Modeling the interactions between sodium channels provides insight into the negative dominance of certain channel mutations

Echrak Hichri, Zoja Selimi, Jan P. Kucera
Department of Physiology, University of Bern, Bern, Switzerland

This is the source MATLAB code generating (in MATLAB version 2015b) the results of an article that is presently under peer-review for publication.
The code allows replication of all the results in that article.

General instructions

Place all files in a dedicated folder.
Create a subfolder named "Figures" in this dedicated folder.
The following main scripts can directly be run (a summary description is provided below).
Details regarding the operation of the scripts and functions are presented as comments at the beginning of the scripts/functions as well as throughout the code.

Main scripts

Figure2_ReadClatotS8.m

Script generating the panels of Figure 2.
It uses the following auxiliary file:

ClatotS8-extractedData-Table-validated.txt
This file contains the L1 and L2 counts extracted from the vectorized PDF of Figure S8 (WT $Na_V1.5$ at -20 mV) in Clatot et al., in the following format:
column 1: integer multiples of the sampling period (30 kHz -> period is 0.033333 ms)
column 2: wild type, control, L1 (WTcL1) (integer number)
column 3: wild type, control, L2 (WTcL2) (integer number)
column 4: wild type, difopein, L1 (WTdL1) (integer number)
column 5: wild type, difopein, L2 (WTdL2) (integer number)

Clatot, J., Hoshi, M., Wan, X., Liu, H., Jain, A., Shinlapawittayatorn, K., Marionneau, C., Ficker, E., Ha, T., and Deschenes, I. (2017). Voltage-gated sodium channels assemble and gate as dimers. Nat Commun 8, 2077.

Figure3_MainScript_OneSim2state.m
Figure5_MainScript_OneSim3state.m
Figure8_MainScript_OneSimClancy_InteractionI.m
Figure9_MainScript_OneSimClancy_InteractionII.m
Figure10A_MainScript_Clancy_IV.m
Figure10B_MainScript_Clancy_IV.m
Figure11B_MainScript_Clancy_IV_WT.m
Figure11B_MainScript_Clancy_IV_Var.m
Figure12_MainScript_OneSimClancy_WTxVar_NoInteraction.m
Figure12_MainScript_OneSimClancy_WTxVar_InteractionI.m
Figure12_MainScript_OneSimClancy_WTxVar_InteractionII.m
Figure13_MainScript_Clancy_IV_Currents_NoInteraction.m
Figure13_MainScript_Clancy_IV_Currents_InteractionI.m
Figure13_MainScript_Clancy_IV_Currents_InteractionII.m
FigureS1_MainScript_OneSim2state.m
FigureS3_MainScript_OneSim3state.m
FigureS5_MainScript_Clancy_InteractionII.m
FigureS6_MainScript_Clancy_InteractionII_Gillespie.m
FigureS7_MainScript_FiveSimClancy_InteractionII.m
FigureS8_MainScript_OneSimClancy_InteractionII_dt10xShorter.m

Scripts generating the curves/panels of the corresponding figures.
The setup of the simulation parameters is commented in detail in the respective scripts.
For Figures 8-9, the scripts call the function *clancy1999.m*.
For Figures 10-13, the scripts also call the function *clancy1999mutL325R.m*.
At the end of the scripts for figures 3, 5, 8, 9, 12, S1, S3 and S8, the principal function *SymmInteractionAnalysisFigure.m* is called.
The principal function *SymmInteractionAnalysisFigure.m* is called repeatedly in the scripts for figures 10, 11B, 13, S5 and S7.
The principal function *FigureS6_MainScript_Clancy_InteractionII_Gillespie.m* calls the function *SymmInteractionAnalysisFigureGillespie.m* for figure S6.

Main scripts for the sensitivity analyses

To generate the plot in **Figure 4**, proceed as follows:
First, run the following script:
PrepareData_Figure4_SensiAnalysis_2state.m
This script generates the data to be analyzed and saves them as file *dataCO12.mat* for the 2-state model of Figure 3. This *.mat* file is already available, so this script can be skipped. However, the *.mat* file can be regenerated in this way if needed.
Then, run the following script
Figure4_SensiAnalysis_2state_PostProc.m

To generate the plot in **Supplementary Figure S2**, proceed as follows:
First, run the following script:
PrepareData_FigureS2_SensiAnalysis_2state.m
This script generates the data to be analyzed and saves them as file *dataCO21.mat* for the 2-state model of Figure S1. This *.mat* file is already available, so this script can be skipped. However, the *.mat* file can be regenerated in this way if needed.
Then, run the following script
FigureS2_SensiAnalysis_2state_PostProc.m

To generate the plot in **Figure 6**, proceed as follows:
First, run the following script:
PrepareData_Figure6_SensiAnalysis_3state.m
This script generates the data to be analyzed and saves them as file *dataCOIlinear.mat* for the 3-state model of Figure 5. This *.mat* file is already available, so this script can be skipped. However, the *.mat* file can be regenerated in this way if needed.
Then, run the following script
Figure6_SensiAnalysis_3state_PostProc.m

To generate the plot in **Figure S4**, proceed as follows:
First, run the following script:
PrepareData_FigureS4_SensiAnalysis_3state.m
This script generates the data to be analyzed and saves them as file *dataCOItriangular.mat* for the 3-state model of Figure 6. This *.mat* file is already available, so this script can be skipped. However, the *.mat* file can be regenerated in this way if needed.
Then, run the following script
FigureS4_SensiAnalysis_3state_PostProc.m

To generate the data shown in the plot in **Figure 7**, proceed as follows:
First, run the following script:
PrepareData_Figure7_SensiAnalysis_Clancy.m
This script generates the data to be analyzed and saves them as file *dataClancy.mat* for the wild type Clancy model. This *.mat* file is already available, so this script can be skipped. However, the *.mat* file can be regenerated in this way if needed.
This script uses the function *clancy1999.m*.
Then, run the following script
Figure7_SensiAnalysis_Clancy_PostProc.m
Note that in the final article figure, sensitivities are represented only for the subset of energy changes yielding sensitivities of 0.1 or more (in absolute value).

