Improving the In Silico Assessment of Proarrhythmia Risk by Combining hERG (Human Ether-à-go-go-Related Gene) Channel–Drug Binding Kinetics and Multichannel Pharmacology

Zhihua Li, PhD; Sara Dutta, PhD; Jiansong Sheng, PhD; Phu N. Tran, PhD; Wendy Wu, PhD; Kelly Chang, PhD; Thembi Mdluli, PhD; David G. Strauss, MD, PhD; Thomas Colatsky, PhD

Background—The current proarrhythmia safety testing paradigm, although highly efficient in preventing new torsadogenic drugs from entering the market, has important limitations that can restrict the development and use of valuable new therapeutics. The CiPA (Comprehensive in vitro Proarrhythmia Assay) proposes to overcome these limitations by evaluating drug effects on multiple cardiac ion channels in vitro and using these data in a predictive in silico model of the adult human ventricular myocyte. A set of drugs with known clinical torsade de pointes risk was selected to develop and calibrate the in silico model.

Methods and Results—Manual patch-clamp data assessing drug effects on expressed cardiac ion channels were integrated into the O’Hara–Rudy myocyte model modified to include dynamic drug–hERG channel (human Ether-à-go-go-Related Gene) interactions. Together with multichannel pharmacology data, this model predicts that compounds with high torsadogenic risk are more likely to be trapped within the hERG channel and show stronger reverse use dependency of action potential prolongation. Furthermore, drug-induced changes in the amount of electronic charge carried by the late sodium and L-type calcium currents was evaluated as a potential metric for assigning torsadogenic risk.

Conclusions—Modeling dynamic drug–hERG channel interactions and multi-ion channel pharmacology improves the prediction of torsadogenic risk. With further development, these methods have the potential to improve the regulatory assessment of drug safety models under the CiPA paradigm. (Circ Arrhythm Electrophysiol. 2017;10:e004628. DOI: 10.1161/CIRCEP.116.004628.)

Key Words: biomarkers ■ ion channels ■ torsade de pointes

Current regulatory guidelines focus on 2 surrogate biomarkers to assess the risk of drug-induced torsade de pointes (TdP): effects on the hERG (human Ether-à-go-go-Related Gene) channel and prolongation of the corrected QT interval (QTc) in humans. Although this paradigm has largely eliminated the entry of new torsadogenic drugs into the market, these 2 biomarkers have proven to be poor surrogates for the primary end point of clinical concern, TdP. A key issue is that hERG channel block may be offset or exacerbated by concomitant blockage of other cardiac currents. Numerous examples exist of drugs showing similar QTc prolongation effects but different proarrhythmic potential. Consequently, the CiPA (Comprehensive in vitro Proarrhythmia Assay) has been proposed as a new regulatory paradigm that examines drug effects on multiple cardiac ion channels in vitro and integrates these pharmacology data into an in silico model of the human adult ventricular cardiomyocyte to assess TdP risk. A set of 28 drugs with known levels of clinical risk was identified for evaluation in the CiPA paradigm and categorized by expert consensus into 3 clinical TdP risk groups (high, intermediate, and low/no risk). Of these, an initial training set of 12 compounds covering a wide range of electrophysiological properties was selected for use in model calibration, and the remainder reserved for validation studies to be conducted at a later time.

Previous attempts to assess TdP risk using in silico models usually only separated drugs into high- and low-risk categories. The few attempts where additional categories were used could not distinguish between high- and intermediate-risk CiPA training compounds. Specifically, in all previous classification models, simulations with dofetilide (high risk) and cisapride (intermediate risk) produced similar results and were assigned the same proarrhythmia risk. One confound is using half-maximal blocking concentration (IC50) in these
WHAT IS KNOWN
- Current regulatory guidelines focus on 2 biomarkers to assess the risk of drug-induced TdP: effects on the hERG (human Ether-a-go-go-Related Gene) channel and prolongation of the corrected QT interval (QTc) in humans.
- These 2 markers are poor surrogates for TdP, and a new paradigm has been proposed to integrate in vitro multi-channel pharmacology into an in silico model to assess TdP risk directly as an end point.
- Previous attempts in line with this proposed paradigm shifted used IC50 to represent drug effects and ignored dynamic time- and state-dependent drug–channel interaction.

WHAT THE STUDY ADDS
- A new model that combines dynamic drug–hERG interactions and multichannel pharmacology was developed.
- A metric based on drug-induced change in inward currents was established from this model and shown to be able to classify TdP risk levels in a mechanistic manner.

Simulations and ignoring dynamic time- and state-dependent drug–channel interactions,10,11 dofetilide and cisapride have vastly different unbinding rates because of the tendency of dofetilide but not cisapride to be trapped within the hERG channel.12 Although it has been shown theoretically that trapped drugs may have more proarrhythmia liability compared with nontrapped drugs with similar hERG-blocking potency,13,14 no attempt has been made to classify TdP risk using experimentally derived data on drug–channel interactions in an in silico model.

Here, we report development of a novel hERG dynamic model and the characterization of drug–hERG channel interactions for the 12 CiPA training compounds. Data were obtained using a voltage-clamp protocol that captured both block development and the tendency for drugs to be trapped within the hERG channel. We demonstrate that this approach can identify mechanistic differences and quantitative metrics that can separate drugs based on relative clinical TdP risk.

Methods

Cell Line, Drugs, and Electrophysiology
Experiments were performed on a HEK293 cell line that stably expresses hERG1a subunit (provided by Dr Gail Robertson, University of Wisconsin–Madison).15 D,L-Sotalol hydrochloride, cisapride monohydrate, dofetilide, azimilide dihydrochloride, chlorpromazine hydrochloride, and dimethyl sulfoxide were purchased from Sigma (St Louis, MO). All other drugs were purchased from Tocris Bioscience (Minneapolis, MN). All compounds are listed in Table 1. All electrophysiology experiments were conducted at 37°C. The detailed experimental procedure can be found in Methods section in the Data Supplement.

Dynamic hERG-Binding Model
The dynamic hERG-binding model contained a physiological component representing channel gating and a pharmacodynamic component representing drug binding. The physiological component is based on a recently published temperature-dependent hERG model (Figure 1)16 and further optimized to recapitulate the behavior of the native rapid delayed rectifier potassium current (IKr) (see Figure 4 for the current trace used). The pharmacodynamic component has 3 drug-bound states: open-bound (O*), closed-bound (C*), and open-inactivated-bound (IO*). Ku is the unbinding reaction (O*→O) rate while the binding reaction rates (O*→IO* and IO*→IO+) are equal to KuEmax(D), where Emax is a sigmoid model describing the concentration response of each drug and D is the drug concentration in nanomoles per liter (nM). The Emax model is defined as

\[
\text{Emax} = \frac{\text{EC}50}{\text{EC}50 + D^{K_{DD}^n n}},
\]

where EC50 is the concentration where half maximum effect is achieved. When EC50>>D, the Emax model becomes a linear model \(\text{K}_{\text{DD}^n n} \times \text{D}^{n}\); thus, drugs with different concentration–response relationships can be modeled by the same equation. The trapping rate (Kt) of channel closing with drug bound (O*→C* and O*→C*) was manually fixed at 3.5×10⁻³ ms⁻¹ to achieve the best separation between trapped and nontrapped drugs and the lowest fitting error (Figure XIII in the Data Supplement). The rate of channel opening when drug is bound (C*→O* and C*→IO*) is equal to Kt*X(V), where X is based on the steady-state gate activation function for IKr in O’Hara–Rudy (ORD) model17 by replacing the constant −8.337 with a parameter Vhalf (see below) and V is the membrane potential in millivolts (mV). X(V) is defined as 1/(1+exp(-(V–Vhalf)/6.789)), where Vhalf is the membrane voltage at which half of drug-bound channels are open. The IO* to IO transition rate is determined by microscopic reversibility.18 The detailed fitting procedure and the mathematical formulations of the model can be found in Methods section and Table I in the Data Supplement, respectively. The parameterization was repeated for each drug from different starting points (random seeds) to ensure a solution near the global optimum was reached (Table II in the Data Supplement).

IKr-Dynamic ORd Model
The formulation of IKr in the ORd model (endocardial) was replaced by the dynamic hERG model, with the IKr conductance adjusted to 0.0418 ms/μF to better replicate original IKr behavior (Figure 4). For all simulations, 1000 pacing cycles were used to achieve steady state. The metric calculated for each drug is calculated as

\[
\text{cqInward} = \frac{\text{AUC}_{\text{qu}}}{\text{AUC}_{\text{control}}},
\]

where AUC is the integrated area under the curve of the native rapid delayed rectifier potassium current (INaL and L-type calcium (ICaL) current traces during

Table 1. The 12 CiPA Training Compounds

<table>
<thead>
<tr>
<th>Drug</th>
<th>Indication</th>
<th>CiPA TdP Category</th>
</tr>
</thead>
<tbody>
<tr>
<td>Quinidine</td>
<td>Antiarrhythmic</td>
<td>High</td>
</tr>
<tr>
<td>Bepridil</td>
<td>Angina</td>
<td>High</td>
</tr>
<tr>
<td>Dofetilide</td>
<td>Antiarrhythmic</td>
<td>High</td>
</tr>
<tr>
<td>Sotalol</td>
<td>Antiarrhythmic</td>
<td>High</td>
</tr>
<tr>
<td>Chlorpromazine</td>
<td>Antipsychotic</td>
<td>Intermediate</td>
</tr>
<tr>
<td>Cisapride</td>
<td>Gastrokinetic</td>
<td>Intermediate</td>
</tr>
<tr>
<td>Terfenadine</td>
<td>Antihistamine</td>
<td>Intermediate</td>
</tr>
<tr>
<td>Ondansetron</td>
<td>Antimimetic</td>
<td>Intermediate</td>
</tr>
<tr>
<td>Diltiazem</td>
<td>Hypertension/angina</td>
<td>Low/no risk</td>
</tr>
<tr>
<td>Mexiletine</td>
<td>Antiarrhythmic</td>
<td>Low/no risk</td>
</tr>
<tr>
<td>Ranolazine</td>
<td>Angina</td>
<td>Low/no risk</td>
</tr>
<tr>
<td>Verapamil</td>
<td>Hypertension/angina</td>
<td>Low/no risk</td>
</tr>
</tbody>
</table>
steady-state action potential simulation with (_drug) or without (_control) drugs. More details can be found in Methods section in the Data Supplement.

Results

Dynamic hERG-Binding Model

Current understanding of drug–hERG interactions assumes that most drugs only bind when the channel opens.\(^{19}\) When the channel closes on repolarization, some compounds can readily unbind, whereas others will be trapped within the channel and require channel reopening to unbind.\(^{11,19}\) To provide a unified framework for modeling drug–hERG dynamics, we complemented our previously published hERG gating model\(^ {16}\) with 3 drug-bound states: open-bound (O\(^ *\)), inactivated open-bound (IO\(^ *\)), and closed-bound (C\(^ *\)). The closing of drug-bound channels (from O\(^ */IO\(^ *\)) to C\(^ *\)) is affected by bound drug to varying degrees, represented by the drug-specific parameter \(V_{\text{half-trap}}\) (Table 2). The concept of \(V_{\text{half-trap}}\) is taken from the modeling of activation/deactivation of IKr in the ORd model,\(^ {17}\) where a \(V_{\text{half}}\) parameter represents the membrane voltage at which half of the channels are closed at steady state. In our drug-binding model, \(V_{\text{half-trap}}\) represents the membrane voltage at which half of the drug-bound channels are closed. A negatively shifted \(V_{\text{half-trap}}\) (compared with the original \(V_{\text{half}}\) value of \(-8.337\) mV) indicates that the drug impairs the ability of the channel to close and thus is less likely to be trapped.

Drug-binding parameters (Table 2) were estimated using a modified Milnes et al\(^ {12}\) protocol containing repeated episodes of depolarization to 0 mV from a holding potential of \(-80\) mV (Figure 1B). Representative current traces from this protocol during the 0-mV step before and after application of 10 nmol/L cisapride are shown in Figure 1C. The Milnes protocol was designed to differentiate between trapped and nontrapped drugs\(^ {12}\) because the former will have little relief of block during the 15-second interpulse interval at \(-80\) mV, resulting in pronounced reductions of peak current in later episodes. We found a similar pattern when this protocol was applied to the CiPA training compounds. For example, dofetilide,\(^ {20}\) bepridil,\(^ {11}\) and terfenadine\(^ {11}\) are known to be trapped, and each of these drugs significantly reduced peak current at the tenth episode compared with the first (Figure 2A). By contrast, another set of 3 CiPA training drugs (cisapride,\(^ {11}\) verapamil,\(^ {21}\) and ranolazine\(^ {22}\)) are known to be nontrapped, and these drugs produced only minor changes in peak current across successive depolarizations (Figure 2B). Importantly, nearly all of the compounds tested, including those well-known nontrapped drugs, show a gradual reduction of peak current, supporting the assumption that all drugs are trapped to varying degrees.

The current traces and the trapping phenotype from the modified Milnes protocol were used to estimate drug-binding parameters for the 12 CiPA training compounds, each using 4 concentrations (see Methods section in the Data Supplement for details). The fitting of the model across all episodes and the estimated drug-binding parameters for all drugs can be

---

**Figure 1.** The model and voltage protocol used to study drug–hERG (human Ether-à-go-go-Related Gene) dynamic interactions. A. The structure of the hERG model. The left part (physiological component) is based on the hERG model published earlier.\(^ {16}\) C1 and C2 are closed states, O is open state, and the corresponding inactivated states are IC1, IC2, and IO. The right part (pharmacodynamic component) assumes there are 3 drug-bound states: open bound (O\(^ *\)), inactivated open bound (IO\(^ *\)), and closed bound (C\(^ *\)). Drug may be trapped in the C\(^ *\) state. The transition rates between different drug-bound states are listed above the arrows. See Methods section for details. B. The voltage command used to estimate drug-binding parameters, modified from the study by Milnes et al.\(^ {12}\) Top. The voltage steps of one episode; 10 such episodes are applied consecutively for control (before drug wash-in) and drug-treated (after drug wash-in) conditions, respectively. Bottom. The no-pulse protocol applied during drug wash-in. C. A typical current trace during the 0-mV step obtained before (control) and after applying 10 nmol/L cisapride.
found in Figures I through XII in the Data Supplement and Table 2, respectively. Figure 3 shows representative fitting of the first and tenth episodes for 3 representative drugs: dofetilide (Figure 3A), terfenadine (Figure 3B), and verapamil (Figure 3C). These drugs show different amplitudes of initial block increase (corresponding to the peak current reductions (Figure 3A), terfenadine (Figure 3B), and verapamil (Figure 3C). This sug-

### IKr-Dynamic ORd Model

The dynamic hERG model was integrated into the ORd adult human left ventricular cardiomyocyte model\(^17\) (see Methods section). To ensure this integration did not change the physiological behavior of the ORd model, the dynamic hERG model was optimized to replicate the ORd IKr behavior under drug-free conditions. As shown in Figure 4A, the shape of the hERG current under an action potential (AP) waveform voltage-clamp protocol in the dynamic hERG model is close to that of IKr in the original ORd model. The resulting IKr-dynamic ORd model has the same steady-state rate dependence as the original ORd model, as shown by identical AP duration (APD)–cycle length (CL) relationships (Figure 4B). The simulated AP waveforms are also nearly identical in the original and IKr-dynamic ORd models at different CLs (Figure 4C).

