocks can be included which each define a chemical substance. Each substance code block begins with a “- name:” field, where the the “-” is an important feature that signifies the beginning of a new chemical substance block. Within each substance code block, the user can specify membrane and free diffusion constants (Dm and Do, respectively), substance charge z, the initial environmental and cell concentrations, whether the substance can permeate through tight junctions (e.g. dissolved oxygen), and whether it is impermeable to gap junctions (e.g. large macromolecules and proteins).

Substances can have an optional “growth and decay” sub-block, which can simply be commented out or deleted if it is not needed. The “growth and decay” sub-block defines base rates of maximum production, decay, tissue profiles to which growth applies (or “all” for all

cells in the cluster), and the ability to specify a modulator function (or “None” for homogeneous cell production). In the “growth and decay” sub-block, other user-defined substances can be indicated as activators or inhibitors of substance production, which are provided as a list of substance names inside square brackets.

Plotting settings for a substance’s concentration are indicated in the chemical substance code block:

```
- name:  'Substance A'
    Dm:  0.0
    Do:  1.0e-9
    z:  -2
    env conc:  0.0
    cell conc:  2.5
    TJ permeable:  False
    GJ impermeable:  False
    TJ factor:  1.0
    ignore ECM: False
    growth and decay:
        production rate:  1.0
        decay rate:  0.5
        Km:  1.0
        n:  1.0
        apply to:  ['spot']
        modulator function:  gradient_x
        activators:  None
        Km activators:  None
        n activators:  None
        inhibitors:  ['Substance B']
        Km inhibitors:  [1.0]
        n inhibitors:  [3.0]
    plotting:
        plot 2D: True
        animate:  True
        autoscale colorbar:  True
        max val:  2.0
        min val:  0.0
```

Any number of substance code blocks (each with its own unique name) can be included in a BETSE network simulation. For details regarding custom biomolecule definition, please see the “extra_config_template.yaml” file available in the “extra configs” folder that is created when the betse ***config*** command is called.

15.3 Custom Chemical Reactions

Chemical reactions are defined under the header 'Custom Chemical Reactions' of a BIGR network config file or block. Chemical reactions allow some user-defined substances to be converted into others, as in a standard reaction. Each chemical reaction is defined by its own code block, which allows a user to specify a reaction name, list reactants, list products, specify a maximum rate for the reaction, state a standard free energy (where "None" indicates a non-reversible reaction), and allow other user-defined substances to activate or inhibit the base reaction rate:

```
- name:  consume_A
    reaction zone:  cell
    reactants:  ['A']
    reactant multipliers:  [1]
    Km reactants:  [1.0]
    products:  ['B', 'C']
    product multipliers:  [1, 1]
    Km products:  [0.1, 0.1]
    max rate:  5.0e-3
    standard free energy:  None
    reaction activators:  None
    activator Km:  None
    activator n:  None
    activator zone:  None
    reaction inhibitors:  None
    inhibitor Km:  None
    inhibitor n:  None
    inhibitor zone:  None
```

Any number of reaction code blocks (each with its own unique name) can be included in a BETSE network simulation. For details regarding custom reaction definition, please see the "extra_config_template.yaml" file available in the "extra configs" folder that is created when the betse ***config*** command is called.

15.4 Custom Transporters

Ion pumps and various transporters are defined under the header 'Custom Transporters' of a BIGR network config file or block. Transporters are chemical reactions where one or more of the reactants or products (which include user-defined substances as well as all core ions of the base BETSE simulator) may involve a transport from the inside to the outside of the cell, thereby allowing a user to define a wide range of different transporters. Each transporter is defined by a code block, which allows a user to specify a reaction name, list

reactants, list products, specify a maximum rate for the reaction, state a standard free energy (where “None” indicates a non-reversible reaction), and allow other user-defined substances to activate or inhibit the base reaction rate. For transporters, one must additionally specify which substances are transported into or out of the cell. Ions are specified by their common names (’Na’, ’K’, ’Cl’, ’Ca’):

```
transporters:
- name:  Transport_A
    reaction zone:  cell
    reactants:  ['Substance_A', 'Na']
    reactant multipliers:  [1, 1]
    Km reactants:  [0.1, 1.0]
    products:  ['Substance_A', 'Na']
    product multipliers:  [1, 1]
    Km products:  [0.1, 1.0]
    transfered out of cell:  ['Na']
    transfered into cell:  ['Substance_A']
    max rate:  1.0e-5
    standard free energy:  0
    apply to:  ['nerves', 'spot']
    ignore ECM: True
    transporter activators:  None
    activator Km:  None
    activator n:  None
    activator zone:  None
    inhibitors:  None
    inhibitor Km:  None
    inhibitor n:  None
    inhibitor zone:  None
```

Any number of transporter code blocks (each with its own unique name) can be included in a BETSE network simulation. For details regarding custom transporter definition, please see the “extra_config_template.yaml” file available in the “extra configs” folder that is created when the betse ***config*** command is called.

15.5 Custom Channels

Custom ion channels are defined under the header ’Custom Channels’ of a BIGR network config file or block. Custom channels allow a user to evoke one of the available channel classes (e.g. voltage gated Na channels) and a specific channel type (e.g. NaV1p2) and to optionally have custom substances activate/inhibit (co-regulate) the channel behavior:

```
- name:  ATP-K-channel
    channel class:  K
    channel type:  KLeak
    max Dm:  1.0e-8
    apply to:  all
    channel activators:  ['Substance A']
    activator Km:  [0.1]
    activator n:  [3.0]
    activator zone:  ['cell']
    channel inhibitors:  ['Substance B']
    inhibitor Km:  [1.5]
    inhibitor n:  [3.0]
    inhibitor zone:  ['cell']
```

Any number of custom channel code blocks (each with its own unique name) can be included in a BETSE network simulation. For details regarding custom channel definition, please see the “extra_config_template.yaml” file available in the “extra configs” folder that is created when the betse ***config*** command is called.

The Custom Channels section of the config file allows you to turn on specialty channels that generate changes to membrane permeability of specific ions (e.g. Na +) that also occur in relation to feedback values with other simulation variables, such as V_{mem} . Most dynamic ion channel models currently available in BETSE were sourced from experimental data derived Hodgkin-Huxley style models sourced from the online database, Channelpedia:

http://channelpedia.epfl.ch/

The type of channel is specified by defining one of the available channel classes, and specifying a channel type that belongs to the class, where available channel classes and respective types are summarized in Table 1.

The “max Dm” field allows you to specify the maximum contribution to membrane diffusion coefficient for the ions that move through the channel when it is maximally activated/open.

In addition to intrinsic feedback of the channel’s open/closed state with V_{mem}, the channel’s state may be altered by concentrations of ions or custom biomolecules defined in your network to produce a ligand-gated ion channel. These can be applied using the ’activator’ and ’inhibitor’ blocks, as detailed above.

