Uncertainty Quantification in Computational Models of Atrial Fibrillation

Eugene TY Chang, Richard H Clayton

Department of Computer Science,
University of Sheffield

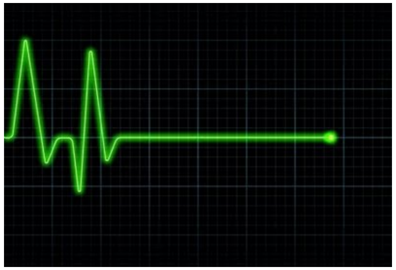
7th July 2015

INSIGNEO Institute for *in silico* Medicine

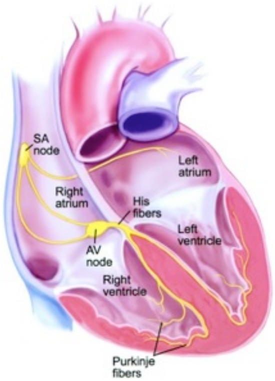
The University Of Sheffield.

Sheffield Teaching Hospitals NHS
NHS Foundation Trust

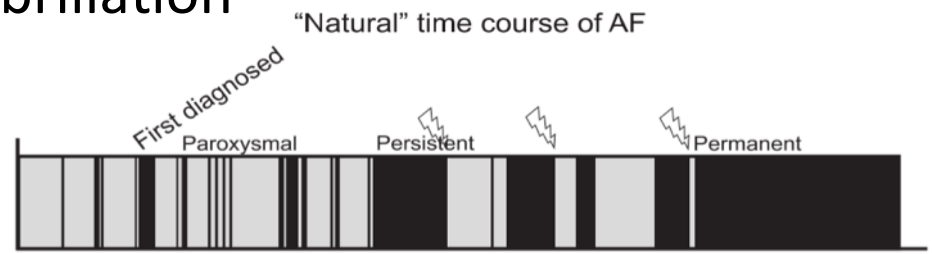
Atrial Fibrillation (AF)



- Atrial Fibrillation (AF): dissynchronous electrical activity in the atria leading to ineffective atrial contraction



- Disease progression from spontaneous intermittent episodes to permanent fibrillation



Schotten, U., S. Verheule, P. Kirchhof, and A. Goette, (2011), Physiological Reviews, 91(1): p.265-325.

Motivation

- Electrical activity in the heart tissue is simulated using PDEs coupled to ODEs representing cardiac cellular activity
- Cardiac cell (and tissue) models have many input parameters
- ***Uncertainty propagation****, aka sensitivity analysis*
  - How is uncertainty in model outputs attributed to uncertainty from (different) inputs
- Doing a complete sensitivity analysis via MCMC is computationally demanding!

Principal Idea

- Develop tools to carry out Uncertainty Quantification (UQ) in models of AF
- Gaussian Process (GP) emulators are models of models, or `meta models', capable of rapid sensitivity analysis
- Proof of concept studies in single cell ODE models thus far:
  - LR1: Luo-Rudy 1991 (guinea pig ventricular)
  - CRN: Courtemanche Ramirez Nattel 1998 (human atrial)
- Initial PDE + ODE simulations in 2D (Nektar++)

Gaussian Process: Introduction

- If our model is described by

$$\mathbf{y} = f(\mathbf{x})$$

- Then if *f* is a Gaussian process, then evaluating f(***x***) for a set of inputs ***x*** yields outputs ***y*** that are a probability density with mean and variance.
- A GP can be formulated as follows

$$f(\mathbf{x}) = h(\mathbf{x})^T \beta + Z(\mathbf{x})$$

where the first term is a mean function, and the second is a covariance with a mean of zero.

- So the process of building a GP emulator involves choosing forms for the mean and covariance functions along with suitable hyper-parameters.