However, we expected the APD–CL relationship to differ between the 2 models when drug was present. Several lines of evidence have suggested that drug-induced increases in ventricular APD have an intrinsic property of reverse use-dependency (RUD), meaning that the degree of APD prolongation will be more pronounced at slower heart rates.\(^{23,25}\) But other rate-dependent mechanisms could modify this intrinsic property of RUD, resulting in different, drug-specific amounts of RUD.\(^{23}\) It is conceivable that, in contrast to trapped drugs, hERG blockers without significant trapping may show forward use-dependent blocking at the channel level\(^1\) (ie, less block at slow heart rates) because of quick channel unbinding at diastolic membrane potentials, which may attenuate the magnitude of RUD at the myocyte level.\(^{26}\) To test this hypothesis, we used the hERG dynamic binding parameters from Table 2 and published IC50 values\(^27\) against 6 of the major cardiac ion channels—KvLQT1/mink (slow delayed rectifier potassium current or IKs), Nav1.5 (peak/late sodium currents or INa/L), Kir2.1 (inward rectifier potassium current or IK1), Kv4.3 (transient outward potassium current or Ito), and Cav1.2 (ICaL)—obtained using manual patch clamp (Table 3) to calculate APD at 90% repolarization

<table>
<thead>
<tr>
<th>Drug</th>
<th>(K_{\text{m}}) max</th>
<th>(A_u) ms(^{-1})</th>
<th>EC50(^{\text{c}}), nmol/L</th>
<th>(N)</th>
<th>V(_{\text{half-trap}}), mV</th>
<th>Free (C_{\text{m}}) max, nmol/L</th>
</tr>
</thead>
<tbody>
<tr>
<td>Quinidine</td>
<td>5770</td>
<td>0.01</td>
<td>1.00E+06</td>
<td>0.8311</td>
<td>−64.87</td>
<td>3237</td>
</tr>
<tr>
<td>Bepridil</td>
<td>37350000</td>
<td>0.0001765</td>
<td>1.00E+09</td>
<td>0.9365</td>
<td>−54.93</td>
<td>33</td>
</tr>
<tr>
<td>Dofetilide</td>
<td>1.00E+08</td>
<td>1.79E−05</td>
<td>5483000000</td>
<td>0.9999</td>
<td>−1.147</td>
<td>2</td>
</tr>
<tr>
<td>Sotalol</td>
<td>2403</td>
<td>0.01985</td>
<td>9619000</td>
<td>0.7516</td>
<td>−115</td>
<td>146904</td>
</tr>
<tr>
<td>Chlorpromazine</td>
<td>206000</td>
<td>0.03866</td>
<td>56770000</td>
<td>0.8871</td>
<td>−14.57</td>
<td>38</td>
</tr>
<tr>
<td>Cisapride</td>
<td>9.997</td>
<td>0.0004161</td>
<td>42.06</td>
<td>0.9728</td>
<td>−199.5</td>
<td>2.6(^5)</td>
</tr>
<tr>
<td>Terfenadine</td>
<td>9884</td>
<td>8.18E−05</td>
<td>41380</td>
<td>0.65</td>
<td>−77.49</td>
<td>4(^9)</td>
</tr>
<tr>
<td>Ondansetron</td>
<td>33540</td>
<td>0.02325</td>
<td>99500000</td>
<td>0.8874</td>
<td>−82.11</td>
<td>139</td>
</tr>
<tr>
<td>Diltiazem</td>
<td>251</td>
<td>0.2816</td>
<td>1.00E+06</td>
<td>0.9485</td>
<td>−90.89</td>
<td>122(^4)</td>
</tr>
<tr>
<td>Mexiletine</td>
<td>9.996</td>
<td>0.09967</td>
<td>23080000</td>
<td>1.304</td>
<td>−86.26</td>
<td>4129(^9)</td>
</tr>
<tr>
<td>Ranolazine</td>
<td>55.84</td>
<td>0.01929</td>
<td>1472000</td>
<td>0.95</td>
<td>−94.87</td>
<td>1948.2(^{27})</td>
</tr>
<tr>
<td>Verapamil</td>
<td>46460</td>
<td>0.0007927</td>
<td>91840000</td>
<td>1.043</td>
<td>−100</td>
<td>81(^1)</td>
</tr>
</tbody>
</table>

For most drugs, the estimated EC50>>concentrations tested, and the \(E_{\text{max}}\) model used to describe dose–response relationship essentially becomes a linear model (see Methods section). For other drugs like cisapride and diltiazem, the estimated EC50s are not significantly higher than the tested concentrations (EC50 <10-fold of estimated IC50), suggesting there is indeed a sigmoidal dose–response relationship. The free \(C_{\text{m}}\)max values are either directly taken from the referenced article or, in the case of ondansetron, calculated using reported total \(C_{\text{m}}\) and protein-binding data from FDA drug labels. CiPA indicates Comprehensive in vitro Proarrhythmia Assay; \(C_{\text{m}}\)max, maximum therapeutic plasma concentrations; EC50, concentration where half maximum effect is achieved; FDA, Food and Drug Administration; hERG, human Ether-à-go-go-Related Gene; IKr, rapid delayed rectifier potassium current; \(K_{\text{m}}\), maximum drug effect at saturating concentrations; \(A_u\), unbinding reaction rate; and \(V_{\text{half-trap}}\), membrane voltage at which half of drug-bound channels are open.
In Silico Assessment of Proarrhythmia Risk

(APD90) at 4 different CLs for all 12 CiPA training compounds at their corresponding therapeutic free plasma concentrations (Cmax; Table 2). A linear regression between APD90 and CL was used to estimate the degree of RUD similar to the linear regression between QT and RR used in clinical studies. As shown in Figure 5A, the slope of the regression line (R) was significantly increased after 2 nmol/L dofetilide treatment compared with control (from 0.053 to 0.077), suggesting that the trapped drug dofetilide caused RUD at therapeutic concentrations, consistent with clinical QT studies. In contrast, the cisapride (2.6 nmol/L) regression line is largely parallel to that of control (slope slightly decreased from 0.053 to 0.045), suggesting that intrinsic RUD was completely masked by this drug’s forward use-dependency at the channel level, similar to other nontrapped drugs. Figure 5B lists the ratio between Rs in the presence and absence of each of the CiPA training compounds as an indicator of the degree of RUD at each drug’s Cmax. It is evident that compounds in the high TdP risk group tend to have greater degree of RUD (higher ratio R_drug/R_control). The same simulations were conducted using the original ORd model, where all drug–channel interactions were represented by IC50s (Table 3). As expected, in the absence of frequency-dependent blocking to counteract intrinsic RUD, both dofetilide and cisapride have higher R values than control (Figure 5C), and many compounds across different TdP risk groups now have RUD (R_drug/R_control >1) at clinical concentrations (Figure 5D). A direct comparison of the degree of RUD with and without considering hERG-binding dynamics (Figure 5E) suggests that for most of the high-risk compounds, the inclusion of hERG-blocking dynamics (with high degree of trapping) increases the predicted degree of RUD, whereas the effect is opposite for most of intermediate and low-/no-risk drugs (with low degree of trapping). In addition, we found that non-hERG blocking can contribute to determining the degree of RUD as well because the ICaL- or INaL-blocking activities can decrease the degree of RUD of low-/no-risk drugs (Figure 5F), consistent with a previous report that these 2 currents have opposite effects compared with IKr in causing RUD.

Candidate Risk Metrics

Next, we began a systematic search for a quantitative TdP risk metric using the IKr-dynamic ORd model. The ideal metric would enable a clear separation of the 3 risk categories, with metric values indicating the probability of eliciting an early afterdepolarization (EAD), a mechanistic precursor to TdP. Using the above criteria to evaluate previously reported candidate metrics, including those thoroughly tested by Mirams et al., we found none performed adequately in classifying relative TdP risk (data not shown). Because TdP (and EAD) occurs only under certain conditions, we then...
performed a focused screening targeting those model outputs when pacing the model at 30 beats per minute (CL 2000 ms) because bradycardia is one well-known TdP risk factor. For each drug, a range of concentrations from 1× to 25× $C_{max}$ was used to examine the dose–response relationship of the candidate metrics (see Table 3 legend for details). One of the model outputs, cqInward, representing the drug-induced change in the amount of electronic charge carried by INaL ($qNaL$) and ICaL ($qCaL$), shows a promising feature of separating drugs with different TdP risks. As illustrated in Figure 6, at high concentrations (44 nmol/L or 22× free $C_{max}$), dofetilide caused an EAD (Figure 6A, left, dotted line), and the associated $qNaL$ and $qCaL$ (Figure 6A, middle and right, AUC of dotted lines) are dramatically increased compared with control. Note that dofetilide is a relatively pure outward current blocker ($IKr/hERG IC50$ at least 50-fold lower than that of inward currents INaL and ICaL; Table 3), so its effects on inward currents are primarily through extended channel opening because of delayed repolarization caused by its blocking of outward currents. In contrast, the potent IKr and INaL blocker ranolazine can cause severe APD prolongation but not EADs (Figure 6B, left), even at >22× free $C_{max}$ (50 μmol/L). The prolonged APD still increases the charge carried by inward currents, as evidenced by the increase (1.3× of control) in $qCaL$ (Figure 6B, right). However, for INaL, dofetilide’s direct blocking effects overshadow the effects of extended opening, causing a more significant decrease (0.24× of control) in $qNaL$ (Figure 6B, middle). This results in an overall drug-induced reduction of inward charge during the AP. The full profile of concentration-dependent cqInward for all CiPA training compounds can be found in Figure 6C. All 4 compounds in the high-risk group show a rapid concentration-dependent increase of cqInward, with 3 of them (dofetilide, bepridil, and quinidine) reaching a threshold dose, where cqInward is close to 1.2 (stars in Figure 6C), after which an EAD occurs. Sotalol did not elicit EADs at the highest dose tested (25× free $C_{max}$), but its cqInward values are close to the threshold at the highest dose. For the 4 compounds in the medium-risk group (cisapride, chlorpromazine, ondansetron, and terfenadine), their cqInward values change slowly as dose increases, resulting in largely flat curves and no EADs at the highest doses. The 4 drugs in the low-risk group (ranolazine, mexiletine, diltiazem, and verapamil) actually displayed decreased cqInward values at higher
doses, suggesting that their blocking of inward currents has a protective effect which counteracts any potential reduction of outward currents. Thus, the 3 CiPA TdP risk groups are separated in a mechanistic and concentration-dependent manner.

In contrast, the original ORd model with IC50-based conductance scaling for all channels show more mixed results using this metric, especially for cisapride (Figure 7). In addition, the IKr-dynamic ORd model, but not the original ORd model, was able to correctly predict the TdP categories for all 12 CiPA compounds in a leave-one-out validation study (Table III in the Data Supplement).

**Discussion**

This study supports the need to model the dynamic nature of hERG channel pharmacology, especially trapping, to explain why some drugs with similar hERG-blocking potency have different proarrhythmic liabilities. Together with multichannel pharmacology, our IKr-dynamic ORd model was able to stratify all CiPA training compounds into their corresponding TdP risk groups.

A unique feature of our approach is the use of a continuous parameter (Vhalf-trap) to represent the tendency of a drug to be trapped on hERG channel closing. Although the Vhalf-trap parameters appear consistent with previous classifications of compounds as either trapped or nontrapped by block recovery experiments, as in the case of dofetilide, bepridil, cisapride, and verapamil, some discrepancies exist. For instance, quinidine and terfenadine were considered nontrapped and trapped, respectively, in previous block recovery experiments; however, quinidine seems to be more trapped than terfenadine based on their Vhalf-trap parameters (Figure 3D). The differences in temperatures (37°C versus room temperature) and expression systems (HEK cells versus Xenopus oocytes) may be one reason for the discrepancy. Alternatively, the peak current measurements may be inadequate to constrain the trapping parameters for all drugs. Identifying and incorporating more trapping phenotypes may help the model to estimate trapping potential more accurately.

In the IKr-dynamic ORd model, the tendency of CiPA high-risk compounds to be trapped is associated with more significant RUD at therapeutic concentrations (Figure 5A). The original ORd model, by contrast, showed a much more uniform degree of RUD across all risk categories. Directly correlating these modeling results with experimental data proves difficult. Even though many drugs’ ability to cause RUD was extensively surveyed, most of these experiments were animal studies using drug concentrations much higher than clinical observations. Sometimes, different species or tissue preparations led to opposite conclusions. However, as RUD has been suggested as one of the features for torsadogenic drugs and bradycardia is one of the well-established risk factors for TdP, the prediction from our IKr-dynamic ORd model that CiPA high-risk compounds tend to cause more significant RUD on cardiomyocytes is likely the mechanism behind the different torsadogenic potentials for some drugs with similar channel-blocking potencies (eg, dofetilide versus cisapride).

![Figure 4. Integration of the dynamic hERG (human Ether-à-go-go-Related Gene) model into O’Hara–Rudy (ORd) cardiomyocyte model. A. An action potential (AP; cycle length 2000 ms) waveform voltage-clamp protocol (upper) simulation was applied to both the original ORd and dynamic hERG model, and the resulting rectifier potassium current (IKr; bottom, circles) is compared with the hERG current (bottom, solid line). B. The IKr component of the original ORd model was replaced by the dynamic hERG model, and the IKr conductance was decreased by 1.1-fold. The resulting model (IKr-dynamic ORd model) was used to simulate APs at different cycle lengths, and the steady-state AP duration (APD) rate dependence was compared with the original ORd model. Symbols are original ORd simulations, whereas lines are IKr-dynamic ORd simulations. C. AP waveform comparisons when CL is 1000 ms (upper) and 500 ms (bottom). Solid lines are IKr-dynamic ORd simulations, whereas symbols are original simulations. Because the AP shapes generated by these 2 models are virtually identical, the lines and symbols are superimposed and not visually separable.](http://circep.ahajournals.org/)}
Of the various candidate metrics screened, the drug-induced change in qNaL and qCaL during bradycardia (cqInward) was the only one that separated the 3 CiPA categories in a mechanistic and concentration-dependent manner. It is well known that AP repolarization depends on a subtle balance between outward and inward currents.36 The value of cqInward reflects the drug-induced shift of balance between inward and outward currents. The higher the cqInward value, the more the balance tilts toward inward currents, and the closer the system is to an EAD. Even though our work presents a computational framework that differentiates proarrhythmic potential, this is not necessarily the final model/metric to be used under CiPA, and some important limitations need to be considered in further model development. First, the current IKr-dynamic ORd model only considers frequency-dependent hERG/IKr blocking while ignoring frequency-dependent block of other channels, for instance ICaL and INaL. Also, it has been proposed that some currents in the ORd model may need to be reformulated to more accurately reproduce drug-blocking experiments.37 Work is currently underway to improve the mathematical representation of these channels in the ORd model. Second, the proposed candidate metric may place a compound without any channel-blocking activities into the intermediate-risk group, as its cqInward curve is expected to be flat as well. This, however, can be remedied by automatically assigning non-hERG blockers to the no-risk group. Another limitation of the charge metric is the highest concentrations simulated in this study were ≤25× free Cmax, as inferred blocking levels calculated from the IC50s may no longer be accurate at concentrations too much above clinical Cmax (see legend of Table 3 for details). Patch-clamp assays will need to use wider concentration ranges so that the simulation can be pushed to even higher concentrations.