15.6 Custom Modulators

Custom modulators allow any of the user-defined substances to activate or inhibit the state of core BETSE simulation objects, such as gap junctions and the Na/K-ATPase pump:

Channel Class	Channel Types	Description
Na	Nav1p2, Nav1p3, NavRat1, NavRat2, Nav1p6, NaLeak	Voltage gated and 'leak' sodium channels
K	K_Slow, K_Fast, Kv1p1, Kv1p2, Kv1p3, Kv1.4, Kv1.5, Kv1.6, Kv2p1, Kv2p2, Kv3p1, Kv3p2, Kv3p3, Kv3p4, Kir2p1, KLeak	Voltage gated and 'leak' potassium channels
Cl	ClLeak	Voltage gated and 'leak' chloride channels
Ca	Cav1p2, Cav1p3, Cav2p1, Cav2p3, Cav3p1, Cav3p3, Cav_L2, Cav_L3, Cav_G, CaLeak	Voltage gated and 'leak' calcium channels
Cat	CatLeak, CatLeak2	Voltage gated and 'leak' cation channels (permeable to K+, Na+ and Ca++)
Fun	HCN1, HCN2, HCN4	Hyperpolarization activated cyclic nucleotide gated channels
ML	Kv_ML1, Kv_ML2, Nav_ML, Cav_L_ML, Cav_N_ML, Cav_T_ML, Kir_ML, HCN2_ML, HCN4_ML	Various Morris-Lecar channel models

Table 1: Summary of BETSE channel 'class' and channel 'type' for available dynamic channels in the BETSE ion channel library.

```
- name:  GJ-block
    target:  gj
    max effect:  1.0
    activators:  ['Substance A']
    activator Km:  [0.1]
    activator n:  [3.0]
    activator zone:  ['cell']
    inhibitors:  ['Substance B']
    inhibitor Km:  [1.5]
    inhibitor n:  [3.0]
    inhibitor zone:  ['cell']
```

Any number of custom modulator code blocks (each with its own unique name) can be included in a BETSE network simulation. For details regarding custom modulator definition, please see the "extra_config_template.yaml" file available in the "extra configs" folder that is created when the betse ***config*** command is called.

15.7 Gene Regulatory Networks

An alternative regulatory network can operate in BETSE, but which can be read into a BETSE configuration model file from an extra configuration file can be defined using the "gene regulatory network settings" fields. To use the gene regulatory network, set the "gene regulatory network simulated" boolean to "True" (see Figure 22), and supply a path to the extra config file (which is relative to the main config file) using the "gene regulatory network config" field to specify a path (see Figure 22). Definition of the regulatory network in the extra config file follows the exact same format as that described for the General Network described above, with the same blocks defining Network Optimizer, Custom Biomolecules, Custom Chemical Reactions, Custom Transporters, Custom Channels, and Custom Modulators, as described above. A separate "gene regulatory network" might be preferred to a regulatory network embedded in the main config for cases where the network is very long and complex.

If valid paths to valid extra config files are supplied under the "gene regulatory network config" fields, the BETSE command:

```
betse sim-grn my_config.yaml
```

can be called to test-drive the metabolism and GRN networks, respectively, without running the bioelectric component of the BETSE simulator. Note that the above call to ***sim-grn*** reference the main config file (e.g. my_config.yaml), which in turn references the metabolism config or gene regulatory network config under paths specified in *the Gene Regulatory Network* banner of the main config file.

```
betse plot sim-grn my_config.yaml
```

```
644 #--------------------------------------------------------------------------------------------
645 # GENE REGULATORY NETWORK
646 #--------------------------------------------------------------------------------------------
647 # Define an auxillary network in an external config file and read it in and run from that
648 # extra file.
649
650 gene regulatory network settings:
651
652   gene regulatory network simulated: False # Enable gene regulatory network (GRN)?
653
654   # Configuration file for network. Ignored if "gene regulatory network simulated" is False.
655   gene regulatory network config: extra_configs/grn_basic.yaml
656
657   sim-grn settings: # Settings for "betse sim-grn" subcommand, calculating network at last
658                     # time step of previously run simulations.
659     run as sim: False      # Run network as an initialization (False) or simulation (True)?
660     run network on: seed # Type of previously run simulation phase to "piggyback" (i.e.,
661                          # restart) the current network from, as any following string:
662                          # * "seed", networking an uninitialized, unsimulated cell cluster
663                          #   (i.e., neither initialized to steady-state concentrations nor
664                          #   simulated with non-steady-state bioelectrical phenomena).
665                          # * "init", networking an initialized, unsimulated cell cluster.
666                          # * "sim", networking an initialized, simulated cell cluster.
667     save to directory: RESULTS/GRN # Directory containing the following file.
668     save to file: GRN_1.betse.gz    # File of networking results created by "betse sim-grn".
669                                     # Supported filetypes are as above.
670     load from:             # File of prior networking results created by "betse sim-grn" to
671                            # restart this network from, relative to configuration directory
672                            # (e.g., "RESULTS/GRN/GRN_1.betse.gz"). Defaults to empty string,
673                            # in which case a new network is created from scratch.
674     time step: 0.1         # Time step-size [s]
675     total time: 1.8e2      # Time to end sim run [s]
676     sampling rate: 1.8e1 # Period to sample data [s] (at least time step or larger)
677
```

Figure 22: *Gene Regulatory Network* section of a BETSE configuration file.

15.8 Network Output Tools

When simulations running any kind of custom network are plotted, BETSE automatically exports a csv file with LaTeX-compatable math expressions corresponding to all of the growth and decay, chemical reactions, and transporter expressions involved in the BIGR network. In the Results section of the main config file, the field:

```
plot networks:  True
```

Will create basic plots of BIGR network structure (example shown in Figure 23), if the programs networkx and pydot (optional dependencies) are installed on your system. In network plots, black arrows show paths of reaction flow, whereas blue edges with circular ends indicate activator relationships and red edges with flat ends indicate an inhibitory relationship. Substances and V_{mem} are indicated by elliptical shapes, reactions by square boxes, transporters by diamonds, and ion channels by pentagons.

Note that this feature requires installation of optional dependencies Networkx and Pydot to function.

15.9 Using Expression Profiles

Expression profiles allow relative levels of a biomolecule's growth, reaction rate, transporter maximum rate, or channel maximum diffusion constant to be specified for each tissue profile of a model. These relative levels may be obtained from experimental data (your own, or downloaded from databases), allowing for generation of a more realistic bioelectric model.

To use expression profiles, first specify an external 'expression data file' using the relative link under the network header, as shown in Figure 24.

This 'expression data file' is a yaml-format file that specifies modulatory levels of a factor that will be applied to each tissue profile. For example, if in your network you define a transporter named 'H-ATPase', and wish to specify 'expression data' to modulate the 'max rate' of the transporter , then specify 'express' in the 'apply to' field of the transporter (see Figure 25). Then, in the expression data file for an entry entitled 'H-ATPase', specify levels of activity/expression for each of the tissue profiles defined in your model (see Figure 26), where the numbers will multiply the 'max rate' term of the H-ATPase for each tissue profile entry. In order to use expression data, it is important to have an entry for all tissue profiles defined in your model.