Fitting a GP to design data (1)

- With a Bayesian approach, we can choose prior forms for the mean and covariance functions, and their hyper-parameters.

$$f(\mathbf{x}) = h(\mathbf{x})^T \beta + Z(\mathbf{x}) \quad (5)$$

- A linear form for the mean is commonly used

$$h(\mathbf{x})^T \beta = \beta_0 + \beta_1 x_1 + \ldots + \beta_P x_P \quad (6)$$

- Where $\mathbf{x} = (x_1, x_2, \ldots, x_P)$ are P inputs or parameters, and β are coefficients.
- With covariance

$$Cov(Z(\mathbf{x}), Z(\mathbf{x}') | \sigma^2, \delta) = \sigma^2 exp \left\{ -\sum_{p=1}^{P} \left(\frac{x_p - x'_p}{\delta_p} \right) \right\} \quad (7)$$

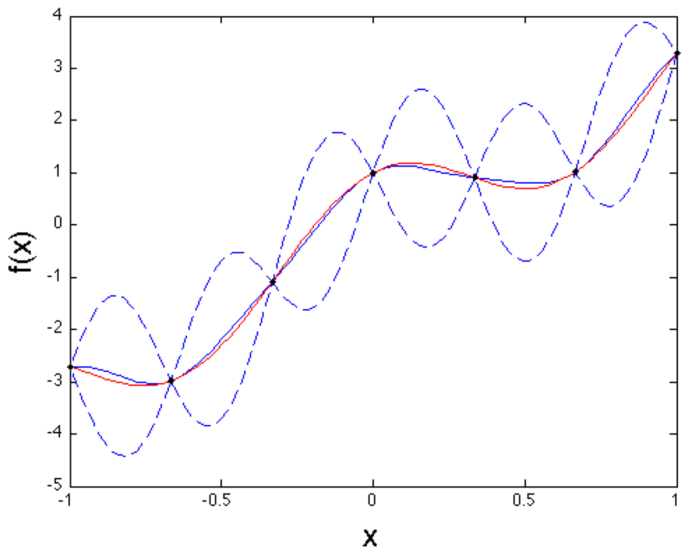
Fitting a GP to design data (2)

- A posterior distribution of the GP emulator can be calculated, conditional on **both** the training **and** the hyper-parameters $\{\beta, \sigma^2, \delta\}$ and is described by mean and covariance functions.
- This process also yields a posterior distribution of $\{\beta, \sigma^2, \delta\}$, $\pi^*(\beta,\sigma^2,\delta)$ but for practical purposes we would like to specify these as variables rather than distributions.
- A useful simplifying trick is to assume *weak prior* information on β and σ^2, so that
- $\pi^*(\beta,\sigma^2,\delta) = k\, \sigma^{-2}\, \pi(\delta)$, where π represents a prior distribution.
- This greatly simplifies the calculation of the posterior distribution $\pi^*(\delta)$, and a suitable value for δ, \hat(δ), can be chosen by maximising $\pi^*(\delta)$.
- \hat(δ) can then be used to calculate values for \hat(σ²)and \hat(β).

INSIGNEO

Fitting a Gaussian Process to design (training) data

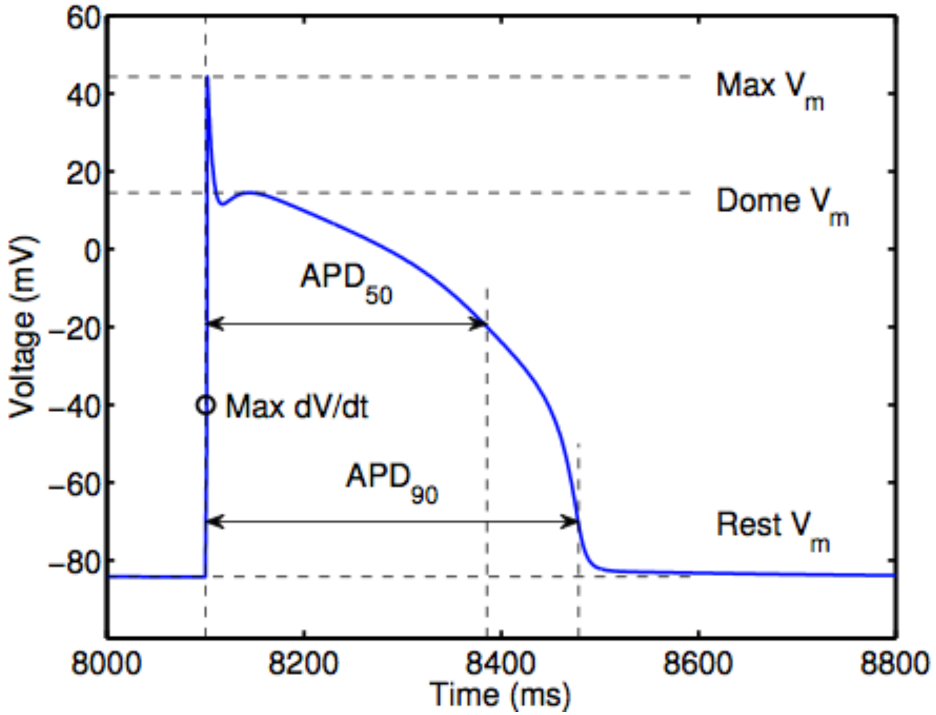
- Example from http://mucm.aston.ac.uk/MUCM/MUCMToolkit/index.php?page=MetaFirstExample.html



- The emulator is an uncertain function
  - At the design points the GP mean matches the simulator output, and the variance is zero.
  - Elsewhere, the variance indicates *uncertainty* about the output.