### Table 3. IC50s Used in AP Simulation

<table>
<thead>
<tr>
<th>Drug</th>
<th>hERG</th>
<th>INaL</th>
<th>ICaL</th>
<th>Ina</th>
<th>Ito</th>
<th>IK1</th>
<th>IKs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Quinidine</td>
<td>992</td>
<td>9417</td>
<td>51592.3</td>
<td>12329</td>
<td>3487.4</td>
<td>39589919</td>
<td>4898.9</td>
</tr>
<tr>
<td>Quinidine</td>
<td>h</td>
<td>0.8</td>
<td>1.3</td>
<td>0.6</td>
<td>1.5</td>
<td>1.3</td>
<td>0.4</td>
</tr>
<tr>
<td>Bepridil</td>
<td>50</td>
<td>1813.9</td>
<td>2808.1</td>
<td>2929.3</td>
<td>8594</td>
<td>0</td>
<td>26628.3</td>
</tr>
<tr>
<td>Bepridil</td>
<td>h</td>
<td>0.9</td>
<td>1.4</td>
<td>0.6</td>
<td>1.2</td>
<td>3.5</td>
<td>0</td>
</tr>
<tr>
<td>Dofetilide</td>
<td>4.9</td>
<td>753160.4</td>
<td>260.3</td>
<td>380.5</td>
<td>18.8</td>
<td>394.3</td>
<td>0</td>
</tr>
<tr>
<td>Dofetilide</td>
<td>h</td>
<td>0.9</td>
<td>0.3</td>
<td>1.2</td>
<td>0.9</td>
<td>0.8</td>
<td>0</td>
</tr>
<tr>
<td>Sotalol</td>
<td>110600</td>
<td>0</td>
<td>7061527</td>
<td>1.14E+9</td>
<td>43143455</td>
<td>3050260</td>
<td>4221856</td>
</tr>
<tr>
<td>Sotalol</td>
<td>h</td>
<td>0.8</td>
<td>0</td>
<td>0.9</td>
<td>0.5</td>
<td>0.7</td>
<td>1.2</td>
</tr>
<tr>
<td>Chlorpromazine</td>
<td>929.2</td>
<td>4559.6</td>
<td>8191.9</td>
<td>4535.6</td>
<td>1761711</td>
<td>9269.9</td>
<td>0</td>
</tr>
<tr>
<td>Chlorpromazine</td>
<td>h</td>
<td>0.8</td>
<td>0.9</td>
<td>0.8</td>
<td>2</td>
<td>0.4</td>
<td>0</td>
</tr>
<tr>
<td>Cisapride</td>
<td>10.1</td>
<td>0</td>
<td>9258076</td>
<td>0</td>
<td>219112.4</td>
<td>29498</td>
<td>81192862</td>
</tr>
<tr>
<td>Cisapride</td>
<td>h</td>
<td>0.7</td>
<td>0</td>
<td>0.4</td>
<td>0</td>
<td>0.2</td>
<td>0.5</td>
</tr>
<tr>
<td>Terfenadine</td>
<td>23</td>
<td>20056</td>
<td>700.4</td>
<td>4803.2</td>
<td>239960.8</td>
<td>0</td>
<td>399754</td>
</tr>
<tr>
<td>Terfenadine</td>
<td>h</td>
<td>0.6</td>
<td>0.6</td>
<td>0.7</td>
<td>1</td>
<td>0.3</td>
<td>0</td>
</tr>
<tr>
<td>Ondansetron</td>
<td>1320</td>
<td>19180.8</td>
<td>22551.4</td>
<td>57666.4</td>
<td>1023378</td>
<td>0</td>
<td>569807</td>
</tr>
<tr>
<td>Ondansetron</td>
<td>h</td>
<td>0.9</td>
<td>1</td>
<td>0.8</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Diltiazem</td>
<td>13150</td>
<td>21868.5</td>
<td>112.1</td>
<td>110859</td>
<td>2.82E+09</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Diltiazem</td>
<td>h</td>
<td>0.9</td>
<td>0.7</td>
<td>0.7</td>
<td>0.7</td>
<td>0.2</td>
<td>0</td>
</tr>
<tr>
<td>Mexiletine</td>
<td>28880</td>
<td>8956.8</td>
<td>38243.6</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Mexiletine</td>
<td>h</td>
<td>0.9</td>
<td>1.4</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Ranolazine</td>
<td>8270</td>
<td>7884.5</td>
<td>0</td>
<td>68774</td>
<td>0</td>
<td>0</td>
<td>36155020</td>
</tr>
<tr>
<td>Ranolazine</td>
<td>h</td>
<td>0.9</td>
<td>0.9</td>
<td>0</td>
<td>1.4</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Verapamil</td>
<td>288</td>
<td>7028</td>
<td>201.8</td>
<td>0</td>
<td>13429.2</td>
<td>3.49E+8</td>
<td>0</td>
</tr>
<tr>
<td>Verapamil</td>
<td>h</td>
<td>1</td>
<td>1</td>
<td>1.1</td>
<td>0</td>
<td>0.8</td>
<td>0.3</td>
</tr>
</tbody>
</table>

IC50s and Hill coefficients (hs) for hERG are estimated using Milnes protocol data in this study. Other channels' IC50 and h values are calculated using the blocking data from the study by Crumb et al.27 Zeros in the table indicate no detectable blocking for that channel. We chose to use our own data instead of Crumb et al data to calculate hERG IC50s because results are to be compared with those of the IKr-dynamic ORd model, where hERG dynamic parameters were estimated from our Milnes protocol data. Note that many drugs' IC50s are high because only a narrow concentration range around Cmax was tested in the study by Crumb et al.27 In those cases, the estimated IC50s may not be accurate in predicting block at high concentrations. Therefore, all simulations in this study were limited to ≤25× Cmax. AP indicates action potential; hERG, human Ether-à-go-go-Related Gene; IC50, half-maximal blocking concentration; ICaL, L-type calcium current; IK, potassium current; INa, peak sodium current; INaL, late sodium current; and Ito, transient outward potassium current.
Finally, the performance of this model/metric will need to be independently evaluated using the set of 16 CiPA validation compounds, the data for which are in the process of being collected. Right now, it is unclear whether a 12-drug training set is adequate to train the model. It is possible that expanding the training set is necessary to sufficiently
capture various mechanisms behind drug-induced TdP. Possible training set expansion strategies could be combining current training and validation compounds to use leave-one-out validation, or categorizing more compounds into separate training and validation sets.

**Acknowledgments**

We would like to thank Dr Gary Mirams, Dr Adam Hill, Dr Jamie Vandenberg, Dr Jules Hancox, Dr Bernard Fermini, Dr Jim Kramer, and Dr Najah Abi-Gerges for the insightful discussions about this article. This report is not an official US Food and Drug
Figure 7. The metric of cqInward as simulated by the original O’Hara–Rudy (ORd) model using half-maximal blocking concentration (IC50s) for 7 ion channels. The simulation protocol was exactly the same as Figure 6C, except that hERG/IKr IC50s (Table 3) were used in place of the dynamic hERG model to simulate action potentials (APs) in the original ORd model. The intermediate-risk compound cisapride is mixed with other high-risk compounds. Stars indicate the threshold dose as in Figure 6C. Note that quinidine induced early afterdepolarizations (EADs) within this concentration range even though its cqInward metric values before EAD were inseparable from those of intermediate-risk compounds.

Administration guidance or policy statement. No official support or endorsement by the US Food and Drug Administration is intended or should be inferred.

Sources of Funding
This project was funded by the Food and Drug Administration Critical Path Award.

Disclosures
None.

References


Improving the In Silico Assessment of Proarrhythmia Risk by Combining hERG (Human Ether-à-go-go-Related Gene) Channel – Drug Binding Kinetics and Multichannel Pharmacology

Zhihua Li, Sara Dutta, Jiansong Sheng, Phu N. Tran, Wendy Wu, Kelly Chang, Thembi Mdluli, David G. Strauss and Thomas Colatsky

*Circ Arrhythm Electrophysiol.* 2017;10:e004628
doi: 10.1161/CIRCEP.116.004628

*Circulation: Arrhythmia and Electrophysiology* is published by the American Heart Association, 7272 Greenville Avenue, Dallas, TX 75231
Copyright © 2017 American Heart Association, Inc. All rights reserved.
Print ISSN: 1941-3149. Online ISSN: 1941-3084

The online version of this article, along with updated information and services, is located on the World Wide Web at:
http://circep.ahajournals.org/content/10/2/e004628

An erratum has been published regarding this article. Please see the attached page for:
/content/10/3/e000024.full.pdf

Data Supplement (unedited) at:
http://circep.ahajournals.org/content/suppl/2017/02/15/CIRCEP.116.004628.DC1

Permissions: Requests for permissions to reproduce figures, tables, or portions of articles originally published in *Circulation: Arrhythmia and Electrophysiology* can be obtained via RightsLink, a service of the Copyright Clearance Center, not the Editorial Office. Once the online version of the published article for which permission is being requested is located, click Request Permissions in the middle column of the Web page under Services. Further information about this process is available in the Permissions and Rights Question and Answer document.

Reprints: Information about reprints can be found online at:
http://www.lww.com/reprints

Subscriptions: Information about subscribing to *Circulation: Arrhythmia and Electrophysiology* is online at:
http://circep.ahajournals.org/subscriptions/
Correction to: Improving the In Silico Assessment of Proarrhythmia Risk by Combining hERG (Human Ether-à-go-go-Related Gene) Channel–Drug Binding Kinetics and Multichannel Pharmacology

In the article by Li et al, “Improving the In Silico Assessment of Proarrhythmia Risk by Combining hERG (Human Ether-à-go-go-Related Gene) Channel–Drug Binding Kinetics and Multichannel Pharmacology”, which published online on February 15, 2017, and appeared in the February 2017 issue of the journal (Circulation: Arrhythmia and Electrophysiology. 2017;10:e004628. DOI: 10.1161/CIRCEP.116.004628, corrections are needed.

1. In the author byline, “David G. Strauss, PhD” should be “David G. Strauss, MD, PhD”.
2. In the Acknowledgments, Dr Najah Abi-Gerges was omitted from the doctors we wish to thank. He has been added in “We would like to thank Dr Gary Mirams, Dr Adam Hill, Dr Jamie Vandenberg, Dr Jules Hancox, Dr Bernard Fermini, Dr Jim Kramer, and Dr Najah Abi-Gerges for the insightful discussions about this article.”

The authors apologize for these errors.

These corrections have been made to the current online version of the article, which is available at http://circep.ahajournals.org/content/10/2/e004628.
Supplemental Material
Supplemental Methods

Cell line

Experiments were performed on a HEK293 cell line that stably expresses hERG1a subunit (provided by Dr. Gail Robertson, University of Wisconsin-Madison) \(^1\). Cells were passaged using trypsin, and seeded onto small sterilized glass coverslips in 35 mm petri dishes containing minimum essential media (MEM; Gibco, Thermo Fisher Scientific) supplemented with 10% fetal bovine serum (Gibco, Thermo Fisher Scientific) and geneticin (G418; 100 \(\mu\)g/mL; Gibco, Thermo Fisher Scientific). Following passage, cells were incubated at 37°C for a minimum of 24 hrs prior to use for electrophysiology studies. All cells were studied within 56 hrs of seeding.

Drugs

D,L-sotalol hydrochloride, cisapride monohydrate, dofetilide, azimilide dihydrochloride, chlorpromazine hydrochloride, and DMSO were purchased from Sigma. All other drugs were purchased from Tocris Bioscience. All drugs were dissolved in DMSO to make stock solution. The final concentration of DMSO applied to cells did not exceed 0.03%. The list of compounds used are listed in Table 1. The selection of the 4 concentrations for each drug in the \textit{in vitro} experiment was based on the aim to have the lowest concentration causing 10%-25% steady state block while the highest concentration causing 75%-90% steady state block. The other two concentrations cause intermediate blocking of hERG at steady state. This way a full dose-response relationship could be established and used to estimate the kinetic parameters. Because the concentration ranges were established by trial and error, occasionally a drug’s lowest or highest degree of block is out of the ideal range that was sought. Among the 12 drugs, only one (mexiletine) blocked hERG above the ideal lower bound at lowest concentration (10 \(\mu\)M blocking 26.9%), while only one other drug (sotalol) blocked hERG less than the ideal upper
bound at the highest concentration (300 μM blocking 70.8%). The drug concentrations used for each compound can be found in Fig S1-S12.

Electrophysiology
Experiments were conducted at 37°C. Glass coverslips with cells were placed in a recording chamber mounted on an inverted (Zeiss Axiovert 135TV) or an upright microscope (Zeiss Axio Examiner D1), and the recording chamber was continuously perfused with an external solution flowing at a rate of 2-3 mL/min and containing the following (in mM): 130 NaCl, 10 HEPES, 5 KCl, 1 MgCl₂·6H₂O, 1 CaCl₂·H₂O, 12.5 dextrose; pH adjusted to 7.4 with 5 M NaOH; ~280 mOsM. Cells were visualized using phase contrast method for the inverted microscope and differential interference contrast-infrared method for the upright microscope for guided patching. Recording electrodes were pulled from filamented borosilicate glass pipettes (BF150-86-10; Sutter Instrument, CA), and had tip resistances < 3 MΩ when filled with the following internal solution (in mM): 120 K-gluconate, 20 KCl, 10 HEPES, 5 EGTA, 1.5 MgATP; pH adjusted to 7.3 with 1 M KOH; ~280 mOsM. Temperatures of the in-line solution heater and recording chamber were maintained with a dual channel temperature controller (TC2BIP from Cell MicroControls for the inverted microscope setup; TC-344C from Warner Instruments for the upright microscope setup), and temperature of the perfusate near the cells was recorded throughout the experiment with an additional thermistor placed inside the recording chamber near the recording site. Recordings were obtained using a Multiclamp 700B amplifier (Molecular Devices, CA). For whole-cell voltage clamp recordings, the command potential values were corrected for the 15 mV liquid junction potential that resulted from using the above internal solution. For protocols involving step voltage waveforms, signals were filtered at 2-3 kHz, digitized using a Digidata 1550A1 interface (Molecular Devices, CA) at 5-10 kHz, and
transferred to a computer using pClamp10 software (Molecular Devices, CA). Seal resistance ranged from 2-10 GΩ, and series resistance was electronically compensated for at 80%.

Membrane properties of all cells included in this study (n = 276) are as follows: resting membrane potential (recorded upon whole cell formation) = -65.1±0.5 mV, holding current at -80 mV = -18.8±0.8 pA; whole cell capacitance = 14.9±0.3 pF; input resistance = 939.4±32.3 MΩ; hERG current peak amplitude = 772.9±24.9 pA. HERG current was evoked with a 10 s depolarizing voltage step from -80 mV to 0 mV for 10 s every 25 s. To evaluate the effects of drugs on hERG current, baseline hERG current stability - defined as <5% difference in hERG current amplitude for 10 consecutively recorded traces - was achieved prior to drug application for every cell. Each cell was exposed only to one concentration of drug.

Fitting of the hERG binding model to experimental data from Milnes protocol.

The raw abf files were pre-processed using in-house developed matlab codes to re-sample the traces during the 0 mV depolarization step at a frequency of 100 Hz with a sliding smoothing window of 4 ms. This resulted in 1000 data points for each 10 s depolarization step. The first point was taken at 5 ms after the beginning of the 0 mV step to avoid artificial voltage spikes. Then for each episode the data after drug wash-in were normalized against the corresponding episode in the control data (before drug wash-in) for the same cell to obtain fractional block. The fractional block data were finally averaged across cells for each concentration respectively. Because during the initial ~100 ms of the depolarization step the channel opening process is mixed with the drug block development process, we have found that the use of data from this period should be minimized as much as possible to obtain a more accurate estimation of the drug binding kinetics. Since different drugs have different binding rate, we used a two-step decision process to determine how much of the initial phase should be used during model fitting:
1) For fast blockers that reached over 50% block at the 11th (105 ms into the 0 mV step), data points from the first point reaching 50% block to the last (1000th) were used, thus excluding part of the initial channel opening phase; 2) for slow blockers whose blocking percentage did not exceed 50% at the 11th data point, data points from 11 to 1000 were used, thus excluding almost the entire initial channel opening phase from model fitting. The number of points to use was determined through the above method based on the first episode for each of the drug concentrations, and was kept the same for all remaining 9 episodes. After the data points to be used are determined the free parameters for each drug were then adjusted by minimizing the cost function describing the difference (error) between observed experimental data and model simulations during the 0 mV clamp steps. The cost function is

\[
\text{Error} = w_1 \cdot \sum_{i=1}^{N} (\text{ExperimentalData}_i - \text{SimulationData}_i)^2 + w_2 \cdot \sum_{j=1}^{n} (\text{ExperimentalP1}_j - \text{SimulationP1}_j)^2 + w_3 \cdot \sum_{j=1}^{n} (\text{ExperimentalP2}_j - \text{SimulationP2}_j)^2
\]

Here the first term compares the model results to the actual trace data (normalized as fractional block and averaged across cells as described above), where \(N\) is the total number of data points used across all doses and all episodes, \(\text{ExperimentalData}_i\) and \(\text{SimulationData}_i\) are the observed and simulated data points at the \(i\)th time point respectively, and \(w_1\) is the weight and fixed as 1 for the first term. The second and third terms compare the observed and simulated “trapping phenotype” as described in the main text. For the second term, \(n\) is the number of doses where at least 50% block were achieved, \(\text{ExperimentalP1}_j\) is the relative peak reduction between the 1st and 2nd episodes for the \(j\)th dose, \(\text{SimulationP1}_j\) is the corresponding simulated value, and \(w_2\) is manually fixed as \(0.2 \cdot N/4\), where \(N\) is the total number of selected points and 4 is the total number of doses. The relative peak reduction is defined as \((\text{Peak2} - \text{Peak1})/\text{Peak1}\) where Peak2 and Peak1 are the peak values (operationally defined as the 1st data point selected using the above two-step decision process) for the 2nd and 1st episodes respectively.
The third term is the same as the second term except that here the relative peak reductions (ExperimentalP2 and SimulationP2) are between the 1st and 10th episodes. Fitting was done using the Genetic Algorithm-based Parameterization for Systems Modeling (GAPSM) developed in-house earlier and implemented on a High Performance Computing cluster hosted by FDA Center for Devices and Radiological Health. Typically 50-100 iterations (generations) of GAPSM were needed to obtain a good set of parameters, and the algorithm was repeated multiple times with different random seeds to evaluate the uniqueness of the obtained parameters (Table S2).