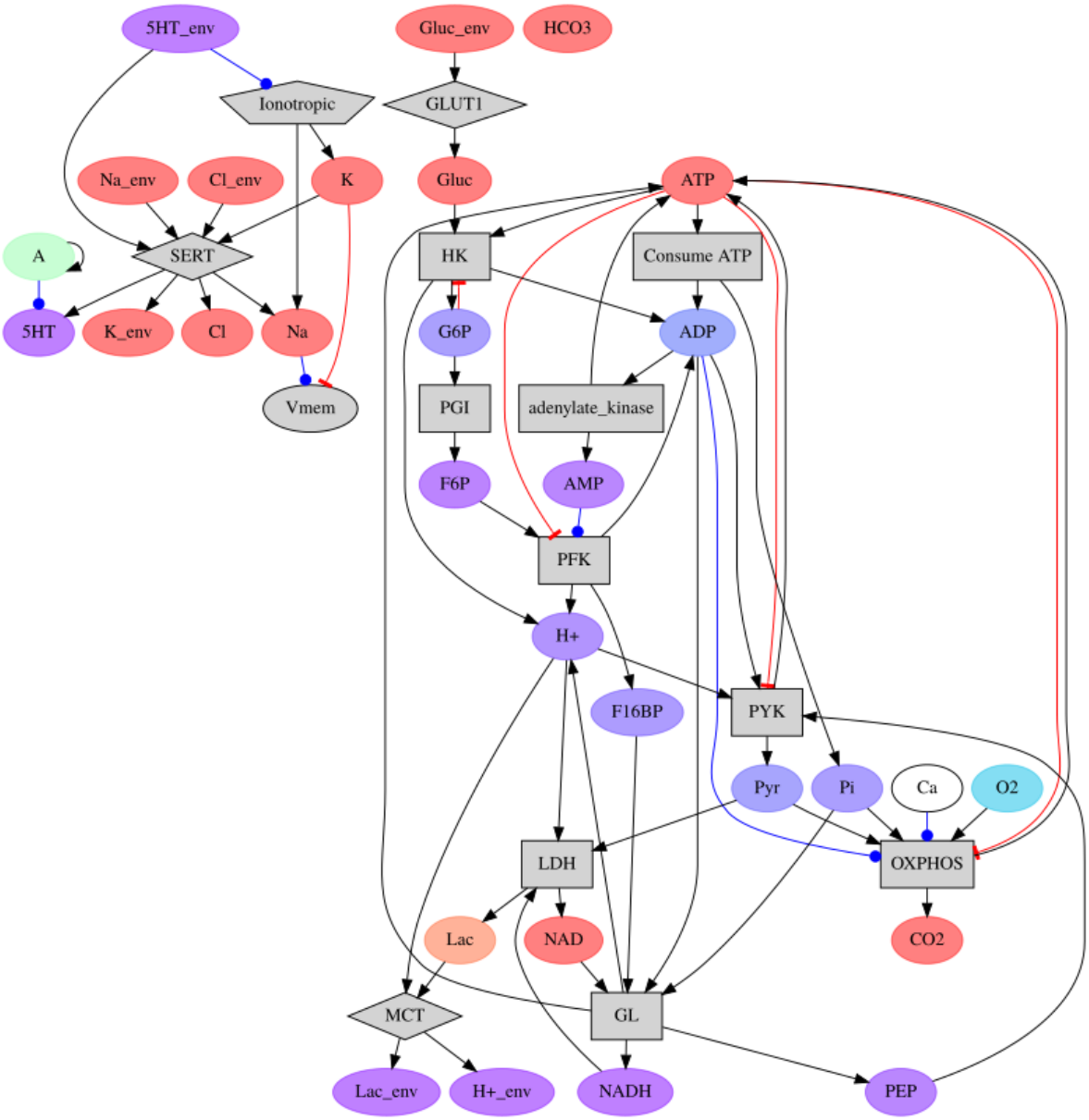


Figure 23: Example network plot created in BETSE.

```
518 #-------------------------------------------------------------------------------------------
519 #  GENERAL NETWORK
520 #-------------------------------------------------------------------------------------------
521 # The "General Network" allows you to define a bioelectricity-integrated gene and reaction
522 # network with custom biomolecules, chemical reactions, channels, and transporters. Use this
523 # section to define general network main properties.  See "extra_config_template.yaml" in the
524 # "extra_configs" directory for examples.
525 #
526 # NOTE: The "optimization" is *not* required to run a general network; this is a tool that
527 # supplies you with a list of membrane diffusion constants and reaction/transporter maximum
528 # rate constants that are a "best fit" match to the target Vmem and initial concentrations
529 # listed for the "custom biomolecules" section of the network.
530
531 general network:
532
533   implement network: True  # Turn the entire network defined below on (True) or off (False)
534
535   expression data file: extra_configs/expression_data.yaml
536
```

Figure 24: The yaml file containing expression data is loaded from the relative path defined under the 'expression data file' field of the general or gene regulatory network.

```
1930  transporters:
1931
1932    - name: H-ATPase          # unique identifier for the transporter **DO NOT** end with '_growth' or '_decay'
1933      reaction zone: cell     # reaction zone identifier: 'cell' or 'mit'; where reaction takes place
1934      reactants: ['H+']   # list of reagents; must be from set of names defined for biomolecules in above
1935      reactant multipliers: [1]  # reaction coefficients for reagents, in same order as reagent definition
1936      Km reactants: [0.05]    # list of half-max coefficients for reagents, in same order as reagents list
1937      products: ['H+']          # list of products; must be from set of names defined for biomolecules in above
1938      product multipliers: [1]   # reaction coefficients for products, in same order as product definition
1939      Km products: [1.0]          # list of half-max coefficients for reagents, in same order as reagents list
1940      transfered out of cell: ['H+']   # items of reaction moved out of cell during forwards reaction
1941      transfered into cell: []      # items of reaction moved into cell during forwards reaction
1942      max rate: 1.5e-11            # 1.0e-7 maximum rate of transport [mol/(m2*s)]
1943      standard free energy: -36.0e3  # standard free energy of reaction in J/mol. 'None' creates non-reversible reaction
1944      apply to: express       # apply transporter only on certain profiles?
1945      ignore ECM: False
```

Figure 25: Telling BETSE to read from the expression data file by setting 'apply to' field to 'express' for a transporter named 'H-ATPase' under the 'transporters' section of a regulatory network defined in a BETSE configuration file.

```
319 H-ATPase:
320   col_1: 672.792
321   col_2: 12307.630000000001
322   cortex_10: 3378.7509999999997
323   cortex_2: 1373.693
324   cortex_3: 2385.4300000000003
325   cortex_4: 2818.86
326   cortex_5: 2484.755
327   cortex_6: 4409.01
328   cortex_7: 5637.5
329   cortex_8: 7086.07
330   cortex_9: 1787.817
331   endodermis_10: 1708.613
332   endodermis_2: 648.005
333   endodermis_3: 1383.06
334   endodermis_4: 1701.154
335   endodermis_5: 1464.5520000000001
336   endodermis_6: 2613.71
337   endodermis_7: 3070.64
338   endodermis_8: 4051.8599999999997
339   endodermis_9: 1070.944
```

Figure 26: Entry in the extra configuration file detailing expression data levels for a transporter named 'H-ATPase'. The H-ATPase transporter is defined under the 'transporters' section of a regulatory network defined in a BETSE configuration file. The expression data lists expression levels for the H-ATPase transporter on all tissue profiles defined in the main BETSE configuration file.