Remarks:

All the sensitivity analysis scripts called “PrepareData_... .m” repeatedly call the principal function *SymmInteractionAnalysisFigure.m* to run the simulations.

Note that in the final article figures, the sensitivity analysis for the energies of the composite states is shown *above* the sensitivity analysis for the energies of the barriers. In the MATLAB figures, the sensitivity analysis for the energies of the composite states is shown *below* the sensitivity analysis for the energies of the barriers.

The sensitivity analysis scripts also generate a set of figures (one per marker), each showing a set of plots (laid out as the Cartesian graph product of the model diagrams) of the logarithm of the marker vs the energy change (from -2 kT to +2 kT), along with the regression line (in blue) as well as the numerical values of the regression slope and squared correlation coefficient.

Main functions

clancy1999.m
Generates the Clancy model “Q”-matrix for a given membrane potential for the wild type channel.

clancy1999mutL325R.m
Generates the Clancy model “Q”-matrix for a given membrane potential for the L325R variant channel.

SymmInteractionAnalysisFigure.m
This is the principal function used in all simulations.
It runs a composite model of two identical or non-identical Markovian ion channel models. It is assumed that the two base models have the same state diagram, but the rate coefficients of the two base models can be different.
Interactions between the two channels can be incorporated as changes in the free energies of the composite states and/or as changes in the free energies of the barriers between composite states. It is assumed that the interaction is symmetric, i.e., that the action of the first channel on the second is the same as the action of the second on the first.
The function runs a control simulation without interactions using both a deterministic algorithm and a stochastic algorithm.
The function then runs a simulation with interactions using again both a deterministic algorithm and a stochastic algorithm.
The function performs various analyses, plots the results, and returns analysis markers in the structures N, I, F (for details about the markers and these structures, consult the MATLAB file).
Further details on how to configure the parameters of the models can be found, e.g., in the script *Figure3_MainScript_OneSim2state.m.*
Details about the operation of this function are presented at the beginning of the MATLAB file (input parameters and outputs) as well as throughout the file.

Auxiliary functions

RaiseWellSym.m
Raises the energy level (energy well) of a composite state in a symmetric manner (i.e., for example, if the energy of CO is raised, then OC is also raised by the same amount).
For further details, see comments in the MATLAB code.

ixCompound.m
Obtains the indices of composite states in the Q matrix of the composite (compound) model.
For further details, see comments in the MATLAB code.

QintE.m
Changes the energy of a composite state in a composite model Q by multiplying all outgoing rates by the same factor. This corresponds to raising the energy of that state by kT*log(factor).
For further details, see comments in the MATLAB code.

RaiseBarrSym.m
Raises the energy barrier between two composite states in a symmetric manner
(i.e., for example, if the energy of the barrier between CC and CO is raised, then the energy of the barrier between CC and OC is also raised by the same amount).
For further details, see comments in the MATLAB code.

QintB.m
Changes energy of a barrier between composite states in a composite model Q by multiplying forward and reverse rates by the same factor. This corresponds to raising the energy of that barrier by kT*log(factor).
For further details, see comments in the MATLAB code.

f_StateSequence.m
Core function running a stochastic simulation using the matrix exponential algorithm for a model (single channel or composite of two channels) given by matrix Q.
For details regarding the input parameters and the output, see comments in the MATLAB code.

f_DetermPSequence.m
Core function running a deterministic simulation using the matrix exponential algorithm for a model (single channel or composite of two channels) given by matrix Q.
For details regarding the input parameters and the output, see comments in the MATLAB code.

findSS.m
Finds the steady state probability distribution “s” for a Markovian model given by its “Q”-matrix by solving the linear system Qs=0 under the constraint that all elements of “s” sum to 1.
For further details, see comments in the MATLAB code.

findLongestSeq.m
Finds the longest sequence of ones in a binary vector (one-dimensional array) consisting of zeros or ones.
For further details, see comments in the MATLAB code.

extractFields.m
Extract field names and corresponding values from a MATLAB structure
For further details, see comments in the MATLAB code.

New functions created during revision of the manuscript in response to reviewer comments

f_StateSequenceGillespie.m
Core function running a stochastic simulation using Gillespie’s algorithm for a model (single channel or composite of two channels) given by matrix Q.
The function has the same syntax (same input parameters) and the same structure of its output as *f_StateSequence.m* and can be used instead.
For details regarding the input parameters and the output, see comments in the MATLAB code.

SymmInteractionAnalysisFigureGillespie.m
Same function as *SymmInteractionAnalysisFigure.m*, but using Gillespie’s algorithm instead of the matrix exponential algorithm in stochastic simulations. It is the same function, except that it uses *f_StateSequenceGillespie.m* instead of *f_StateSequence.m*.