**IKr-dynamic ORd Model**

The original ORd model was transferred from Matlab to R (http://www.r-project.org) and C code using the numerical solving package deSolve (http://desolve.r-forge.r-project.org/) and implemented on the FDA High Performance Computer (HPC). Action potential duration (APD) was measured at 30, 50, 70 and 90% repolarization (APDx), which is defined as the time the membrane voltage repolarizes X% between the peak and resting membrane potential. To assess the degree of reverse use dependent (RUD) APD prolongation, the formula $APD_{90} = R \cdot \sqrt{C} + B$ was used, where R is the slope, and B is the intercept of the linear regression line. The amount of charges passing through each channel was calculated by integrating the area under the curve (AUC) of the corresponding current during the action potential. The highest doses tested for metric searching are 25x Cmax for each individual drug, because the IC50s used were calculated from blocking experiments using drug concentrations no more than 10x Cmax for some drugs and doses in the simulation too much above the highest doses tested experimentally may have inaccurate blocking level prediction. The three CiPA categories are well separated at high concentrations.
**Leave-one-out validation**

A leave-one-out cross validation was performed for the cqInward metric at the highest cmax tested 25x, or the cmax prior to developing an EAD for quinidine, bepridil and dofetilide. Briefly, as described in Mirams study\(^5\), one drug was removed from the data set and a linear discriminant analysis (LDA) classifier was built based on the cqInward values of the remaining drugs, and then used to predict the risk category of the drug that was left out. This was performed in turn for each drug within the data set. The IKr-dynamic ORd model returned 100% accuracy while the original ORd model with IC50 information had two mis-predictions as seen in Table S3. The code was written in matlab using the function classify.

---

**Supplemental Tables**

**Differential Equations for the hERG Model with Drug Binding**

The 9-state hERG model with drug binding is defined by the following equations:

\[
\frac{dIC_1}{dt} = A_{21} \cdot e^{b_{21} \cdot V} \cdot IC_2 \cdot q_{21}^{(T-20)/10} - A_{11} \cdot e^{b_{11} \cdot V} \cdot IC_1 \cdot q_{11}^{(T-20)/10} + A_{51} \cdot e^{b_{51} \cdot V} \cdot IC_1 \cdot q_{51}^{(T-20)/10} - A_{61} \cdot e^{b_{61} \cdot V} \cdot IC_1 \cdot q_{61}^{(T-20)/10}
\]
\[
\frac{dI_C_2}{dt} = \left( A_{11} \cdot e^{B_{11} \cdot V} \cdot I_C_1 \cdot q_{11} \right)^{(T-20)}_{/10} - \left( A_{21} \cdot e^{B_{21} \cdot V} \cdot I_C_2 \cdot q_{21} \right)^{(T-20)}_{/10} - A_3 \cdot e^{B_3 \cdot V} \cdot I_C_2 \cdot q_3^{(T-20)}_{/10} \\
+ A_4 \cdot e^{B_4 \cdot V} \cdot I \cdot O \cdot q_4^{(T-20)}_{/10} + A_{52} \cdot e^{B_{52} \cdot V} \cdot C_2 \cdot q_{52}^{(T-20)}_{/10} - A_{62} \cdot e^{B_{62} \cdot V} \cdot I C_2 \cdot q_6^{(T-20)}_{/10}
\]

\[
\frac{dC_1}{dt} = A_2 \cdot e^{B_2 \cdot V} \cdot C_2 \cdot q_2^{(T-20)}_{/10} - A_1 \cdot e^{B_1 \cdot V} \cdot C_1 \cdot q_1^{(T-20)}_{/10} - A_{51} \cdot e^{B_{51} \cdot V} \cdot C_1 \cdot q_{51}^{(T-20)}_{/10} + A_{61} \\
\cdot e^{B_{61} \cdot V} \cdot I C_1 \cdot q_{61}^{(T-20)}_{/10}
\]

\[
\frac{dC_2}{dt} = A_1 \cdot e^{B_1 \cdot V} \cdot C_1 \cdot q_1^{(T-20)}_{/10} - A_2 \cdot e^{B_2 \cdot V} \cdot C_2 \cdot q_2^{(T-20)}_{/10} - A_{31} \cdot e^{B_{31} \cdot V} \cdot C_2 \cdot q_{31}^{(T-20)}_{/10} + A_{41} \\
\cdot e^{B_{41} \cdot V} \cdot O \cdot q_{41}^{(T-20)}_{/10} - A_{52} \cdot e^{B_{52} \cdot V} \cdot C_2 \cdot q_{52}^{(T-20)}_{/10} + A_{62} \cdot e^{B_{62} \cdot V} \cdot I C_2 \cdot q_{62}^{(T-20)}_{/10}
\]

\[
\frac{dO}{dt} = A_{31} \cdot e^{B_{31} \cdot V} \cdot C_2 \cdot q_{31}^{(T-20)}_{/10} - A_{41} \cdot e^{B_{41} \cdot V} \cdot O \cdot q_{41}^{(T-20)}_{/10} - A_{53} \cdot e^{B_{53} \cdot V} \cdot O \cdot q_{53}^{(T-20)}_{/10} \\
+ A_{63} \cdot e^{B_{63} \cdot V} \cdot I O \cdot q_{63}^{(T-20)}_{/10} + K_u \cdot O^* - K_u \cdot \frac{K_{max} \cdot D}{D + E C_{50}} \cdot O
\]

\[
\frac{dI_O}{dt} = A_3 \cdot e^{B_3 \cdot V} \cdot I C_2 \cdot q_3^{(T-20)}_{/10} - A_4 \cdot e^{B_4 \cdot V} \cdot I O \cdot q_4^{(T-20)}_{/10} + A_{53} \cdot e^{B_{53} \cdot V} \cdot O \cdot q_{53}^{(T-20)}_{/10} - A_{63} \\
\cdot e^{B_{63} \cdot V} \cdot I O \cdot q_{63}^{(T-20)}_{/10} + K_u \cdot \frac{A_{53} \cdot e^{B_{53} \cdot V} \cdot q_{53}^{(T-20)}_{/10}}{A_{63} \cdot e^{B_{63} \cdot V} \cdot q_{63}^{(T-20)}_{/10}} \cdot I O^* - K_u \cdot \frac{K_{max} \cdot D}{D + E C_{50}} \cdot I O
\]

\[
\frac{dI_O_{bound}}{dt} = K_u \cdot \frac{K_{max} \cdot D}{D + E C_{50}} \cdot I O - K_u \cdot \frac{A_{53} \cdot e^{B_{53} \cdot V} \cdot q_{53}^{(T-20)}_{/10}}{A_{63} \cdot e^{B_{63} \cdot V} \cdot q_{63}^{(T-20)}_{/10}} \cdot I O_{bound} \\
+ K_t \left(1 + e^{-\frac{(V-V_{half-trap})}{6.789}} \right)^{-1} \cdot C_{bound} - K_t \cdot I O_{bound}
\]
\[
\frac{dO_{\text{bound}}}{dt} = K_u \cdot \frac{K_{\text{max}} \cdot D^n}{D^n + E_{C50}} \cdot O - K_u \cdot O_{\text{bound}} + K_t \left( 1 + e^{-\frac{(V - V_{\text{half-trap}})}{6.789}} \right)^{-1} \cdot C_{\text{bound}} - K_t \cdot O_{\text{bound}}
\]

\[
\frac{dC_{\text{bound}}}{dt} = K_t \cdot O_{\text{bound}} - K_t \left( 1 + e^{-\frac{(V - V_{\text{half-trap}})}{6.789}} \right)^{-1} \cdot C_{\text{bound}} + K_t \cdot IO_{\text{bound}}
\]

As defined previously \(^6\), IC1, IC2, C1, C2, O and IO are the 6 drug-unbound states of the model: two inactive closed states 1 and 2, two closed states 1 and 2, one open state and one open inactive state. \(V\) is the membrane potential (mV), \(A_x\) and \(B_x\) reflect the energy barrier height in the absence and presence of electrical field respectively for each transition \(x\), \(q_x\) is the commonly used temperature extrapolating Q10 value defined as the change in rate for each 10°C change in temperature for each state transition \(x\), and \(T\) is the temperature.

As defined in the main text, IO_{\text{bound}}, O_{\text{bound}}, and C_{\text{bound}} are the open inactive, open, and closed drug-bound states, respectively. \(K_t\) is the trapping rate, which was manually fixed at \(3.5 \cdot 10^{-5} \text{ ms}^{-1}\). \(K_u\) is the drug unbinding rate, \(K_{\text{max}}\) is the maximum drug effect at saturating concentrations, \(D\) is the drug concentration, \(n\) is the Hill coefficient, \(EC_{50}\) is the half maximal effective concentration of the drug, and \(V_{\text{half-trap}}\) is the membrane voltage at which half of the drug-bound channels are open (see Table 2).
Table S1

<table>
<thead>
<tr>
<th>Parameters for the physiological component of IKr model</th>
</tr>
</thead>
<tbody>
<tr>
<td>$A_1 = 0.0264$</td>
</tr>
<tr>
<td>$A_2 = 4.986e-06$</td>
</tr>
<tr>
<td>$A_3 = 0.001214$</td>
</tr>
<tr>
<td>$A_4 = 1.854e-05$</td>
</tr>
<tr>
<td>$A_{11} = 0.0007868$</td>
</tr>
<tr>
<td>$A_{21} = 5.455e-06$</td>
</tr>
<tr>
<td>$A_{31} = 0.005509$</td>
</tr>
<tr>
<td>$A_{41} = 0.001416$</td>
</tr>
<tr>
<td>$A_{51} = 0.4492$</td>
</tr>
<tr>
<td>$A_{52} = 0.3181$</td>
</tr>
<tr>
<td>$A_{53} = 0.149$</td>
</tr>
<tr>
<td>$A_{61} = 0.01241$</td>
</tr>
<tr>
<td>$A_{62} = 0.3226$</td>
</tr>
<tr>
<td>$A_{63} = 0.008978$</td>
</tr>
</tbody>
</table>

Table S2

Dynamic hERG binding parameters for all CiPA training compounds obtained by different fitting runs with different starting points (random seeds). Fitting results for $K_u$, $n$, and $V_{\text{half-trap}}$ were generally consistent between different seeds. $V_{\text{half-trap}}$ determines $X(-80)$, the steady-state ratio of $O^*$ and $C^*$ states at the resting membrane potential (-80 mV), which also remained consistent between seeds. Fitting results for $K_{\text{max}}$ and $EC_{50}^n$ were consistent for cisapride and diltiazem. For other drugs, the estimated $EC_{50}^n$ was much greater than the concentrations tested experimentally ($EC_{50}^n > 10$ fold of estimated IC50), and the dose-response relationship was approximately linear with slope $K_{\text{max}}/EC_{50}^n$ (see Methods). In these cases, $K_{\text{max}}/EC_{50}^n$
was generally consistent between different seeds. Fitting results for seed = 100 were used in
the hERG-dynamic ORd model (see Table 2).

<table>
<thead>
<tr>
<th>drug</th>
<th>seed</th>
<th>( K_u ) (ms(^{-1}))</th>
<th>( K_{max} )</th>
<th>( E_{C_{50}}^{n} ) (mM)</th>
<th>( K_{max}^{n} )</th>
<th>( n )</th>
<th>( V_{half/trap} ) (mV)</th>
<th>( X(-80) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>quinidine</td>
<td>100</td>
<td>1.00E-02</td>
<td>5.77E+03</td>
<td>1.00E+06</td>
<td>5.77E-03</td>
<td>0.83</td>
<td>-64.87</td>
<td>0.097</td>
</tr>
<tr>
<td></td>
<td>200</td>
<td>8.42E-03</td>
<td>5.12E+02</td>
<td>9.64E+04</td>
<td>5.30E-03</td>
<td>0.84</td>
<td>-65.14</td>
<td>0.101</td>
</tr>
<tr>
<td></td>
<td>300</td>
<td>1.14E-02</td>
<td>9.85E+02</td>
<td>1.88E+05</td>
<td>5.23E-03</td>
<td>0.85</td>
<td>-64.76</td>
<td>0.096</td>
</tr>
<tr>
<td>bepridil</td>
<td>100</td>
<td>1.77E-04</td>
<td>3.74E+07</td>
<td>1.00E+09</td>
<td>3.74E-02</td>
<td>0.94</td>
<td>-54.93</td>
<td>0.024</td>
</tr>
<tr>
<td></td>
<td>200</td>
<td>1.72E-04</td>
<td>3.74E+07</td>
<td>9.80E+08</td>
<td>3.81E-02</td>
<td>0.94</td>
<td>-61.62</td>
<td>0.063</td>
</tr>
<tr>
<td></td>
<td>300</td>
<td>1.72E-04</td>
<td>2.04E+06</td>
<td>5.39E+07</td>
<td>3.80E-02</td>
<td>0.94</td>
<td>-61.37</td>
<td>0.060</td>
</tr>
<tr>
<td>dofetilide</td>
<td>100</td>
<td>1.79E-05</td>
<td>1.00E+08</td>
<td>5.48E+08</td>
<td>1.82E-01</td>
<td>1.00</td>
<td>-1.147</td>
<td>0.000</td>
</tr>
<tr>
<td></td>
<td>200</td>
<td>1.63E-05</td>
<td>2.70E+04</td>
<td>1.40E+05</td>
<td>1.92E-01</td>
<td>1.00</td>
<td>-2.35</td>
<td>0.000</td>
</tr>
<tr>
<td></td>
<td>300</td>
<td>1.69E-05</td>
<td>8.24E+05</td>
<td>4.37E+06</td>
<td>1.89E-01</td>
<td>1.00</td>
<td>-2.049</td>
<td>0.000</td>
</tr>
<tr>
<td>sotalol</td>
<td>100</td>
<td>1.99E-02</td>
<td>2.40E+03</td>
<td>9.62E+06</td>
<td>2.50E-04</td>
<td>0.75</td>
<td>-55</td>
<td>0.025</td>
</tr>
<tr>
<td></td>
<td>200</td>
<td>2.22E-02</td>
<td>7.28E+04</td>
<td>2.91E+08</td>
<td>2.50E-04</td>
<td>0.75</td>
<td>-52.05</td>
<td>0.016</td>
</tr>
<tr>
<td></td>
<td>300</td>
<td>2.23E-02</td>
<td>1.98E+05</td>
<td>8.06E+08</td>
<td>2.46E-04</td>
<td>0.75</td>
<td>-52.08</td>
<td>0.016</td>
</tr>
<tr>
<td>chlorpromazine</td>
<td>100</td>
<td>3.87E-02</td>
<td>2.06E+05</td>
<td>5.68E+07</td>
<td>3.63E-03</td>
<td>0.89</td>
<td>-14.57</td>
<td>0.000</td>
</tr>
<tr>
<td></td>
<td>200</td>
<td>4.52E-02</td>
<td>3.63E+06</td>
<td>1.00E+09</td>
<td>3.63E-03</td>
<td>0.89</td>
<td>-14.47</td>
<td>0.000</td>
</tr>
<tr>
<td></td>
<td>300</td>
<td>4.62E-02</td>
<td>3.63E+06</td>
<td>1.00E+09</td>
<td>3.63E-03</td>
<td>0.89</td>
<td>-14.35</td>
<td>0.000</td>
</tr>
<tr>
<td>cisapride</td>
<td>100</td>
<td>4.16E-04</td>
<td>1.00E+01</td>
<td>4.21E+01</td>
<td>2.38E-01</td>
<td>0.97</td>
<td>-199.5</td>
<td>1.000</td>
</tr>
<tr>
<td></td>
<td>200</td>
<td>4.16E-04</td>
<td>1.02E+01</td>
<td>4.23E+01</td>
<td>2.42E-01</td>
<td>0.96</td>
<td>-200</td>
<td>1.000</td>
</tr>
<tr>
<td></td>
<td>300</td>
<td>4.17E-04</td>
<td>1.00E+01</td>
<td>4.21E+01</td>
<td>2.38E-01</td>
<td>0.97</td>
<td>-200</td>
<td>1.000</td>
</tr>
<tr>
<td>terfenadine</td>
<td>100</td>
<td>8.18E-05</td>
<td>9.88E+03</td>
<td>4.14E+04</td>
<td>2.39E-01</td>
<td>0.65</td>
<td>-77.49</td>
<td>0.409</td>
</tr>
<tr>
<td></td>
<td>200</td>
<td>7.78E-05</td>
<td>8.82E+04</td>
<td>3.54E+05</td>
<td>2.49E-01</td>
<td>0.65</td>
<td>-81.67</td>
<td>0.561</td>
</tr>
<tr>
<td></td>
<td>300</td>
<td>7.68E-05</td>
<td>9.86E+03</td>
<td>3.93E+04</td>
<td>2.51E-01</td>
<td>0.65</td>
<td>-82.11</td>
<td>0.577</td>
</tr>
<tr>
<td>ondansetron</td>
<td>100</td>
<td>2.33E-02</td>
<td>3.35E+04</td>
<td>9.95E+06</td>
<td>3.37E-03</td>
<td>0.89</td>
<td>-82.11</td>
<td>0.577</td>
</tr>
<tr>
<td></td>
<td>200</td>
<td>2.34E-02</td>
<td>3.25E+06</td>
<td>1.00E+09</td>
<td>3.25E-03</td>
<td>0.89</td>
<td>-82.17</td>
<td>0.579</td>
</tr>
<tr>
<td></td>
<td>300</td>
<td>2.35E-02</td>
<td>3.30E+05</td>
<td>1.00E+08</td>
<td>3.30E-03</td>
<td>0.89</td>
<td>-82.05</td>
<td>0.575</td>
</tr>
<tr>
<td>diltiazem</td>
<td>100</td>
<td>2.82E-01</td>
<td>2.51E+02</td>
<td>1.00E+06</td>
<td>2.51E-04</td>
<td>0.95</td>
<td>-90.89</td>
<td>0.833</td>
</tr>
<tr>
<td></td>
<td>200</td>
<td>3.26E-01</td>
<td>2.64E+02</td>
<td>1.00E+06</td>
<td>2.64E-04</td>
<td>0.94</td>
<td>-89.81</td>
<td>0.809</td>
</tr>
<tr>
<td></td>
<td>300</td>
<td>2.82E-01</td>
<td>2.63E+02</td>
<td>1.00E+06</td>
<td>2.63E-04</td>
<td>0.94</td>
<td>-91.01</td>
<td>0.835</td>
</tr>
<tr>
<td>mexiletine</td>
<td>100</td>
<td>9.97E-02</td>
<td>1.00E+01</td>
<td>2.31E+06</td>
<td>4.33E-06</td>
<td>1.30</td>
<td>-86.26</td>
<td>0.715</td>
</tr>
<tr>
<td></td>
<td>200</td>
<td>6.92E-02</td>
<td>1.42E+01</td>
<td>1.02E+06</td>
<td>1.40E-05</td>
<td>1.18</td>
<td>-87.67</td>
<td>0.756</td>
</tr>
<tr>
<td></td>
<td>300</td>
<td>6.76E-02</td>
<td>1.43E+01</td>
<td>9.71E+05</td>
<td>1.47E-05</td>
<td>1.17</td>
<td>-88.33</td>
<td>0.773</td>
</tr>
<tr>
<td>ranolazine</td>
<td>100</td>
<td>1.93E-02</td>
<td>5.58E+01</td>
<td>1.47E+05</td>
<td>3.79E-04</td>
<td>0.95</td>
<td>-94.87</td>
<td>0.899</td>
</tr>
<tr>
<td></td>
<td>200</td>
<td>2.21E-02</td>
<td>4.11E+01</td>
<td>1.00E+05</td>
<td>4.11E-04</td>
<td>0.94</td>
<td>-94.68</td>
<td>0.897</td>
</tr>
<tr>
<td></td>
<td>300</td>
<td>1.63E-02</td>
<td>2.75E+02</td>
<td>4.67E+05</td>
<td>5.89E-04</td>
<td>0.90</td>
<td>-96.19</td>
<td>0.916</td>
</tr>
<tr>
<td>verapamil</td>
<td>100</td>
<td>7.93E-04</td>
<td>4.65E+04</td>
<td>9.18E+06</td>
<td>5.06E-03</td>
<td>1.04</td>
<td>-100</td>
<td>0.950</td>
</tr>
<tr>
<td></td>
<td>200</td>
<td>8.19E-04</td>
<td>3.07E+04</td>
<td>6.10E+06</td>
<td>5.03E-03</td>
<td>1.04</td>
<td>-96.35</td>
<td>0.917</td>
</tr>
<tr>
<td></td>
<td>300</td>
<td>8.20E-04</td>
<td>7.49E+03</td>
<td>1.49E+06</td>
<td>5.05E-03</td>
<td>1.04</td>
<td>-100.1</td>
<td>0.951</td>
</tr>
</tbody>
</table>
**Table S3**