16 Variable Settings

```
#-----------------------------------------------------------------------------------------------------
# VARIABLE SETTINGS
#-----------------------------------------------------------------------------------------------------
# These settings can be changed in between seed, init and sim runs, meaning there are no synchronization issues.

variable settings:

  env boundary concentrations:    # these concentrations are placed at the global boundary during init and sim
                                  # altering these is an easy way to change extracellular ion concentrations
                                  # uncomment the fields below to set new env boundaries, otherwise feature ignored!

#     Na: 145.0     # Na+ concentration at bound [mM]
#     K: 4.0        # K+ concentration at bound [mM]
#     Cl: 115.0     # Cl- concentration at bound [mM]
#     Ca: 2.0       # Ca2+ concentration at bound [mM]
#     P: 9.0        # Protein- concentration at bound [mM]
#     M: 29.0       # Anion (bicarbonate) concentration at bound [mM]

  temperature: 310              # system temperature [K]


# FIXME: DECIDE on how to handle these deformations
  deformation:

    turn on: False          # include deformation under the osmotic pressure influx and electric fields?
    galvanotropism: 1.0e-9        # factor by which cells change shape to global electric field (can be 0.0 or negative)
    viscous damping: 0.01         # viscous damping of elastic waves in tissue (0.0 to 1.0; larger = higher damping)
    fixed cluster boundary: True    # Exterior cells of the cluster are fixed (True) or free to move (False)
    young modulus: 1.0e3          # Young's modulus (elastic modulus) of individual cell [Pa] (1 kPa to 10 e6 Pa)

# FIXME: FIX THESE
  pressures:

    include osmotic pressure: False     # include osmotic pressure in flow & deformation?
    membrane water conductivity: 1e-2   # mem conductivity for osmotic inflow (aquaporin per unit surface) (~ 1e-3)
```

Figure 27: The *Variable Settings* section of a BETSE configuration file.

The variable settings section of the config file collects a number of simulation parameters that can be changed at any point in an init or sim, without breaking synchronization between previously run BETSE output files.

16.1 Environmental Boundary Concentrations and Temperature

The environmental boundary concentrations represent fixed concentrations that will be applied to the global boundaries of the environmental square during simulation (valid only when "simulate extracellular spaces" is True). The fields under "env boundary concentrations" can be deleted or commented out, which assigns environmental concentrations equal to those of your selected ion profile (see Figure 27). However, when these fields are active, they allow you to respecify new steady-state concentration of ions in the environment. Note that these "env boundary concentrations" should sum to zero charge for the ions involved.

You can also specify the temperature of the world in degrees Kelvin [K]. Note zero degrees Celsius is 273 degrees K.

```
temperature:  310
```

16.2 Deformations

```
deformation:
    turn on:  False
    galvanotropism:  1.0e6
    viscous damping:  0.01
    fixed cluster boundary:  True
    young modulus:  1.0e3
    time dependent deformation:  False
```

16.3 Osmotic Pressure

Setting osmotic pressure to "True" enables calculation of the osmotic pressure difference between the intracellular and extracellular spaces using the total sum of ions inside and out of the cell.

```
pressures:
    include electrostatic pressure:  True
    include osmotic pressure:  False
    membrane water conductivity:  1e-12
```

16.4 Noise

In BETSE simulations, there are two types of noise: a constant (static) noise representing a variation in V_{mem} for individual cells and determined by adding a variation in the level of K^+ leak channels on cell membranes, and a flickering (dynamic) noise represented a small random walk on V_{mem} created by slightly altering the concentration of negatively charged proteins in cells. Note that both types of noise can cause simulation instability (and consequent program crash with error message) and if this is the case either decrease the "time step" by using the "custom init" or "custom sim" time profiles in the *Initialization Settings* or *Simulation Settings* sections. Alternatively, try decreasing the level of noise.

To specify static noise, simply change the value of "static noise level" to something above 0. An appropriate range is 0 (all cells equivalent; no static noise) to 10.0 (very high variation).

```
static noise level:  0.5
```

If dynamic noise is desired, first set the "dynamic noise" field to True. In version 1.0, dynamic noise will only be present during the 'sim' phase of BETSE, and not during the 'init'.

```
dynamic noise:  True
```

Next, set the level of dynamic noise in the "dynamic noise level" field. This value has a big influence on the simulation, so it's best to keep values small. An appropriate range is from 0 to 1e-6. Note higher values may very well cause simulation instability.

```
dynamic noise level:  1e-7
```

16.5 Gap Junction Properties

Cells in the cluster are connected by (optionally) voltage-sensitive gap junctions.

The "gap junction surface area" field controls the base conductivity of the gap junctions, and is defined in terms of the fraction of cell membrane area that is allocated to gap junctions, therefore, the gap junction surface area parameter is unitless. Gap junctions of $1.0e^{-9}$ or smaller are effectively closed, $1.0e^{-8}$can be considered low gap junction conductivity, $1.0e^{-7}$ is moderate gap junction conductivity, and $1.0e^{-6}$ very high gap junction conductivity. Note that simulations using a time step of $1.0e^{-2}$ s can generally only handle gap junction surface areas of $1.0e^{-7}$ or smaller.

```
gap junction surface area:  1.0e-7
```

Gap junctions are optionally voltage sensitive, meaning they open and close depending on the value of intercellular voltages. To set gap junctions as voltage sensitive, use the field under the gap junctions section of *World Variables*:

```
voltage sensitive gj:  True
```

The voltage sensitivity of gap junctions has significant effects on the cluster's electrochemical dynamics. Use the 'gj voltage threshold' field to specify the cell-cell voltage at which the gap junctions close in volts [V]. The suggested range is from $10e^{-3}$ to $50e^{-3}$ V. Note that a sigmoidal curve is used to control gap junction closure, so this 'gj voltage threshold' represents the voltage at which gap junctions are half closed.

```
gj voltage threshold:  15e-3
```

Use the 'gj voltage window' field to specify the range over which gap junctions go from open to shut [V]. The suggested window value is from 5e-3 to 40e-3. Since sigmoidal curves are being used to control gap junction closure, this window specifies how far below the threshold the gap junction begins to close, and how far above the threshold the gap junction becomes fully closed.

```
gj voltage window:  10e-3
```

16.6 Tight Junction Properties

If "simulate extracellular spaces" (see section 9.2) is True, BETSE considers cells to have extracellular environments. In real tissue, cells are often joined by tight junctions or adherens junctions, which may limit diffusion around cells to various extents.

If simulating a full environment, you can optionally simulate the inclusion of tight junctions by using the "tight junction scaling" field of the *Variable Settings* section of the config file:

```
tight junction scaling:  1.0e-5
```

The "tight junction scaling" factor you supply is used to multiply the free diffusion constants of all ions and substances included in your model, for the extracellular spaces of cells located at the outer boundary of the cell cluster (BETSE automatically detects the boundary for you, see Figure 28).