LR1: Six inputs and eight outputs

- Proof of concept study.
- LR1 code auto generated from CellML.
- Six input parameters: maximum conductances G_{Na}, G_{si}, G_K, G_{K1}, G_{Kp}, G_b.
- Eight outputs: dV_m/dt_{max}, $V_{m,max}$, $V_{m,dome}$, APD, $V_{m,rest}$, plateau duration, APDr slope, Di_{min}.
- Simulation of 10 electrical 'beats'

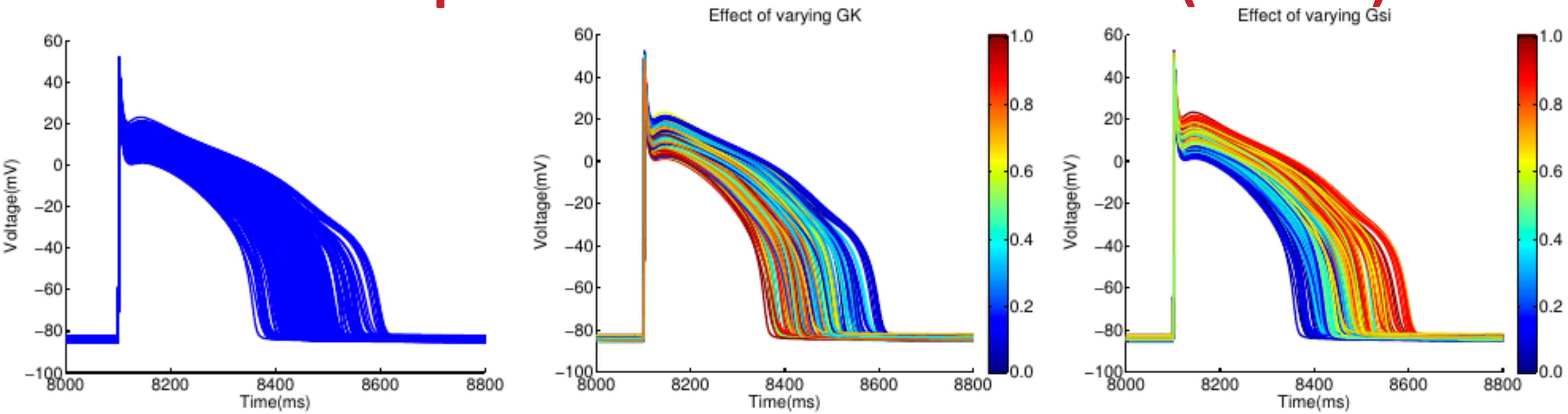


- First six outputs derived from 9th beat, pacing at 1000 ms cycle length.
- Last two outputs determined from dynamic restitution curve, with 10th beat at progressively reducing cycle length