Prediction of each drug’s risk category, actual risk and prediction error after leave-one-out validation using the cqlnward metric computed by either the ORd model with IC50-based conductance scaling or the IKr-dynamic ORd model with both IKr binding dynamics and other channels’ IC50s. The IKr-dynamic ORd model was able to correctly predict all drugs’ TdP categories while the original ORd model had two mis-predictions (quinidine and cisapride).

<table>
<thead>
<tr>
<th>Drugs</th>
<th>Actual Risk</th>
<th>Predicted by original ORd</th>
<th>Predicted by IKr-dynamic ORd</th>
</tr>
</thead>
<tbody>
<tr>
<td>Quinidine</td>
<td>High</td>
<td>Intermediate</td>
<td>High</td>
</tr>
<tr>
<td>Bepridil</td>
<td>High</td>
<td>High</td>
<td>High</td>
</tr>
<tr>
<td>Dofetilide</td>
<td>High</td>
<td>High</td>
<td>High</td>
</tr>
<tr>
<td>Sotalol</td>
<td>High</td>
<td>High</td>
<td>High</td>
</tr>
<tr>
<td>Chlorpromazine</td>
<td>Intermediate</td>
<td>Intermediate</td>
<td>Intermediate</td>
</tr>
<tr>
<td>Cisapride</td>
<td>Intermediate</td>
<td>High</td>
<td>Intermediate</td>
</tr>
<tr>
<td>Terfenadine</td>
<td>Intermediate</td>
<td>Intermediate</td>
<td>Intermediate</td>
</tr>
<tr>
<td>Ondansetron</td>
<td>Intermediate</td>
<td>Intermediate</td>
<td>Intermediate</td>
</tr>
<tr>
<td>Diltiazem</td>
<td>Low/No</td>
<td>Low/No</td>
<td>Low/No</td>
</tr>
<tr>
<td>Mexiletine</td>
<td>Low/No</td>
<td>Low/No</td>
<td>Low/No</td>
</tr>
<tr>
<td>Ranolazine</td>
<td>Low/No</td>
<td>Low/No</td>
<td>Low/No</td>
</tr>
<tr>
<td>Verapamil</td>
<td>Low/No</td>
<td>Low/No</td>
<td>Low/No</td>
</tr>
</tbody>
</table>
Fig S1: Model fitting to quinidine. Symbols are experimental values with standard deviation while lines are model simulation.
Fig S2: Model fitting to bepridil. Symbols are experimental values with standard deviation while lines are model simulation.
Fig S3: Model fitting to dofetilide. Symbols are experimental values with standard deviation while lines are model simulation.
Fig S4: Model fitting to sotalol.
Symbols are experimental values with standard deviation while lines are model simulation.
Fig S5: Model fitting to chlorpromazine. Symbols are experimental values with standard deviation while lines are model simulation.
Fig S6: Model fitting to cisapride. Symbols are experimental values with standard deviation while lines are model simulation.
Fig S7: Model fitting to terfenadine. Symbols are experimental values with standard deviation while lines are model simulation.
Fig S8: Model fitting to ondansetron. Symbols are experimental values with standard deviation while lines are model simulation.
Fig S9: Model fitting to diltiazem. Symbols are experimental values with standard deviation while lines are model simulation.
Fig S10: Model fitting to mexiletine. Symbols are experimental values with standard deviation while lines are model simulation.
Fig S11: Model fitting to ranolazine. Symbols are experimental values with standard deviation while lines are model simulation.
Fig S12: Model fitting to verapamil. Symbols are experimental values with standard deviation while lines are model simulation.
Fig S13: The impact of the choice of Kt on drug separation and fitting quality. A series of Kt values were tried as fixed parameters in the model while the other 5 hERG binding dynamic parameters were estimated for each drug. A-E) The distribution of estimated Vhalf-trap parameters for the 12 CiPA training drugs when Kt was fixed at 1e-6, 2e-5, 3.5e-5, 5e-5, and 1e-
4 respectively. High TdP Risk drugs were labeled as red, Intermediate as blue, and Low/No Risk as green. Note that when \( Kt < 2e-5 \) (A), the three categories were mixed completely. When \( Kt \geq 2e-5 \) (B – E), the Low/No Risk drugs (green) and High Risk drugs (red) tend to be separated with the Intermediate Risk drugs (blue) scattered around. The higher the \( Kt \), the closer the Low/No Risk drugs are to the High Risk drugs (less separation). F) The total fitting error for the 12 drugs when \( Kt \) was fixed at each value. From the lowest \( Kt \) (1e-6) to the highest (1e-4), the error has a U-shaped curve with the lowest total error achieved when \( Kt \) was fixed at 3.5e-5. This is the \( Kt \) parameter chosen in the current model.
Supplemental References


Appendix: Source code used for fitting the hERG model.
README

Source code for fitting the pharmacodynamic component of the hERG model is provided here. The GAPSM fitting algorithm was implemented in R and run in parallel with RMPISNOW on a Son of Grid Engine cluster at the FDA.

The R script `processcompileDDEs.R` reads `newODEs.txt` to generate C code used to run the hERG Markov model and R scripts used to initialize the model states and parameters. Additional input files are `nnn.txt`, which specifies the upper and lower bounds of the hERG drug binding parameters, and `moreFracBlockMilnes37.txt`, which contains the experimental results used to fit the model.

The main R script is `newdigitGA.R`, which is called by the cluster job script `runsnowmpisge.sh`. The remaining R files contain additional functions or scripts that are called during the fitting procedure.
```r
bincreateNewGeneration <- function(oldgen, errormat, mutateprob, crossprob, elitism,
     tournamentfrac, istournament = T) {
  # oldgen is a matrix where each column is a person's parameter vector
  # errormat is a matrix where first column is the person no. and second is error
  chromlen <- dim(oldgen)[1]
popsize <- dim(oldgen)[2]
parentsvec <- sample.int(popsize, size = popsize * ceiling(popsize * tournamentfrac), replace = T)
selectmx <- matrix(parentsvec, ncol = popsize, byrow = F)
parents <- matrix(F, nrow = chromlen, ncol = popsize)
children <- matrix(F, nrow = chromlen, ncol = popsize)

  # fitnessvec <- 1/errormat[,2] # now Inf will be converted to 0
  fitnessvec <- 1/rank(errormat[,2])^0.5 # matlab's rank scaling
  wheel <- cumsum(fitnessvec)

  # parents selection
  for (i in 1:popsize) {
    if (errormat[i,2] == min(errormat[,2]) & elitism == 1) {
      parents[,i] <- oldgen[,i]
    } else {
      if (istournament) {
        # tournament selection
        memselect <- selectmx[,i]
        bestfitscore <- min(errormat[memselect,2])
        memcount <- memselect[errormat[memselect,2] == bestfitscore][1]
        parents[,i] <- oldgen[,memcount]
      } else {
        # wheel selection
        pointer <- runif(1, 0, sum(fitnessvec))
        memcount <- which(wheel >= pointer)[1]
        parents[,i] <- oldgen[,memcount]
      }
    } # if tournament
  } # if elitism
  # end parents selection

  # crossover
  for (i in 1:popsize) {
    if (errormat[i,2] == min(errormat[,2]) & elitism == 1) {
      children[,i] <- oldgen[,i]
    } else {
      pos <- 1:popsize
      parent_number2 <- sample(pos[-i], 1)
      if (runif(1) < crossprob) {
        site <- sample.int(chromlen - 1, 1)
        children[1:site, i] <- parents[1:site, i]
      } else {
        children[,i] <- parents[,i]
      }
    } # if crossprob
  } # if elitism
  # end crossover

  # mutation
  for (i in 1:popsize) {
    if (errormat[i,2] == min(errormat[,2]) & elitism == 1) {
      # do nothing
    } else {
      # mutate
    }
  }
}
```

else{
    mutatedpos <- sample.int(chromlen, size=ceiling(chromlen*mutateprob))
    children[mutatedpos,i] <- !children[mutatedpos,i]
}
if elitism
  } #mutation
  return(children)
} #end function
ensemblevoWrapper <- function(idx) {
    mytrait <- currenttrait[, subsplitL[[idx]]]
    myfit <- currentfit[subsplitL[[idx]]]
    POP_SIZE <- dim(mytrait)[2]
    NUM_TRAITS <- dim(mytrait)[1]
    errormat <- cbind(subsplitL[[idx]], myfit)
    elitists <- which(errormat[, 2] == min(errormat[, 2]))
    pickedelitist <- as.integer(sample(as.character(elitists), 1))
    # note as.character in case there's only one elitists!!!!

    # encoding
    pop <- matrix(F, nrow = binCHROM_LENGTH, ncol = POP_SIZE)
    for (i in 1:POP_SIZE) {
        tt <- (mytrait[, i] - LOWTRAIT) * (2^mi - 1)/(HIGHTRAIT - LOWTRAIT)
        ttt <- lapply(tt, function(x) rev(intToBits(x)) == "01")
        for (j in 1:NUM_TRAITS) {
            # extrapos <- 1:(32 - mi[j] + 1)
            shortendvec <- rev(rev(ttt[[j]])[1:mi[j]])
            ttt[[j]] <- shortendvec
        }
        pop[, i] <- unlist(ttt)
    }

    # evolution
    newgen <- bincreateNewGeneration(pop, errormat, MUTAT_PROB, CROSS_PROB, ELITISM, TOURNAMENTFRAC, T); # always use tournament for bin

    # decoding
    newmytrait <- matrix(F, nrow = NUM_TRAITS, ncol = POP_SIZE)
    for (i in 1:POP_SIZE) {
        ttt <- split(newgen[, i], rep(1:NUM_TRAITS, times = mi))
        tt <- lapply(ttt, function(x) logicvec2int(x))
        newmytrait[, i] <- LOWTRAIT + unlist(tt) * (HIGHTRAIT - LOWTRAIT) / (2^mi - 1)
        # real elitism
        if (i == pickedelitist && ELITISM == 1) {
            newmytrait[, i] <- mytrait[, i]
        }
    }
    return(newmytrait)
}
deWrapper <- function(idx, weighting=T){
  pidx <- match(names(pars), pnames, nomatch =0)
  sidx <- match(names(states), snames, nomatch =0)

  indidx <- (chrpernode*idx-(chrpernode-1)):(chrpernode*idx)
  inds <- as.matrix(pop[,indidx]) # matrix needed if indidx is 1 element
  timeout <- 10 # plenty of time for c code
  inderror <- c(0,0);
  numerrors <- 3
  indmat <- rep(0, numerrors)

  # for each individual in the subpopulation
  for(p in 1:length(indidx)){
    ind <- inds[,p]
    catalready <- F
    fval <- 0
    errorvec <- rep(0, numerrors)

    # overwrite parameters
    pars[pidx!=0]<-ind[pidx]
    pars["timeout"] <- timeout
    pars[negativeparnames] <- pars[negativeparnames]*(1)
    pars["Kf2"] <- pars["Kf"]
    pars["Kt2"] <- pars["Kt"]

    # overwrite initial values
    states[sidx!=0]<-ind[sidx]

    # add new error calculations
    if(!catalready){
      source("newgetMilneserror.R")
      error5 <- getMilneserror(pars,states,3,F, controlstates,controlvec,wholetrace,
                                yIdx,tp,beats,TRUE)# last argument is usespecerr
      newerrorlist <- list(error5)
      for (n in 1:length(newerrorlist)){
        derror <- newerrorlist[[n]]
        if((length(derror)==1 & & derror[[1]]== 0)||is.na(derror[[1]])){
          inderror <- rbind(inderror,c(indidx[p], 1e50));
          indmat <- rbind(indmat, rep(NA, numerrors))
          catalready <- T
          break
        }else{
          derror[[1]] <- derror[[1]]
          fval <- fval+derror[[1]]
          errorvec <- errorvec+derror[[2]]
        }
      }# for newerrorlist
      if(catalready){
      
    }# if catalready
  }# for p in 1:length(indidx)
inderror<-rbind(inderror,c(indidx[p],fval));