Likewise, cell-cell connections within a cluster may have other types of junctions (e.g. adherens), which may alter diffusion around cells compared to free diffusion in liquid. To include

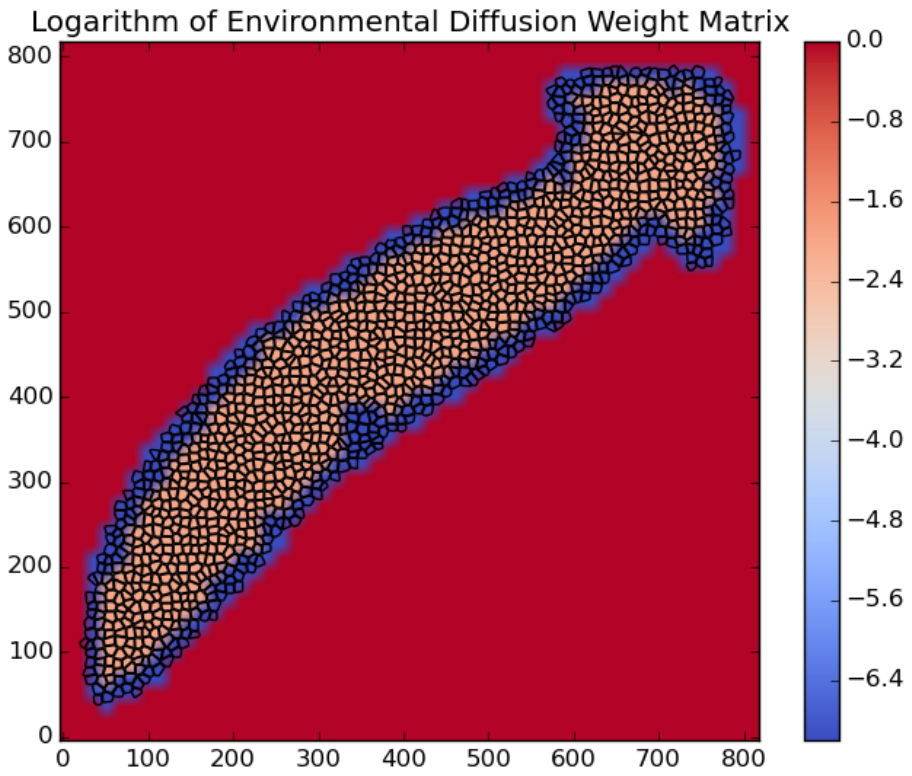


Figure 28: When simulating a full environment, BETSE automatically detects the outer boundary of your cell cluster and can assign different diffusion constants to the outer media, outer cluster boundary, and internal cells. This 'Logarithm of Environmental Diffusion Weigh Matrix' figure (produced automatically when plotting the seeded cluster when 'simulate ECM' is True) shows extracellular spaces of the cluster boundary to have $1e^{-7}$ times lower diffusion constants than free media surrounding the cluster, while interior cells have $1e^{-2}$ times lower diffusion constants than the free media.

the influence of internal cell-cell connections (i.e. those away from the outer cluster boundary), set the “adherens junction scaling” factor:

```
adherens junction scaling:  1.0e-1
```

To get a better idea of the difference between the cluster boundary specific tight junction scaling and internal adherens junction scaling effects for an example cluster, see Figure 28.

If you don’t want to include the influence of tight junctions at the cluster boundary or adherens junctions for cells in the cluster, simply set the value of these two fields to 1.0.

16.7 Comparing BETSE Vmem Output to Goldman Equation

The Goldman equation is renowned for its ability to predict steady-state membrane potential given the ideal gas constant (R), temperature in Kelvin (T), the Faraday constant (F), membrane permeabilities for all cations (P_M), membrane permeabilities for all ions (P_A), and the concentration of cations and anions inside and outside of a cell (M_{in}, M_{in} and A_{in}, A_{out}, respectively):

$$V_{mem} = \frac{R\,T}{F}\,ln\left(\frac{\sum P_M\,M_{out} + \sum P_A\,A_{in}}{\sum P_M\,M_{in} + \sum P_A\,A_{out}}\right) \tag{1}$$

BETSE now has an option allowing you to use the Goldman equation to calculate an alternative value of V_{mem} from a simulation’s parameters, for validation and comparison purposes.

```
use Goldman calculator:  True
```

For simulations that do not limit extracellular diffusion via tight and adherens junctions and which do not have high values of gap junction surface area (aspects not accounted for by the Goldman equation), BETSE V_{mem} compares directly (less than 1% discrepancy) to V_{mem} obtained from the Goldman equation (Table 2).

17 Results Output and Visualization

The *Results* section of the configuration file (Figure 29) allows you to specify what kind of plots, animations and raw data exports you want, whether you want to automatically save images, and where you want to save all of this information on your computer. The initial portion of the results section allows you to specify whether you wish to see plots while solving,

P_m Na	P_m K	P_m Cl	ext K [mmol/L]	V_{mem} BETSE [mV]	V_{mem} Goldman [mV]
1	200	20	5	-84.298	-84.252
1	100	20	5	-81.842	-81.833
1	100	20	35	-36.801	-36.237
1	15	2	5	-61.418	-61.304
10	1	1	5	36.097	36.601

Table 2: Comparison between Vmem obtained from BETSE and from Goldman Equation for a range of different membrane permeability profiles and environmental potassium concentration. Membrane permeability (P_m) are relative ratios and are therefore unit-less.

if you wish to see plots when an init or sim finishes, or if you simply want plots to be saved to disk. Settings for animations are also found in the first portion of the Results section. Please see the default BETSE configuration file for annotated details on each field.

```
# ------------------------------------------------------------------------------------------------
# RESULTS
# ------------------------------------------------------------------------------------------------
# Configure results viewing and saving options for simulations. All of the
# parameters under 'results options' can be altered at any time between seed,
# init and sim.

results options:

  show cells: True             # Visualize discrete cells or only a homogeneous gradient of cell data?
  enumerate cells: False       # Number the cells on 2D plots with their simulation index?

  #FIXME: Refactor this into a profile list and add plots with legends to 1D plots.
  plot cell index: 0           # 0-based index of cell to visualize for single-cell time plots.
                               # (Set "enumerate cells: True" above to display cell indices).

  overlay currents: False      # Overlay electric current or concentration flux streamlines on 2D plotted data?
  streamline density: 2.0      # For current plots, density of streamlines in the range [0.5, 5.0].
  plot total current: True     # For currents and morphgen fluxing, if simulating extracellular spaces,
                               # plot total current if True or plot gap junction current if False.

  plot cutlines: True          # Plot any scheduled cut-lines on the cell cluster plot as black regions.
  plot masked geometry: True # Plot the geometry using the overall shape as a mask.
```

Figure 29: The *Results* section of a BETSE configuration file.

In BETSE, each cell is indexed by a number. Every time an initialization is run and a new world created, the numbering scheme changes. To see what number corresponds to what cell in the cluster, set the 'enumerate cells' field to True. In the absence of an interactive GUI, this can help you decide which cell you are interested in getting specific data on (for single cell graphs plotted over time) or to target with a scheduled intervention. A warning that

'enumerate cells' should not be used for world sizes over 200 um as there are far too many cells to resolve the numbers.

```
enumerate cells:  True
```

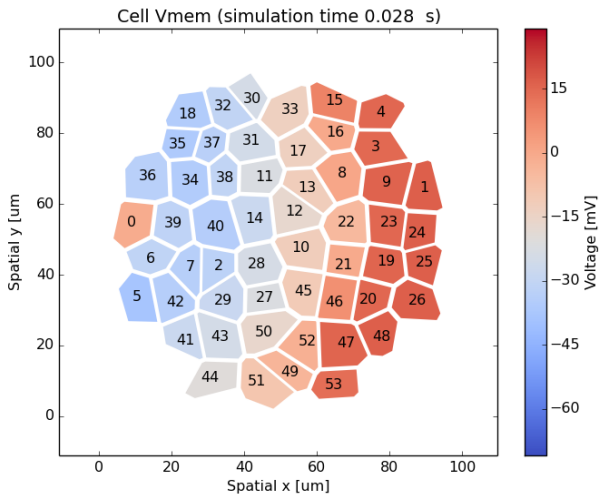


Figure 30: The enumerate cells option puts the cell index on the plot so you can identify individual cells.

