**PDE solver (Fortran)**

Input files:

* data\_new: parameter values
* input\_bio\_data: input parameters like the radius of the cell and reaction parameters. Most of the parameter values in here will get overwritten by the values in data\_new.
* input\_num\_data: defines the time step, max time, and how often you want to print the data to the terminal (however, I suppressed the printing – this can be turned back on in the main file)

Code files:

* mainYP\_2d.f: this contains the main loop
* const.f: reads in the input data
* forceYP\_2d.f: contains the RHS of the PDE with the spatial derivatives approximated by central difference
* ode.f: contains the ODE solvers to advance in time
* B1starGbnstar.f: calculates the Hill functions B1\* and Gbgn\*

Output files:

* da\_ind: stores the time points
* da\_xyp: contains 4 columns – x-coordinate, y-coordinate, ligand concentration, and alpha (angle from negative y-axis)
* da\_cont: stores all the system output

Other files:

* parame.h: defines global parameters to be used across different subroutines
* Makefile: this creates the executable file to run the code

To create/run the executable…

Navigate to the folder containing all these files in Terminal. Enter ‘make’ in the command line – this runs the Makefile. This will create an executable called ‘job’. Execute the job by entering ‘./job’ in the command line.

**PDE sampling**

Subfolders:

* master: contains Fortran code to run the PDE simulations.
* Optimization: contains the l1-Magic software package by Emmanuel Candes.

Matlab files:

* para\_range.m: sets parameter ranges in which to sampling
* parameter\_sets.m: generates uniform random samples within the parameter ranges given by para\_range.m. Stores the parameter sets (both within the ranges and normalized to [-1,1]) in text files, and generates copies of the master folder with the different parameter sets.
* extract.m: extracts the steady state data from the folders after the Fortran code has been executed.
* PFcalc.m: calculates the polarization factor for each parameter set based on the steady state C42a data.
* least\_sq5: sets up the linear system to solve for the polynomial coefficients. Stores the matrix in a text file called ‘Xmat’.
* solve\_eps: solves the linear system via l1-minimization, using the l1-Magic package stored in the Optimization folder. Saves the coefficients to a text file.
* sensitivitycalc5: computes the sensitivity coefficients for each parameter based on the 5th degree polynomial computed by solve\_eps.m.

Bash scripts:

* my\_script1: executes the Fortran code in each folder generated by parameter\_sets.m.
* my\_script\_matlab\_ls.sh: runs least\_sq5.m so that you can store the matrix on OSC.
* my\_script\_matlab.sh: runs solve\_eps.m to solve the linear system.

Helpful links for OSC:

<https://www.osc.edu/resources/getting_started>

<https://www.osc.edu/services/cluster_computing>

<https://www.osc.edu/supercomputing/batch-processing-at-osc/job-scripts>

<https://www.osc.edu/supercomputing/batch-processing-at-osc/job-submission>

<https://www.osc.edu/supercomputing/batch-processing-at-osc/monitoring-and-managing-your-job>

To check remaining Resource Units, enter ‘OSCusage’ in terminal.

To run Matlab in OSC, enter ‘module load matlab’ in terminal, then enter ‘matlab’. To quit matlab, enter ‘quit’.

**2D toy model code (MATLAB)**

Input files:

* combs10: This is a text file that contains all possible combinations of pairs of integers 0-10 whose sum is less than or equal to 10. This is used to generate the basis functions for the polynomial space.

Code files:

* main.m: this is the main file that you will run. It defines the parameters/ranges, basis polynomials, MCMC chain length, and number of samples for sensitivity analysis. Then it calculates the polynomial coefficients, computes the sensitivities, and runs MCMC.
* model\_eval.m: evaluates the ODE model given a set of parameters.
* get\_coeffs.m: computes the polynomial coefficients for each response function by calling poly\_build, and stores the coefficients in a global variable.
* poly\_build.m: Generates random samples, evaluates the model at these samples, and builds the linear system to solve for polynomial coefficients. If nr\_samples is greater than or equal to the dimension of the polynomial space, it uses least squares. Otherwise, it uses l1-minimization (codes to do this are stored in the Optimization folder).
* poly\_eval.m: evaluates a polynomial given a set of coefficients for the basis functions and a set of parameters.
* sensitivitycalc.m: For each response function, this computes the expected values of the partial derivatives of the polynomial. It displays the sensitivities as they are computed.
* run\_MCMC.m: Runs the MCMC parameter estimation. It stores the Markov chain in a text file called ‘MC’ and the corresponding outputs in ‘MC\_dat’.
* para\_hists.m: This creates two figures – (1) histograms of the 1D parameter distributions from the MCMC, and (2) a 2D histogram of the multi-parameter distribution.