# indmat<- rbind(indmat, errorvec)
}

# for each person

#indmat<-indmat[-1,]  # only need to return indmat. the sum of errors is the sum of each row.
inderror<-inderror[-1,]  # return sum of errors by now
}
digitmigrationreset.R

```
# migration

program <- "rmpi"

subfitness_bar <- split(fitness_bar[, recycledpopcount], rep(1:4, each = dim(fitness_bar)[1]/4))

if(1>10){
  # use diversity
  alpha <- sapply(subfitness_bar, function(x) (max(x[x!=1e50]) - min(x))/min(x))
  upperq <- sapply(subfitness_bar, function(x) quantile(x, 0.5))
  lowerq <- sapply(subfitness_bar, function(x) quantile(x, 0))
  alpha <- (upperq - lowerq)/lowerq  # avoid using the best because they may be migrants?
  converged <- alpha <= 1
  newdelete <- which(vecmin[,2]==max(vecmin[converged,2])&converged)
  newdelete <- which(lowerq==max(lowerq[converged])&converged)
} else{
  # use elitism
  if(currentdelete == 0){
    newdelete <- which(vecmin[,2]==max(vecmin[,2]))
  } else{
    newdelete <- which(vecmin[,2]==max(vecmin[-currentdelete,2]))
  }
  # !# currentdelete
}

subdeadpos <- lapply(subfitness_bar, function(x) which(x==max(x) | x==1e50))
sublivepos <- lapply(subfitness_bar, function(x) which(x<=max(x)))

everything for( jj in 1:4) {
  # fish chasing tail
  jjtail <- jj%%4+1
  subrange <- (jj*dim(fitness_bar)[1]/4 - dim(fitness_bar)[1]/4+1): (jj*dim(fitness_bar)[1]/4)
  subrangetail <- (jjtail*dim(fitness_bar)[1]/4 - dim(fitness_bar)[1]/4+1): (jjtail*dim(fitness_bar)[1]/4)
  targetpos <- subdeadpos[[jj]]
  livepospool <- sublivepos[[jjtail]]
  pickedlive <- sample(livepospool, length(targetpos), replace=T, prob=1/rank(subfitness_bar[[jjtail]][livepospool])^0.5)
  pickedlive <- sample(livepospool, length(targetpos), replace=T, prob=min(subfitness_bar[[jjtail]][livepospool])/subfitness_bar[[jjtail]][livepospool])
  # let this selection based on fitness score not rank; O(1) algorithm
  trait[,, recycledpopcount][,, subrange][, targetpos] <-
  trait[,, recycledpopcount][,, subrangetail][, pickedlive]
  fitness_bar[, recycledpopcount][subrange][targetpos] <-
  fitness_bar[, recycledpopcount][subrangetail][pickedlive]
}

# go real
for(i in 1:POP_SIZE){
  for(j in seq(1, NUM_TRAITS, 2)){
    realtrait[(j+1)/2, i, recycledpopcount] <- as.numeric(
      paste(substr(trait[j,i, recycledpopcount], 1, sigdigits+1), "e",
      round(as.numeric(trait[j+1,i, recycledpopcount]), sep="")
    )
  }
}
```

# reset
if( length(newdelete)>0){
    newdelete<-newdelete[1]
    jj<-newdelete
    subrange<-(jj*dim(fitness_bar)[1]/4 -
    dim(fitness_bar)[1]/4+1): (jj*dim(fitness_bar)[1]/4)
    for(i in subrange){
        trait[,i,recycledpopcount]<-(runif(NUM_TRAITS)-0.5)*(HIGHTRAIT-LOWTRAIT)+
        0.5*(HIGHTRAIT+LOWTRAIT)
    }
    # reset real
    for(i in subrange){
        for( j in seq(1,NUM_TRAITS,2)){
            realtrait[(j+1)/2,i,recycledpopcount]<- as.numeric(
                paste(substr(trait[j,i,recycledpopcount],1,sigdigits+1),"e",
                round(as.numeric(trait[j+1,i,recycledpopcount])),sep="")
            )
        }
    }
    # fix out-of-bounds traits
    realtrait[,,recycledpopcount]<-apply(realtrait[,,recycledpopcount],2,function(x){
        temp<-pmin(x,realHIGHTRAIT);
        temp<-pmax(temp,realLOWTRAIT);
        return(temp)})
    # go back
    for(i in subrange){
        trait[,i,recycledpopcount]<- as.numeric(sapply(format(realtrait[,i,
        recycledpopcount], scientific=T),
        function(x) sapply(strsplit(x,"e"),
        function(y) y)))
    }
    # reevaluate
    pop<-realtrait[,subrange,recycledpopcount] # update pop with the new random generation
    pop<-signif(pop,digits=sigdigits)
    subevaluationWorkerNum<- length(subrange)/chrpernode # has to be dividable!
    if(program == "rmpi"){
        clusterExport(cl,"pop")
        errorlist<-clusterApplyLB(cl, 1:subevaluationWorkerNum, deWrapper, weighting)
    }else{
        sfExport("pop")
        errorlist<-sfClusterApplyLB(1:subevaluationWorkerNum, deWrapper, weighting)
    }
    errormat<-do.call(rbind, errorlist)
    errormat<-errormat[order(errormat [,1]),]
    fitness_bar[subrange,recycledpopcount]<-errormat [,2]
    if(program == "rmpi"){
        clusterEvalQ(cl, rm(pop))
    }else{
        sfRemove(pop)
    }
}
rm(pop) # otherwise it will affect saving
# Example fractional block data file for Milnes protocol
# Note: file has been truncated

<table>
<thead>
<tr>
<th>time</th>
<th>mean_10nM</th>
<th>std_10nM</th>
<th>mean_30nM</th>
<th>std_30nM</th>
<th>mean_100nM</th>
<th>std_100nM</th>
<th>mean_300nM</th>
<th>std_300nM</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>0.943711076</td>
<td>0.108625839</td>
<td>0.951046623</td>
<td>0.108875365</td>
<td>1.057984273</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>0.921891142</td>
<td>0.074384296</td>
<td>0.901587119</td>
<td>0.11570237</td>
<td>1.02250597</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>0.954367697</td>
<td>0.068449001</td>
<td>0.904319177</td>
<td>0.123942733</td>
<td>1.035763514</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>0.944304601</td>
<td>0.096862528</td>
<td>0.905985232</td>
<td>0.136826886</td>
<td>0.521689371</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>0.898552243</td>
<td>0.058499874</td>
<td>0.782064872</td>
<td>0.109035641</td>
<td>0.867871313</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>0.903439319</td>
<td>0.076393541</td>
<td>0.651015795</td>
<td>0.107682869</td>
<td>0.118262867</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>0.89458151</td>
<td>0.072616128</td>
<td>0.654928929</td>
<td>0.141670866</td>
<td>0.11886265</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>0.91507189</td>
<td>0.18661137</td>
<td>0.645367651</td>
<td>0.10802686</td>
<td>0.107682869</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>0.94304601</td>
<td>0.096862528</td>
<td>0.795985232</td>
<td>0.10880268</td>
<td>0.107682869</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>0.930914594</td>
<td>0.058037932</td>
<td>0.867871313</td>
<td>0.10903564</td>
<td>0.107682869</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>0.880597409</td>
<td>0.095634417</td>
<td>0.627076568</td>
<td>0.118262867</td>
<td>0.481946868</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>0.80734464</td>
<td>0.087222079</td>
<td>0.624215474</td>
<td>0.11886265</td>
<td>0.480267965</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>0.15356816</td>
<td>0.166955561</td>
<td>0.32487889</td>
<td>0.108126916</td>
<td>1.006496375</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>0.1823096</td>
<td>0.803630739</td>
<td>0.057368301</td>
<td>0.102288272</td>
<td>0.868844271</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>0.885727274</td>
<td>0.078236348</td>
<td>0.75679544</td>
<td>0.102288272</td>
<td>0.868844271</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>0.13644983</td>
<td>0.657794242</td>
<td>0.035523673</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

...
<table>
<thead>
<tr>
<th>Page</th>
<th>Value1</th>
<th>Value2</th>
<th>Value3</th>
<th>Value4</th>
<th>Value5</th>
<th>Value6</th>
<th>Value7</th>
<th>Value8</th>
</tr>
</thead>
<tbody>
<tr>
<td>4000</td>
<td>10925</td>
<td>0.876640625</td>
<td>0.069975029</td>
<td>0.598009583</td>
<td>0.108518705</td>
<td>0.437334065</td>
<td>0.0809826328</td>
<td>0.1570964062</td>
</tr>
<tr>
<td>4001</td>
<td>10935</td>
<td>0.878077986</td>
<td>0.063145093</td>
<td>0.609646266</td>
<td>0.113959475</td>
<td>0.429623703</td>
<td>0.081776194</td>
<td>0.155381477</td>
</tr>
<tr>
<td>4002</td>
<td>945</td>
<td>0.926655888</td>
<td>0.125693888</td>
<td>0.878818374</td>
<td>0.125121547</td>
<td>1.020756536</td>
<td>0.083066143</td>
<td>0.814096541</td>
</tr>
<tr>
<td>4003</td>
<td>955</td>
<td>0.891035941</td>
<td>0.122836028</td>
<td>0.745111804</td>
<td>0.124078262</td>
<td>0.829430235</td>
<td>0.084579397</td>
<td>0.662949857</td>
</tr>
<tr>
<td>5000</td>
<td>10925</td>
<td>0.88299974</td>
<td>0.084677034</td>
<td>0.593456886</td>
<td>0.115210987</td>
<td>0.416330397</td>
<td>0.084928193</td>
<td>0.157820728</td>
</tr>
<tr>
<td>5001</td>
<td>10935</td>
<td>0.873141764</td>
<td>0.079414786</td>
<td>0.566672712</td>
<td>0.120038794</td>
<td>0.422969085</td>
<td>0.084026855</td>
<td>0.153638064</td>
</tr>
<tr>
<td>5002</td>
<td>945</td>
<td>0.927433943</td>
<td>0.124185901</td>
<td>0.877062193</td>
<td>0.110578162</td>
<td>0.973519841</td>
<td>0.152931183</td>
<td>0.883136199</td>
</tr>
<tr>
<td>5003</td>
<td>955</td>
<td>0.895246771</td>
<td>0.132622172</td>
<td>0.723464144</td>
<td>0.118592826</td>
<td>0.78328543</td>
<td>0.097259609</td>
<td>0.677461497</td>
</tr>
<tr>
<td>6000</td>
<td>10925</td>
<td>0.868452688</td>
<td>0.081632092</td>
<td>0.593856015</td>
<td>0.117722243</td>
<td>0.409546995</td>
<td>0.084746035</td>
<td>0.149730879</td>
</tr>
<tr>
<td>6001</td>
<td>10935</td>
<td>0.859909156</td>
<td>0.080808034</td>
<td>0.587160078</td>
<td>0.111263636</td>
<td>0.403320095</td>
<td>0.084638854</td>
<td>0.152511703</td>
</tr>
<tr>
<td>6002</td>
<td>945</td>
<td>0.930592479</td>
<td>0.126550044</td>
<td>0.861416573</td>
<td>0.137638161</td>
<td>0.922614383</td>
<td>0.139298615</td>
<td>0.838507307</td>
</tr>
<tr>
<td>6003</td>
<td>955</td>
<td>0.906549114</td>
<td>0.147596409</td>
<td>0.735174867</td>
<td>0.11992238</td>
<td>0.76508273</td>
<td>0.140807635</td>
<td>0.660831148</td>
</tr>
<tr>
<td>7000</td>
<td>10925</td>
<td>0.870544925</td>
<td>0.086222249</td>
<td>0.591342523</td>
<td>0.105206655</td>
<td>0.402094681</td>
<td>0.077243135</td>
<td>0.155040428</td>
</tr>
<tr>
<td>7001</td>
<td>10935</td>
<td>0.855099024</td>
<td>0.081402558</td>
<td>0.598674134</td>
<td>0.109468566</td>
<td>0.406263367</td>
<td>0.074922809</td>
<td>0.147166852</td>
</tr>
<tr>
<td>7002</td>
<td>945</td>
<td>0.927360037</td>
<td>0.147229143</td>
<td>0.865221347</td>
<td>0.090647809</td>
<td>0.954451302</td>
<td>0.184982632</td>
<td>0.83332692</td>
</tr>
<tr>
<td>7003</td>
<td>955</td>
<td>0.877341026</td>
<td>0.186090098</td>
<td>0.71059715</td>
<td>0.090015444</td>
<td>0.775763475</td>
<td>0.158036631</td>
<td>0.655956686</td>
</tr>
<tr>
<td>8000</td>
<td>10925</td>
<td>0.8522097</td>
<td>0.093063675</td>
<td>0.584380094</td>
<td>0.096529717</td>
<td>0.387375316</td>
<td>0.067104589</td>
<td>0.150598696</td>
</tr>
<tr>
<td>8001</td>
<td>10935</td>
<td>0.848191816</td>
<td>0.090937404</td>
<td>0.576619835</td>
<td>0.102931742</td>
<td>0.391833188</td>
<td>0.081633452</td>
<td>0.148086656</td>
</tr>
<tr>
<td>8002</td>
<td>945</td>
<td>0.913119199</td>
<td>0.151918936</td>
<td>0.851058225</td>
<td>0.105257524</td>
<td>1.00708705</td>
<td>0.091789955</td>
<td>0.82170685</td>
</tr>
<tr>
<td>8003</td>
<td>955</td>
<td>0.875283996</td>
<td>0.166221922</td>
<td>0.702698557</td>
<td>0.087435755</td>
<td>0.792924897</td>
<td>0.112520502</td>
<td>0.656723336</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9000</td>
<td>10925</td>
<td>0.834984581</td>
<td>0.087296263</td>
<td>0.580724105</td>
<td>0.094959707</td>
<td>0.381400113</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.054179526</td>
<td>0.147888031</td>
<td>0.026197251</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9001</td>
<td>10935</td>
<td>0.837938085</td>
<td>0.082988932</td>
<td>0.583598127</td>
<td>0.097966295</td>
<td>0.391094577</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.068603087</td>
<td>0.1451451</td>
<td>0.035885615</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9002</td>
<td>945</td>
<td>0.908791297</td>
<td>0.161015326</td>
<td>0.870870497</td>
<td>0.084593566</td>
<td>0.937992488</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.148286792</td>
<td>0.857863293</td>
<td>0.070139575</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9003</td>
<td>955</td>
<td>0.852010124</td>
<td>0.146768633</td>
<td>0.72663248</td>
<td>0.08941084</td>
<td>0.774161835</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.111019481</td>
<td>0.685768946</td>
<td>0.045026579</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9009</td>
<td>10915</td>
<td>0.842014343</td>
<td>0.091810824</td>
<td>0.583296428</td>
<td>0.103626218</td>
<td>0.376654465</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.061620194</td>
<td>0.148349293</td>
<td>0.031881147</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10000</td>
<td>10925</td>
<td>0.840630151</td>
<td>0.099537208</td>
<td>0.590759957</td>
<td>0.104437348</td>
<td>0.369670183</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.054629274</td>
<td>0.146488727</td>
<td>0.035272795</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10001</td>
<td>10935</td>
<td>0.845708012</td>
<td>0.110038485</td>
<td>0.588002214</td>
<td>0.100044781</td>
<td>0.378546315</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>0.059734653</td>
<td>0.150528061</td>
<td>0.029726866</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
newdigitGA.R

```r
rm(list=ls())
seednum<-as.integer(200)
set.seed(seednum)

source("deWrapper_constraints.R")
weighting=F
source("binonlyensemblevoWrapper.R")
source("delaystates.R")
source("delaypars.R")
source("replacepars.R")

library(deSolve)
dyn.load("delaymymod.so")
logicvec2int<-function(x) sum(2^(which(rev(x))-1)) #for binary encoding

#prepare new protocols and controls
source("stepprotocol.R")

#For Milnes protocol
tp<-37
beats<-10
pars["T"]<-tp
wholetrace<-read.delim(paste("moreFracBlockMilnes",tp,".txt",sep=""),sep="\t",header=TRUE,
as.is=TRUE)

nMcols<seq(2,dim(wholetrace)[2],2)
concvec<-as.numeric(sapply(strsplit(colnames(wholetrace)[nMcols],"_|nM|uM"),function(x) x[2]))
if(regexpr("uM",colnames(wholetrace)[2])>0)
concvec<-concvec/1000
yIdx<-list() #yIdx is the list of picked index for each episode
startofnewepi <- c(0,which(with(wholetrace, time[-1]-time[-length(time)])<0)) + 1
startofnewepi<- c(startofnewepi, dim(wholetrace)[1]+1) #prepare to get the last chunk (episode)
for(d in 1:length(concvec)){
yIdx[[d]]<-list()
for(i in 1:beats){
  if(i == 1){
    grp<- startofnewepi[i]:startofnewepi[i+1]-1
depcurve<-wholetrace[grp,nMcols[d]]
startpt<-min(11,which(depcurve<=0.5)[1])
if(is.na(startpt)) #dose is so low that no blocking
  startpt<- 11
  #only the 1st episode needs to redefine startpt
pickedpts <- startpt:1000
yIdx[[d]][[i]]<- pickedpts
  }
  #for each episode
rm(startpt)
}
}#for each dose
```