To tell BETSE what cell you're interesting in plotting (single cell graphs) and exporting single-cell data, specify a cell index in the 'plot cell index' field (note this must be an integer).

```
plot cell index:  5
```

BETSE exports different kinds of plots, which have been categorized as:

→ **plot while solving** — a plot of Vmem that is displayed or saved while your simulation runs (can be disabled using commands under 'while solving'). These are plotted on the 'plot cell index'.

→ **after solving single cell plots** — list of possible line-graph plots showing a parameter in a single cell over time (e.g. see Figure 31)

→ **after solving cluster plots** — list of possible cell cluster plots showing a parameter over the entire cell cluster at last time-step (e.g. see Figure 32)

→ **after solving animations** — list of possible cell cluster animations showing a parameter over the entire cell cluster at each sampled time-step

Each of the above categories contains named plots that can be added to the list of plots that will be displayed and/or saved to disk when you run the *plot init* or *plot sim* commands. The list of available plots can be found in BETSE's default configuration file.

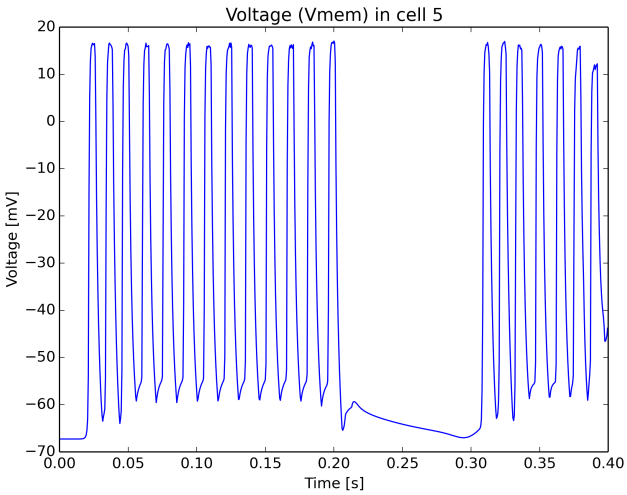


Figure 31: An example of cell Vmem plotted for cell index 5 in the cluster

BETSE allows you to have two main ways to display your 2D data: you can represent it as a color-value with each cell drawn as a discrete object (Figure 32, left) or you can interpolate the data value from the center of each cell and produce a smoothed representation (Figure 32, right). The interpolated option is particularly convenient (and faster to plot) for large cluster sizes (> 500 um) with many thousand cells. To plot discrete cells set the 'show cells' field to True, to plot interpolated data surfaces set this field to False.

```
show cells:  True
```

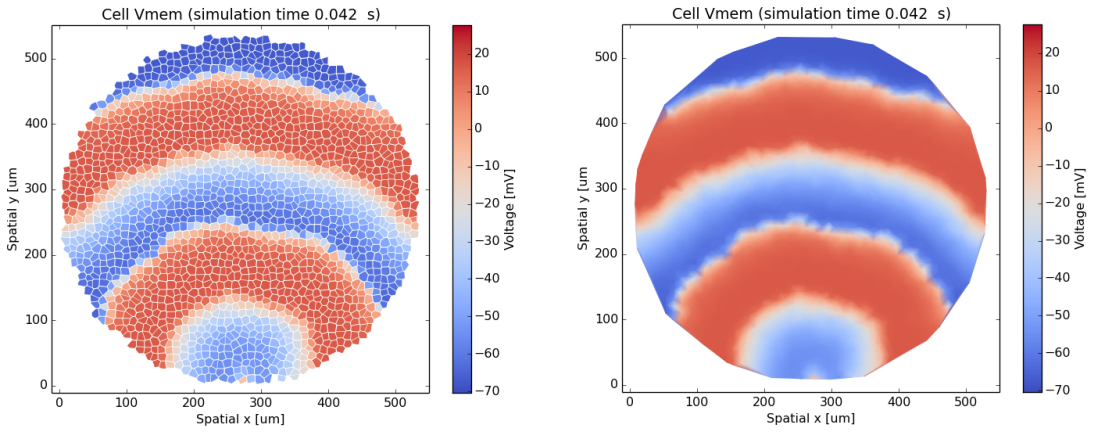


Figure 32: An example of 'show cells' being True (left image) versus False (right image)

If you wish to create a streamline plot of currents as an overlay on all of your plots and animations, choose the:

```
overlay currents:  True
```

Set the density of streamlines shown in these overlays using the option:

```
streamline density:  2.0
```

And if simulating a full environment, specify whether you want total extracellular current (True) or intracellular current (False) to be plotted in a current overlay:

```
plot total current:  True
```

Each plot or animation has settings allowing you to specify whether the plot appears (plot Vmem: True), if the color legend representing data values should be automatically scaled to the minimum and maximum values of the data set obtained in the simulation ('autoscale colorbar: True') or if you would like to supply your own minimum and maximum color legend limits ('autoscale colorbar: False'). State the desired maximum and minimum values in the subsequent fields for the plot of interest:

```
Vmem 2D:
    plot Vmem:  True
    autoscale colorbar:  False
    max val:  80.0
    min val:  -80.0
```

To get more control over the appearance of your plots you can change the colormap. A wide variety of colormaps are available to you, please see the Appendix (section 20), subsection 20.1 for all of the options. The 'default colormap' field specifies the colormap used to plot cell-specific data. The 'gj colormap' is the one used to color gap junctions connecting cells. To change the default or gj colormap, simply type in the name of one of the available colormaps (it is case-sensitive and must be written exactly as displayed in the examples) into the 'default colormap' or 'gj colormap' field of the configuration file. To reverse the color-scheme, simply add on a '_r' to the end of the colormap name. If you misspell a colormap, BETSE will crash with an appropriate error message.

Using the 'rainbow' colormap:

```
default colormap:  rainbow
```

Reversing the 'Blues' colormap by adding a '_r' to the end:

```
gj colormap:  Blues_r
```

It should be noted that there is a quirk to the color-bar scale, which appears whenever there is very little variation between elements of the 2D image (see Figure 33). To get rid of this effect, please set the 'autoscale colorbar' field of the figure to False.

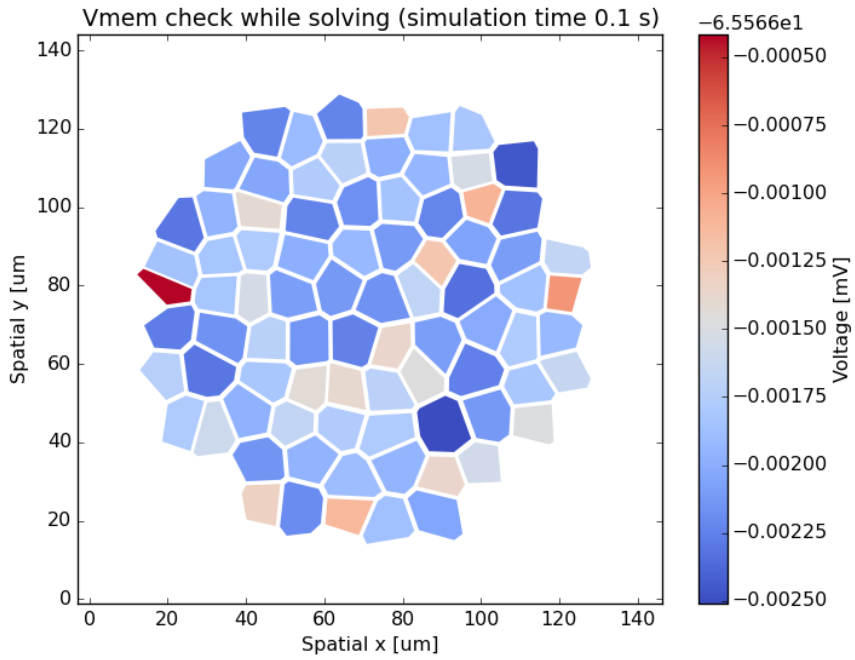


Figure 33: Color-bar rescales to confusing units when there is very little variation between elements of the 2D map.

18 Internal Use Only Parameters

The parameters listed under the *Internal Use Only* section of the configuration file are either well-known reference values (e.g. the free diffusion constants of ions in aqueous media), the result of many hours of trial-and-error during development to find suitable parameters, or factors allowing certain "specialty" or trial features to be easily turned on/off. Therefore, while altering these parameters may be exactly what you need to do to produce realistic output, keep in mind that doing so may create simulation instability (with ensuing BETSE crashes and error messages) or non-realistic output.

```
# ------------------------------------------------------------------------------------------------------------
# INTERNAL USE ONLY
# ------------------------------------------------------------------------------------------------------------
# While the following settings are available to you, please avoid editing them.
# Simulation instability may result from poor choice of these more sensitive parameters.
# BETSE requires these parameters for its internal use only.
internal parameters:

  # default free diffusion constants (cytoplasmic)

  Do_Na:  1.33e-9        # free diffusion constant sodium [m2/s]
  Do_K:  1.96e-9         # free diffusion constant potassium [m2/s]
  Do_Cl:  2.03e-9        # free diffusion constant chloride [m2/s]
  Do_Ca:  1.0e-10        # free diffusion constant calcium [m2/s]
  Do_M:  1.0e-9          # free diffusion constant mystery anchor ion [m2/s]
  Do_P:  0.0         # free diffusion constant protein [m2/s]

  # pump parameters
  alpha_NaK:  1.0e-7   # max rate Na-K-ATPase pump [m/mol*s] (1.0e-6 to 1.0e-12)

  alpha_Ca:  5.0e-8    # pump rate for calcium ATPase in membrane per unit surface area [m/mol*s]

  substances affect Vmem: True  # do ionic biochemicals, metabolites and gene products affect charge state?

  environment volume multiplier: 1.0  # level to multiply size of box-environment (applies for no ecm spaces only) 1.0

  membrane capacitance: 0.05 # ~0.05 to 0.1 cell membrane capacitance [F/m2]

  cell polarizability: 0.0  # cell relative dielectric constant (static, low frequency) (0.0 to 5.0e7)

  dielectric constant: 6.0  # dielectric constant of electrical double layer

  fast update ecm: False  # use a coarse (fast) or fine (slow) method to update between env and cell grids?

  sharpness env: 1.0  # Factor smoothing environmental concentrations, 0.0 max smoothing, 1.0 no smoothing

  sharpness cell: 0.5 # Factor smoothing cellular fields, 0.0 maximum smoothing, 1.0 no smoothing.

```

Figure 34: Internal use only parameters.

Of particular importance are the maximum rates of BETSE's default ion pumps — these can be altered or turned off completely depending on your needs.:

```
alpha_NaK: 1.0e-7
alpha_Ca:  1.0e-7
```

If the field:

```
substances affect Vmem:  True
```

Is true, all substances defined in “Custom Biomolecules” section of the network config file will affect the charge state of the cell (and therefore Vmem). Set this value to False if you wish to simulate a GRN that does not impact Vmem or other bioelectrical parameters.

Other parameters are described in the config file and should not need to be altered under normal use.

19 About BETSE

BETSE was designed and developed by Alexis Pietak and Cecil Curry for the Levin Lab at Tuft’s University, and we are grateful for funding support from the Paul Allen Discovery Award from the Paul G. Allen Frontiers Group.

Cecil (Brian) Curry is a software engineer and avid advocate and contributor to the open source software movement. Cecil is a computer science graduate of California Polytechnic State University in San Luis Obispo, and has worked as a software engineer/architect at *Apple*, on *Wall Street*, and at a Silicon Valley start-up. Cecil has contributed invaluably to BETSE by identifying the suit of open-source tools used to facilitate BETSE development, by implementing the command line interface (and other system-level tools), and with his tireless pursuit of cross-platform portable BETSE executables.

Alexis Pietak is a multi-disciplinary scientist with training in physics, biochemistry, engineering, and biomedical fields. Alexis received her PhD in physics from Queen’s University in Kingston, and has worked as a researcher at universities in New Zealand and Canada. Alexis is currently a member of the Allen Discovery Center at Tufts University, and devotes all of her time to BETSE development and associated research. Alexis first became interested in bio-electrochemical phenomena and their implied role in biological pattern formation after becoming aware of Dr Levin’s research in 2008. In 2013, Alexis set out to create an efficient, flexible, and bio-realistic model that would help explore bio-electrochemical phenomena in the context of spatio-temporal pattern formation, which eventually grew into the BETSE project.

BETSE relies heavily on open-source software tools developed for scientific computing, for which special mention and thanks goes out to:

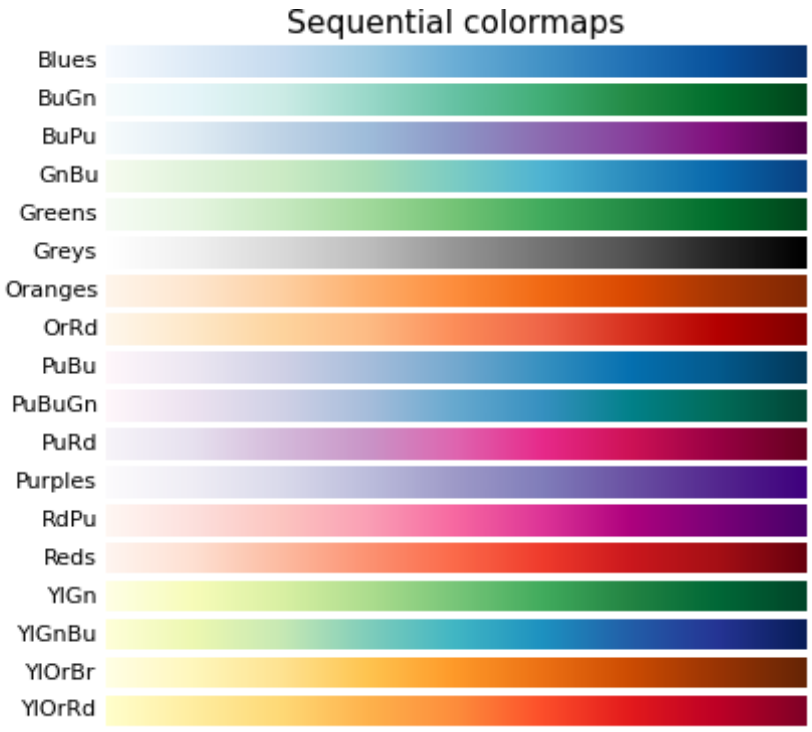
- → NumPy *the fundamental package for scientific computing in Python, especially essential for matrix math*
- → SciPy Library *a suite of additional tools for scientific computing in Python*
- → MatplotLib *a suite of plotting tools for data visualization*