# first get control
# first get the "super control state"
if(1>0){
  # (just closing the gates)
  states["V"]<- -85
  fulltimes<-seq(0,3*60*1000,10000)
  pars["starttime"]<-unclass(as.POSIXct(strptime(date(),"%c")))[1]
  try({
    states<-if(!exists("out")||inherits(out,"try-error")||length(out[,1])!=length(fulltimes) ||
    !all(out[,1]==fulltimes) || any(is.nan(out))){
      return(0)
    }
    states<-controlstates
    controlstates<-states
    rm(out)
  })
  pidx<-match(names(states),colnames(out),nomatch=0)
  states[pidx!=0]<-out[dim(out)[1],pidx]
  source("stepprotocol.R")
  dummy<-stepprotocol(-80, 900,-80,40,0,10000,14060)
  fulltimes<-dummy[[1]]
  peaktimes<-dummy[[2]]
  eventdata<-dummy[[3]]
  states<-controlstates
  beatscontrol<-beats
  maxOvec<-rep(0,beatscontrol)
  controlvec<-0
  for(i in 1:beatscontrol){
    rownum<-dim(wholetrace)[1]/beats  # num of points per episode
    grp<- (i:rownum-(rownum-1)):(i+rownum)
    deptime<-wholetrace$time[grp]
    try({
      out <- ode(states, fulltimes, "derivs", pars,dllname="delaymymod",
        initfunc="initmod", nout=0,rtol=1e-3,atol=1e-6,method="lsoda")
    });
    if(!exists("out")||inherits(out,"try-error")||length(out[,1])!=length(fulltimes) ||
    !all(out[,1]==fulltimes) || any(is.nan(out))){
      return(0)
    }
    # if simulation error
    if(i <= beats){
      idxO<-colnames(out)="O"
      idxV<-colnames(out)="V"
      idxPeakt ime<-out[,1]%in%deptime
      tails<-out[idxPeakt ime, c(1,which(idxO))]}
      maxO<- max(out[idxPeakt ime, idxO])
      maxOvec[i]<-maxO
      controlvec<-c(controlvec, tails[,2])
      rm(tails)
      rm(maxO)
    }
    pidx<-match(names(states),colnames(out),nomatch=0)
    states[pidx!=0]<-out[dim(out)[1],pidx]
  } # for beatscontrol
out_10s <- out
rm(out)
controlvec <- controlvec[-1]

# prepare parameters
sigdigits <- 4
snames <- NULL;
pp <- read.table("nnn.txt", header=T, as.is=T)
pp <- pp[pp$Low != pp$High,]
pnames <- pp$Parameter
negativeidx <- pp$Low < 0
temp <- pp[negativeidx, "Low"]
pp[negativeidx, "Low"] <- -1 * pp[negativeidx, "High"]
pp[negativeidx, "High"] <- -1 * temp
pp[negativeidx, "Initial"] <- -1 * pp[negativeidx, "Initial"]
negativeparnames <- pp$Parameter[negativeidx]

realINITIALTRAIT <- (pp$Initial);
realHIGHTRAIT <- (pp$High);
realLOWTRAIT <- (pp$Low);

HIGHEXPO <- as.numeric(sapply(format(pp$High, scientific=T),
function(x) sapply(strsplit(x,"e"),
function(y) y[2])))
LOWEXPO <- as.numeric(sapply(format(pp$Low, scientific=T),
function(x) sapply(strsplit(x,"e"),
function(y) y[2])))

HIGHDIGIT <- as.numeric(rep(substr(9.999, 1, sigdigits + 1), length(realHIGHTRAIT)))
LOWDIGIT <- as.numeric(rep(substr(1.000, 1, sigdigits + 1), length(realLOWTRAIT)))

HIGHTRAIT <- as.vector(rbind(HIGHDIGIT, HIGHEXPO))
LOWTRAIT <- as.vector(rbind(LOWDIGIT, LOWEXPO))

# genetic parameters
NUM_TRAITS = length(HIGHTRAIT);
realNUM_TRAITS = length(realHIGHTRAIT);
realSIG_FIGS = rep(6, realNUM_TRAITS); # actually decimal point
SIG_FIGS = rep(c(sigdigits-1,0),length(realHIGHTRAIT))
MUTAT_PROB = 0.05;
CROSS_PROB = 0.5;
SELF_ENTERED = 0;
POP_SIZE = 1600;
ELITISM = 1;
DELTA = 500;
EPSILON = -1;
MAX_GENERATION = 200; # more than enough with picked migration
popcountmax = 10; # only keep so many
TOURNAMENTFRAC = 0.2; # may wanna incorporate other selection
chrpernode = 10;
evaluationWorkerNum = POP_SIZE/chrpernode;
realmi <- ceiling(log2(((realHIGHTRAIT-realLOWTRAIT)*(10^realSIG_FIGS)+1)))
mi <- ceiling(log2(((HIGHTRAIT-LOWTRAIT)*(10^SIG_FIGS)+1)))
binCHROM_LENGTH <- sum(mi)

# migration parameters
MIGRATION_INTERVAL=6;  # actually saving interval
NUM_SUBGRPS=4;

subsplitL<-split(1:POP_SIZE, rep(1:NUM_SUBGRPS, each=POP_SIZE/NUM_SUBGRPS))

# print("start initial\n")
# initial population
popcount<-1
trait<-array(0,dim=c(NUM_TRAITS,POP_SIZE, popcountmax));   # only keep the last
generations
for(i in 1:POP_SIZE){
    trait[,i,popcount]<-(runif(NUM_TRAITS)-0.5)*(HIGHTRAIT-LOWTRAIT)+
                      0.5*(HIGHTRAIT+LOWTRAIT)
}

# go real
realtrait<-array(0, dim=c(realNUM_TRAITS, POP_SIZE, popcountmax));  # twosice the size on
the se
for(i in 1:POP_SIZE){
    for( j in seq(1,NUM_TRAITS,2)){
        realtrait[(j+1)/2,i,popcount]<- as.numeric(
            paste(substr(trait[j,i,popcount],1,sigdigits+1),"e",
                   round(as.numeric(trait[j+1,i,popcount])),sep="")
        )
    }
}

# fix out-of-bounds traits
realtrait[,i,popcount]<-apply(realtrait[,i,popcount],2, function(x)
{temp<-pmin(x,realHIGHTRAIT);
 temp<-pmax(temp,realLOWTRAIT);
 return(temp) })

# go back
for(i in 1:POP_SIZE){
    trait[,i,popcount]<- as.numeric(sapply(format(realtrait[,i,popcount], scientific=T),
          function(x) sapply(strsplit(x,"e"),
                      function(y) y)))
}

# prepare parallel computing
cl <- makeCluster()
clusterApply(cl, (1:evaluationWorkerNum)+seednum, set.seed)
clusterEvalQ(cl, library(deSolve))
clusterEvalQ(cl, source("bincreateNewGeneration.R"))
clusterEvalQ(cl, dyn.load("delaymymod.so"))

clusterExport(cl, c("pnames","snames","states","pars","chrpernode"))   # for evaluation
        # for IC50s
clusterExport(cl, c("subsplitL","binCHROM_LENGTH","SIG_FIGS","MUTAT_PROB","CROSS_PROB","ELITISM","TOURNAMENTFRAC","HIGHTRAIT","LOWTRAIT","mi","logicvec2int"));  # for evolution
clusterExport(cl, c("negativeparnames","controlstates","controlvec","wholetrace","tp","beats","yIdx"))   # for Milnes protocol
system("rm -rf ensembleGAprocess")
outputfile<="ensembleGAprocess"

# prepare main loop
fitness_bar<-matrix(1e50, nrow=POP_SIZE, ncol= popcountmax)
bestfitness <- rep(0, popcountmax)
currentdelete <- 0

# main loop
while (popcount <= MAX_GENERATION) {
    recycledpopcount <- popcount%%popcountmax
    if (recycledpopcount == 0) {recycledpopcount = popcountmax}
    previouspopcount <- recycledpopcount - 1
    if (previouspopcount == 0) {previouspopcount = popcountmax}
    print(recycledpopcount);
    # go real
    for (i in 1:POP_SIZE) {
        for (j in seq(1, NUM_TRAITS, 2)) {
            realtrait[(j+1)/2, i, recycledpopcount] <- as.numeric(
                paste(substr(trait[j, i, recycledpopcount], 1, sigdigits+1), "e",
                round(as.numeric(trait[j+1, i, recycledpopcount])), sep = "")
            )
        }
    }
    if (1>0) {
        # boundary check
        realtrait[, recycledpopcount] <- apply(realtrait[, recycledpopcount], 2, function(x) {
            temp <- pmin(x, realHIGHTRAIT);
            temp <- pmax(temp, realLOWTRAIT);
            return(temp)
        })
        # go back
        for (i in 1:POP_SIZE) {
            trait[, i, recycledpopcount] <- as.numeric(sapply(format(realtrait[, i, recycledpopcount], scientific=T),
                function(x) sapply(strsplit(x, "e"),
                function(y) y))
        }
    }
}

# evaluation
# fixed precision <- signif(trait[, recycledpopcount], digits=10)  # so that the saved
current generation is closer to the full digits.
pop <- realtrait[, recycledpopcount]
pop <- signif(pop, digits=sigdigits)
clusterExport(cl, "pop")
errorlist <- clusterApplyLB(cl, 1:evaluationWorkerNum, deWrapper, weighting)
errormat <- do.call(rbind, errorlist)
errormat <- errormat[order(errormat[, 1]),]
fitness_bar[, recycledpopcount] <- errormat[, 2]
clusterEvalQ(cl, rm(pop))

# report
# cat(paste("generation ", popcount, " lowest error is ", min(errormat[, 2]), sep=""), sep="\n" , file=outfile, append=T);
# cat(paste("generation ", popcount, " lowest error is ", min(errormat[, 2]), sep=""), sep="\n")
bestfitness[popcount] <- (1/min(errormat[, 2])) + 0.1
vecmin <- aggregate(errormat[, 2], list(rep(1:4, each=length(errormat[, 2])/4)), min)
cat(paste("generation ", popcount, " lowest error is ", min(vecmin[, 2]), collapse=""),
    sep=""),
sep="\n",file=outputfile,append=T
}

# replacing dead ones/migration
if(popcount%%MIGRATION_INTERVAL ==0){
  source("digitmigrationreset.R")
  currentdelete<- newdelete
}

# migration/saving
if(popcount%%MIGRATION_INTERVAL ==0){
  # save the current generation
  pop<-realtrait[,recycledpopcount]
  pop<-signif(pop,digits=sigdigits)
  write.table(pop,"currentgeneration",quote=F,col.names=F,row.names=F)
  write.table(fitness_bar[,recycledpopcount],"currenterror", quote=F,col.names=F,row .names=F)
}

# parallel evolution
currenttrait<-trait[,recycledpopcount]
currentfit<-fitness_bar[,recycledpopcount]
clusterExport(cl,c("currenttrait", "currentfit"))
newtrait<-clusterApplyLB(cl,1:NUM_SUBGRPS, ensemblevoWrapper)
newtrait<-do.call(cbind,newtrait)
popcount<-popcount+1
recycledpopcount<-popcount%%popcountmax
if(recycledpopcount == 0){recycledpopcount = popcountmax}
trait[,recycledpopcount]<-newtrait
if(popcount > (DELTA+1)){
  if(max(abs(bestfitness[2:popcountmax]- bestfitness[1:(popcountmax -1)]))<=EPSILON)
  {
    break;
  }
}
 # main loop
stopCluster(cl);
getMilneserror <- function(pars, states, numerrors, weighting, controlstates, controllvec, ds, yIdx, temp, beats, usespecerr = FALSE) {
  fval <- 0
  errorvec <- rep(0, numerrors)
  source("stepprotocol.R")
  specerr <- 0
  negativeerror <- 0
  pars["T"] <- temp;
  err <- 0
  killmyself <- 0

  dummy <- stepprotocol(-80, 900, -80, 40, 0, 10000, 14060)
  fulltimes <- dummy[1]
  peaktimes <- dummy[2]
  eventdata <- dummy[3]

  # drug
  nMcols <- seq(2, dim(ds)[2], 2)
  concvec <- as.numeric(sapply(strsplit(colnames(ds)[nMcols], "_[nM|uM]"), function(x) x[2]))
  if (regexpr("uM", colnames(ds)[2]) > 0)
    concvec <- concvec * 1000
  beatsdrug <- beats
  maxOvec <- rep(0, beats)
  yPred <- list()
  yO <- list()
  yTime <- list()
  for (d in 1:length(concvec)) {
    # initializing
    states <- controlstates
    states["D"] <- concvec[d]

    # closing the channel in the presence of drug?
    if (1 > 0) {
      states["V"] <- -80
      closingfulltimes <- seq(0, 3 * 60 * 1000, 10000)
      pars["starttime"] <- unclass(as.POSIXct(strptime(date(), "%c")))[1]
      try({out <- ode(states, closingfulltimes, "derivs", pars, dllname="delaymymod",
            initfunc="initmod", nout=0, rtol=1e-3, atol=1e-6, method="lsoda")});
      pidx <- match(names(states), colnames(out), nomatch = 0)
      states[pidx != 0] <- out[dim(out)[1], pidx]
      rm(out)
    }
    # done closing the channel in the presence of drug
    yPred[[d]] <- list()
    yO[[d]] <- list()
    yTime[[d]] <- list()
    startofnewepi <- c(0, which(with(ds, time[-1] - time[-length(time)]) < 0)) + 1
    startofnewepi <- c(startofnewepi, dim(ds)[1] + 1)  # prepare to get the last chunk (episode)
    for (i in 1:beatsdrug){
      
    
  }
rownum <- dim(ds)[1]/beats  # num of points per episode
# define rownum/points for each beats and each dose
# ds will always be 100*10 rows and beats always 10, so rownum always 10
grp <- (1+rownum-(rownum-1)):i  # grp is always 100 long

deptime <- ds$time[grp][yIdx[d][i]]

pars["starttime"] <- unclass(as.POSIXct(strptime(date(), "%c"))[1]
# try(out <- ode(states, fulltimes, modelfun, pars, rtol=1e-3, method="lsoda",
events=list(data=eventdata)))
try({
  out <- ode(states, fulltimes, "derivs", pars, dllname="delaymymod",
             initfunc="initmod",
             nout=0, rtol=1e-3, atol=1e-6, method="lsoda", events=list(data=eventdata))
});
if(!exists("out") || inherits(out, "try-error") || length(out[,1]) != length(
  fulltimes) || !all(out[,1] == fulltimes) || any(is.nan(out))){
  return(0)
}

# if simulation error
if(i <= beats){
  idxO <- colnames(out) == "O"
  idxV <- colnames(out) == "V"
  idxPeaktime <- out[,1] %in% deptime
  tails <- out[idxPeaktime, c(1, which(idxO))]
  maxO <- max(out[idxPeaktime, idxO])
  maxOvec[i] <- maxO
  yTime[d][i] <- deptime
  yPred[d][i] <- tails[,2]/controlvec[grp][yIdx[d][i]]
  yO[d][i] <- ds[grp,nMcols[d]][yIdx[d][i]]
  fval <- fval + sum((yPred[d][i] - yO[d][i])^2)
  rm(tails)
  rm(maxO)
}  

#for beatsdrug
idxO <- colnames(out) == "O"
idxV <- colnames(out) == "V"
idxPeaktime <- out[,1] %in% deptime
#negative constraints violation. Note that V is in here and it CAN be negative
allval <- out[,c(-1,-which(idxV))]
negativeerror <- sum(apply(allval, 2, function(x) mean(pmin(x, 0)^2)))
# errorvec <- errorvec + c(err, negativeerror, specerr)
rm(out)

#for concvec

if(usespecerr){
  numpt <- length(unlist(yO))
significantidx <- sapply(yO, function(x) any(unlist(x) <= 0.5))  #which doses
  have the first episode
  begining <- sapply(yO, function(x) sapply(x, function(z) z[1]))  #actually the 11th
          (previously 2nd) point
  beginingP <- sapply(yPred, function(x) sapply(x, function(z) z[1]))
}
yOdiff1 <- (begining[1,] - begining[10,]) / begining[1,]
yPdiff1 <- (beginingP[1,] - beginingP[10,]) / beginingP[1,]
yOdiff2 <- (begining[1,] - begining[2,]) / begining[1,]
yPdiff2 <- (beginingP[1,] - beginingP[2,]) / beginingP[1,]

ending <- sapply(yO, function(x) sapply(x, function(z) z[length(z)]))
endingP <- sapply(yPred, function(x) sapply(x, function(z) z[length(z)]))
yOdiff3 <- (ending[1,] - ending[10,]) / ending[1,]
yPdiff3 <- (endingP[1,] - endingP[10,]) / endingP[1,]

yOdiff1 <- pmax(0, yOdiff1); yOdiff2 <- pmax(0, yOdiff2); yOdiff3 <- pmax(0, yOdiff3)

specerr <- specerr + sum(0.2*numpt/length(concvec)*(yPdiff1-yOdiff1)[significantidx]^2) +
sum(0.2*numpt/length(concvec)*(yPdiff2-yOdiff2)[significantidx]^2)

fval <- fval+err+negativeerror+specerr
rm(err)

output <- list(fval, errorvec, yPred, yO, specerr, yTime)
ODEs:
d(dIC1)/dt=1/unnamed*(-ReactionFlux1+ReactionFlux5)
d(dIC2)/dt=1/unnamed*(ReactionFlux1-ReactionFlux4+ReactionFlux6)
d(dC1)/dt=1/unnamed*(-ReactionFlux2-ReactionFlux5)
d(dC2)/dt=1/unnamed*(ReactionFlux2-ReactionFlux3-ReactionFlux6)
d(dO)/dt=1/unnamed*(ReactionFlux3-ReactionFlux7-ReactionFlux9)
d(dIO)/dt=1/unnamed*(ReactionFlux4+ReactionFlux7-ReactionFlux8)
d(dObound)/dt=1/unnamed*(ReactionFlux8+ReactionFlux11)
d(dCbound)/dt=1/unnamed*(-ReactionFlux10+ReactionFlux11)

Fluxes:
ReactionFlux1=A11*exp(B11*V)*IC1*exp((T-20)*log(q11)/10)-A21*exp(B21*V)*IC2*exp((T-20)*log(q21)/10)
ReactionFlux2=A1*exp(B1*V)*C1*exp((T-20)*log(q1)/10)-A2*exp(B2*V)*C2*exp((T-20)*log(q2)/10)
ReactionFlux3=A31*exp(B31*V)*C2*exp((T-20)*log(q31)/10)-A41*exp(B41*V)*O*exp((T-20)*log(q41)/10)
ReactionFlux4=A3*exp(B3*V)*IC2*exp((T-20)*log(q3)/10)-A4*exp(B4*V)*IO*exp((T-20)*log(q4)/10)
ReactionFlux5=A51*exp(B51*V)*C1*exp((T-20)*log(q51)/10)-A61*exp(B61*V)*IC1*exp((T-20)*log(q61)/10)
ReactionFlux6=A52*exp(B52*V)*C2*exp((T-20)*log(q52)/10)-A62*exp(B62*V)*IC2*exp((T-20)*log(q62)/10)
ReactionFlux7=A53*exp(B53*V)*O*exp((T-20)*log(q53)/10)-A63*exp(B63*V)*IO*exp((T-20)*log(q63)/10)
ReactionFlux8=Kf2*Ku*exp(n*log(D))/(exp(n*log(D))+halfmax)*IO-Ku*A53*exp(B53*V)*exp((T-20)*log(q53)/10)/(A63*exp(B63*V)*exp((T-20)*log(q63)/10))*IObound
ReactionFlux9=Kf*Ku*exp(n*log(D))/(exp(n*log(D))+halfmax)*O-Ku*Obound
ReactionFlux10=Kt/(1+exp(-(V-Vhalf)/6.789))*Cbound-Kt*Obound
ReactionFlux11=Kt2/(1+exp(-(V-Vhalf)/6.789))*Cbound-Kt2*IObound

ParameterValues:
B1=4.631e-05
B2=-0.004226
A2=4.986e-06
A1=0.0264
A3=0.001214
B3=0.008516
A4=1.854e-05
B4=-0.04641
A5=0.03776
B6=-6.304e-08
A6=7.057e-09
B7=9.502e-09
Kf=0.05
Kf2=0
Kt=3.5e-05
Kt2=3.5e-05
Vhalf=1
Ku=0.01
T=20
A11=0.0007868
B11=1.535e-08
q11=4.942
A21=5.455e-06
B21=-0.1688
q21=4.156
q1=4.843
q2=4.23
q31=4.22
q41=1.459
A31=0.005509
B31=7.771e-09
A41=0.001416
B41=-0.02877
q3=4.962
q4=3.769
q51=5
q61=5.568
A51=0.4492
B51=0.008595
A61=0.01241
B61=0.1725
q53=2.412
q63=5.682
A53=0.149
B53=0.004668
A63=0.008978
B63=0.02215
q52=4.663
q62=5
A52=0.3181
B52=3.613e-08
A62=0.3226
B62=-0.0006575
n=1
halfmax=1
unnamed=1
trap=1

InitialConditions:
IC1=0
IC2=0
C1=1
C2=0
O=0
IO=0
IObound=0
Obound=0
Cbound=0
D=0
Ψ=-70
This file contains example bounds for dynamic hERG drug binding parameters.
Results were obtained using the following bounds:
Ku lower bound = 2e-5 for chlorpromazine, cisapride, diltiazem, mexiletine,
omdansetron, quinidine, ranolazine, sotalol
Ku lower bound = 2e-8 for dofetilide
halfmax lower bound = 0.1 for bepridil, dofetilide, ondansetron, ranolazine,
sotalol, verapamil
halfmax upper bound = 1e+6 for diltiazem, quinidine, terfenadine
These bounds yielded results similar to those obtained with the example bounds.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Low</th>
<th>Initial</th>
<th>High</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kf</td>
<td>1e-8</td>
<td>5</td>
<td>1e+8</td>
</tr>
<tr>
<td>Ku</td>
<td>1e-9</td>
<td>1e-3</td>
<td>1</td>
</tr>
<tr>
<td>n</td>
<td>0.5</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>halfmax</td>
<td>0.001</td>
<td>0.15</td>
<td>1e+9</td>
</tr>
<tr>
<td>Vhalf</td>
<td>-200</td>
<td>-8</td>
<td>-1</td>
</tr>
</tbody>
</table>
processcompileDDEs.R

ODEs <- readChar("newODEs.txt", file.info("newODEs.txt")$size)
tokens <- strsplit(ODEs, "\n\n")
system("rm -f delaymod.c delaymodfun.R delaypars.R delaystates.R")

# for equations
equations <- substring(tokens[[1]][1], 1, 7)
equationvec <- strsplit(equations, "\n")
equationprocessed <- sub("d\((.+)\)/dt(.+)", "d\1\2", equationvec[[1]])
species <- sub("(.+)=.+", "\1", equationprocessed, perl = T)
equationsval <- sub("d(.+)=(.+)\","\2", equationprocessed, perl = T)
names(equationsval) <- species

# for fluxes
fluxes <- substring(tokens[[1]][2], 9)
fluxesvec <- strsplit(fluxes, "\n")
fluxesval <- sub("(.+)\s?=\s?(.+)\", "\2", fluxesvec[[1]], perl = T)
names(fluxesval) <- sub("(.+)\s?=\s?(.+)\", "\1", fluxesvec[[1]], perl = T)

# for initials. Note sometimes there are more species in here than the equations
states <- substring(tokens[[1]][4], 1, 20)
statesvec <- strsplit(states, "\n")
statespecies <- sub("(.+)=.+", "\1", statesvec[[1]], perl = T)
idxlag <- regexpr("lag\d+.+", statespecies, perl = T) > 0
lagspecies <- statespecies[idxlag]
lagstatesvec <- statesvec[[1]][idxlag]
statesvec <- statesvec[[1]][!idxlag]
tempspecies <- paste("d", species, sep = "")
idx <- tempspecies%in%species

# for parameters. Note I added two additional parameters here
pars <- substring(tokens[[1]][3], 1, 8)
# add two additional parameters
pars <- paste(pars, "timeout=30
starttime=0", sep = "")
parsvec <- strsplit(pars, "\n")
parsname <- sub("(.+)\s?=\s?(.+)\", "\1", parsvec[[1]], perl = T)

# prepare for c code

# lagged species
ytau <- paste("ytau", 0:(length(lagspecies) - 1), "", sep = "")
names(ytau) <- lagspecies

y <- paste("y", 0:(length(tempspecies) - 1), "", sep = "")
names(y) <- tempstatesvec(2)

# replace lag species in fluxesval
for (f in 1:length(ytau)){
  replacement <- paste("\1", ytau[f], "\2", sep = "")
  replaced1 <- paste("(\f+\")", names(ytau)[f], "(\f+\")", sep = "")
  replaced2 <- paste("(^\")", names(ytau)[f], "(^\")", sep = "")
  replaced3 <- paste("(\")", names(ytau)[f], "(\")", sep = "")
}
fluxesval <- gsub(replaced1, replacement, fluxesval, perl=T)
fluxesval <- gsub(replaced2, replacement, fluxesval, perl=T)
fluxesval <- gsub(replaced3, replacement, fluxesval, perl=T)
}

# replace species in fluxesval
for(f in 1:length(y)){
    replacement <- paste("\1", y[f], "\2", sep="")
    replaced1 <- paste("([+-/*\()\], names(y)[f], "([+-/*\)])", sep="")
    replaced2 <- paste("(\^\s\), names(y)[f], "([+-/*\^\s]))", sep="")
    replaced3 <- paste("([+-/*\()\], names(y)[f], "($)", sep="")
    fluxesval <- gsub(replaced1, replacement, fluxesval, perl=T)
    fluxesval <- gsub(replaced2, replacement, fluxesval, perl=T)
    fluxesval <- gsub(replaced3, replacement, fluxesval, perl=T)
}

# end replacing fluxesval

# replace equationsval
for(f in 1:length(fluxesval)){
    replacement <- paste("\1\2", sep="")
    replaced <- paste("^(.*)", names(fluxesval)[f], "((\D.*)|"))", sep="")
    equationsval <- gsub(replaced, replacement, equationsval, perl=T)
}

# end replacing

# writing files
sink("delaymodelfun.R")
cat("modelfun <- function(Time, State, Pars){", sep = "\n")
cat('currenttime<-unclass(as.POSIXct(strptime(date(),"%c"))[1])
');
cat("if(currenttime-Pars["starttime"]>=Pars["timeout"]){
    stop("timeout!");
}
"
);
cat("with(as.list(c(State, Pars)), {", sep = "\n")

# lag definition
laglist <- strsplit(lagstatesvec,"=")
if(length(laglist)>0){
    for(l in 1:length(laglist)){
        token <- strsplit(laglist[l][1], ",\)\[1][1]
        lagtime <- substr(token,4,nchar(token))
        cat(paste("if(Time<", lagtime,"),", sep=""));
        cat("\n");
        cat(paste(laglist[l][1], "<-", laglist[l][2], sep=""));
        cat("\n");
        cat("elseif\n")
        idx <- which(statespecies==strsplit(laglist[l][1], ",\)\[1][1])
        cat(paste(laglist[l][1], "<-lagvalue(Time\-", lagtime, ",", idx, "),", sep=""));
        cat("\n")
    }
}
cat(fluxes, sep = "\n")
cat(equationprocessed, sep = "\n")
}
for(i in tempspecies[[!idx]]){
    cat("
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
asurer
now, so nr[l]

```r
cat("double_ytau", l, ", 1," [l] = \{"laglist[[1]] [2]," ];\n”, sep=""); #use ytautemp to store individual lag species
cat("double_T", l, ", 0 = t - lagtime,\n”, sep=""); cat("if (\n”, sep="");
cat("yout", l, ", nr[l], Nout[l], \"ytau[l],\n”, sep=""); cat("\n"); cat("ytau[l-1] = ytau[l];\n”, sep="";")
```

```r}
# for l

for(q in 1:length(equationsval)){
  qname<-names(equationsval)[q]
  qname<-substr(qname,2,2000) # remove the first "d"
  yidx<-which(names(y)==qname)
  cat(paste("ydot", yidx-1," = \"equationsval[q],\n", sep=""), sep="\n")
}
```

```r

#add species not having equations
idx<-tempspecies%in%species
if(any(!idx)){
  for(i in 1:sum(!idx)){
    cat(paste("ydot", q-1+i," = 0;\n", sep=""), sep="\n")
  }
}
```

```r
}
```

```r
# if there are lagged species
if(length(laglist)>0){
  for(l in 1:length(laglist)){
    cat("yout[l-1," ytau[l-1," ];\n", sep="")
  }
}
```

```r
}
```

```r
# compile
system("R CMD SHLIB delaymymod.c");
```

```r
sink("delaypars.R")
cat("pars<-c(" , sep = "\n")
cat(parsvec[[1]], sep = ",")
cat("\n")
sink()
```

```r
sink("delaystates.R")
cat("states<-c(" , sep = "\n")
cat(statesvec, sep = ",")
cat("\n")
sink()
```
replacepars.R

replacepars <- function(oldpar, currentgenerationfilename, currenterrorfilename, nnnfilename, extraidx){

  # current nnn
  pp <- read.table(nnnfilename, header=T, as.is=T)
  pp <- pp[pp$Low != pp$High,]
  # sometimes currentn
n and current model (oldpar) do not match
  idx <- (pp$Parameter) %in% names(oldpar)
  pp <- pp[idx,]

  # current best parameters
  tg <- read.table(currentgenerationfilename, header=F, as.is=T)
  fitness_bar <- read.table(currenterrorfilename, header=F, as.is=T)
  indidx <- which(fitness_bar[,1] == min(fitness_bar[,1]))[1]
  tg <- tg[idx,]
  fitness_bar <- fitness_bar[idx,]
  pnames <- pp$Parameter
  snames <- NULL;
  ind <- tg[, indidx]
  oldpar[pidx != 0] <- ind[pidx]
  negativeidx <- pp$Low < 0
  negativeparnames <- pp$Parameter[negativeidx]
  oldpar[negativeparnames] <- oldpar[negativeparnames] * (-1)

  # microscopic reversibility
  if(!is.na(oldpar["A61"])){
    oldpar["A61"] = oldpar["A11"]*oldpar["A62"]*oldpar["A2"]*oldpar["A51"]/oldpar["A1"]*oldpar["A52"]*oldpar["A21"]
  }
  if(!is.na(oldpar["B61"])){
    oldpar["B61"] = oldpar["B11"]+oldpar["B62"]+oldpar["B2"]+oldpar["B51"] - (oldpar["B1"] + oldpar["B52"] + oldpar["B21"])  
  }
  if(!is.na(oldpar["q61"])){
    oldpar["q61"] = oldpar["q11"]*oldpar["q62"]*oldpar["q2"]*oldpar["q51"]/oldpar["q1"]*oldpar["q52"]*oldpar["q21"]
  }
  if(!is.na(oldpar["A63"])){
    oldpar["A63"] = oldpar["A4"]*oldpar["A62"]*oldpar["A31"]*oldpar["A53"]/oldpar["A41"]*oldpar["A52"]*oldpar["A3"]
  }
  if(!is.na(oldpar["B63"])){
    oldpar["B63"] = oldpar["B4"]*oldpar["B62"]*oldpar["B31"]*oldpar["B53"] - (oldpar["B41"] + oldpar["B52"] + oldpar["B31"])
  }
  if(!is.na(oldpar["q63"])){
    oldpar["q63"] = oldpar["q4"]*oldpar["q62"]*oldpar["q31"]*oldpar["q53"]/oldpar["q41"]*oldpar["q52"]*oldpar["q3"]
  }
  oldpar <- signif(oldpar, digits=4)
  # backward compatibility: old scripts didn't signif the calculated parameters
  return(oldpar)
}
runsnowmpisge.sh

#!/bin/sh
#$ -cwd
#$ -pe orte 161
#$ -j y
#$ -N testsnowmpisge
#$ -l s_rt=24:00:00
#$ -R y
#$ -l h_rt=26:00:00
#$ -l h_vmem=256M

module add openmpi-x86_64

cat $PE_HOSTFILE
time mpirun -mca plm_base_verbose 30 -mca orte_base_help_aggregate 0 --debug-daemons ~/mylibs/R/snow/RMPISNOW < newdigitGA.R
stepprotocol.R

stepprotocol<-function(holdv, holdt, conv, cont, testv, testt, gapt){
    # generic step protocol consisting a holding step, a conditioning step, and a testing step
    # gapt is the gap from the end of the this pulse to the start of next pulse

times<-c(0, holdt, cont+holdt, cont+holdt+testt)
voltages<-c(holdv, conv, testv, holdv)  # note the unit is mv !!!
eventdata<-data.frame(var="V", time=times, value=voltages, method="replace")

peaktimes<-seq(holdt+cont, holdt+cont+testt, 1)  # for obtaining the peak; should be within 1 s after depolarization

finaltimes<- holdt+cont+testt+gapt
initialtime<- 0  # included in times already

outputtimes<-sort(unique(c(initialtime, peaktimes, finaltimes)))
fulltimes<-sort(c(times, cleanEventTimes(outputtimes, times)))  # times is actually eventtimes

output<-list(fulltimes, peaktimes, eventdata)