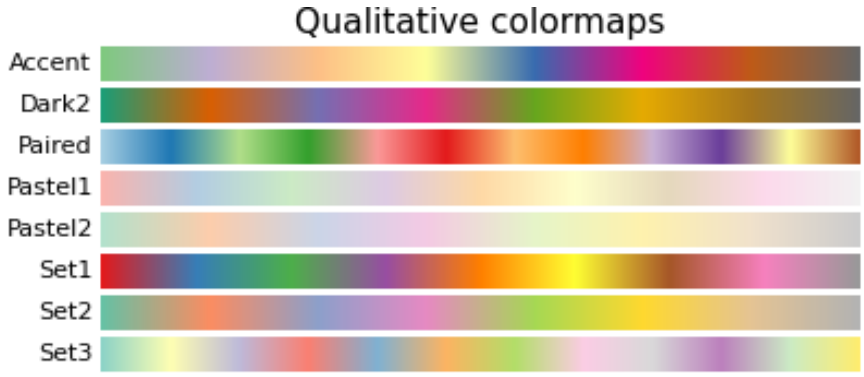
→ IPython *an advanced interactive console used extensively in BETSE development*

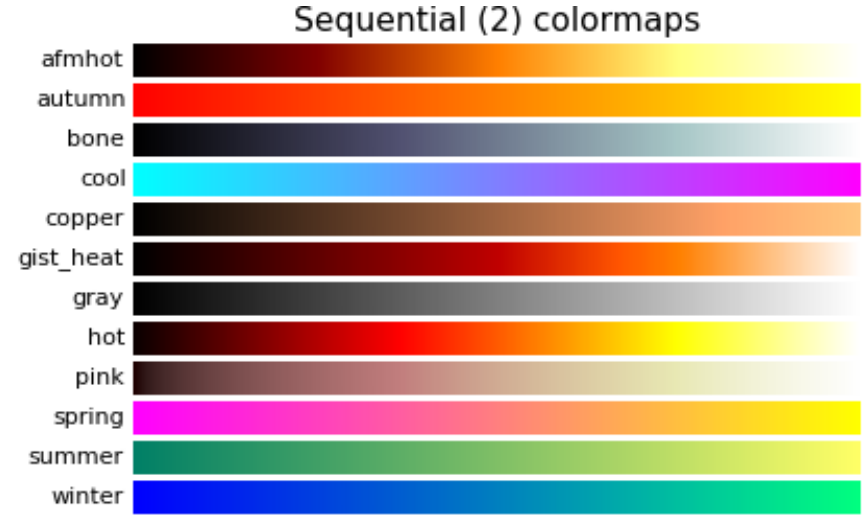
You can contact Alexis at: alexis.pietak@gmail.com and Cecil at leycec@gmail.com.

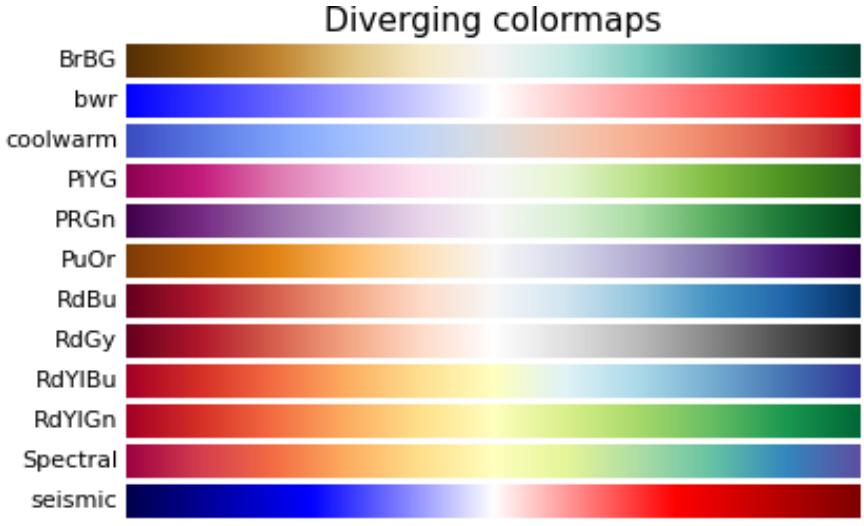
20 Appendix

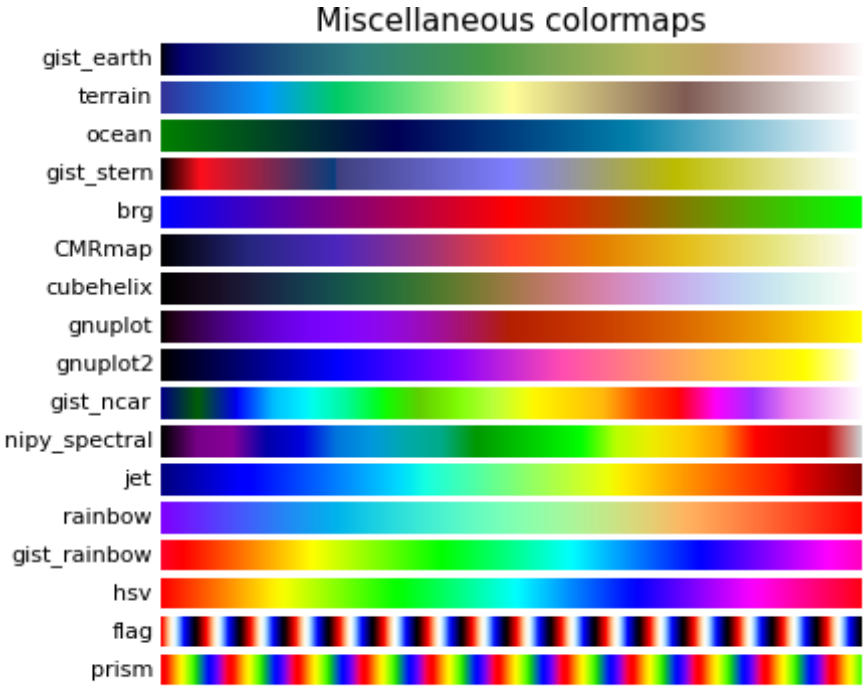
20.1 Available Colormaps




Qualitative colormaps
Accent
Dark2
Paired
Pastel1
Pastel2
Set1
Set2
Set3


Sequential (2) colormaps
afmhot
autumn
bone
cool
copper
gist_heat
gray
hot
pink
spring
summer
winter


Diverging colormaps
BrBG
bwr
coolwarm
PiYG
PRGn
PuOr
RdBu
RdGy
RdYlBu
RdYlGn
Spectral
seismic


Miscellaneous colormaps
gist_earth
terrain
ocean
gist_stern
brg
CMRmap
cubehelix
gnuplot
gnuplot2
gist_ncar
nipy_spectral
jet
rainbow
gist_rainbow
hsv
flag
prism