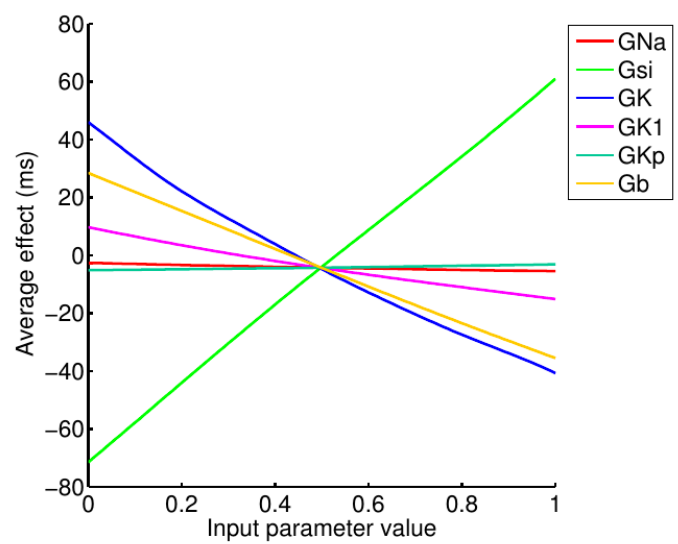
LR1: Six inputs and eight outputs

- Design data (inputs and outputs) obtained from 200 simulator runs, with Latin hypercube sampling of parameter space.
- One emulator built for each output.
- Each emulator validated using an additional set of 20 simulator runs.
- New emulator built with combined design and validation data.
- Results:
  - Mean effect of each input on mean value of each output.
  - Mean and variance of emulator output when all inputs are Gaussian with mean 0.5 and variance 0.02.
  - Contribution of variance in each input to variance of each output.

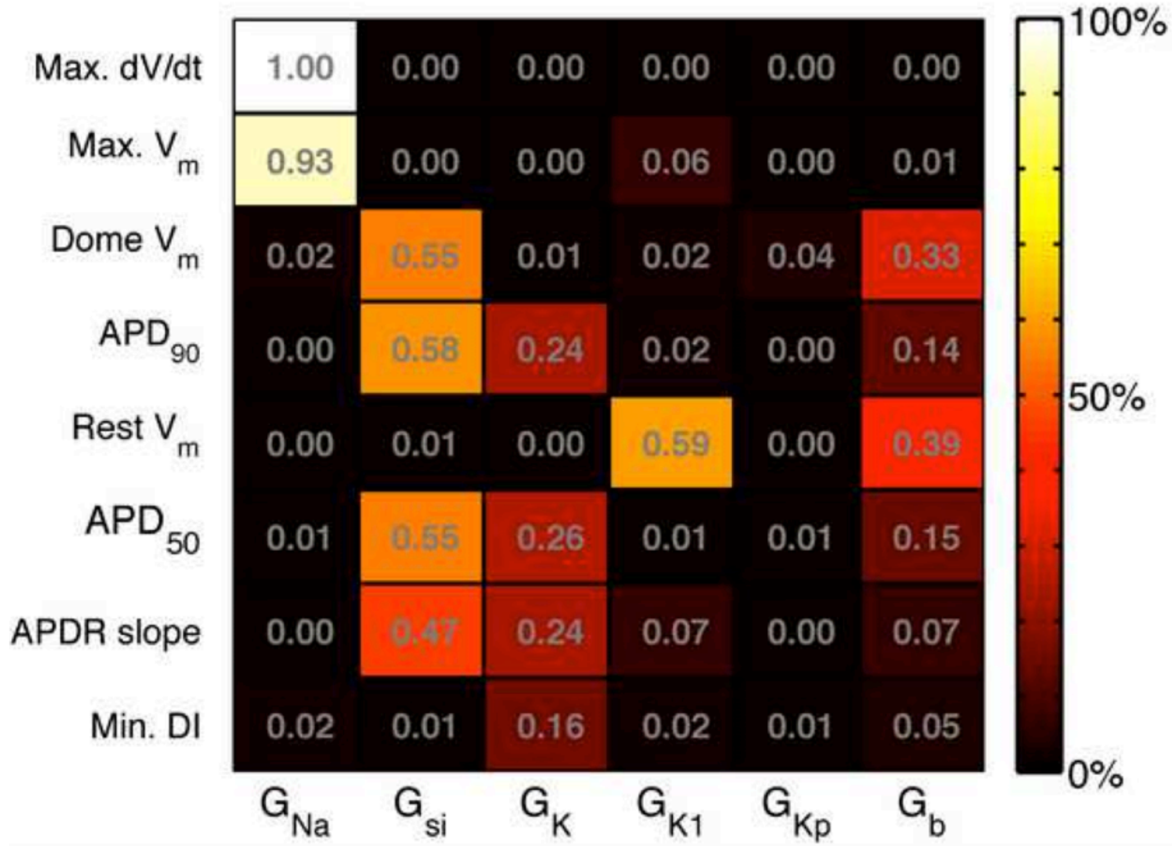
Effect of G_K and G_{si} on action potential duration (APD)



- Increasing G_{si} increases APD (more inward current), and increasing G_K reduces APD (more outward current).
- This can be seen in the design data (above)
- Mean Effect output of GP emulator (right): Distribution of APD when five out of six inputs are held at mean values, and the sixth input is varied across its range.

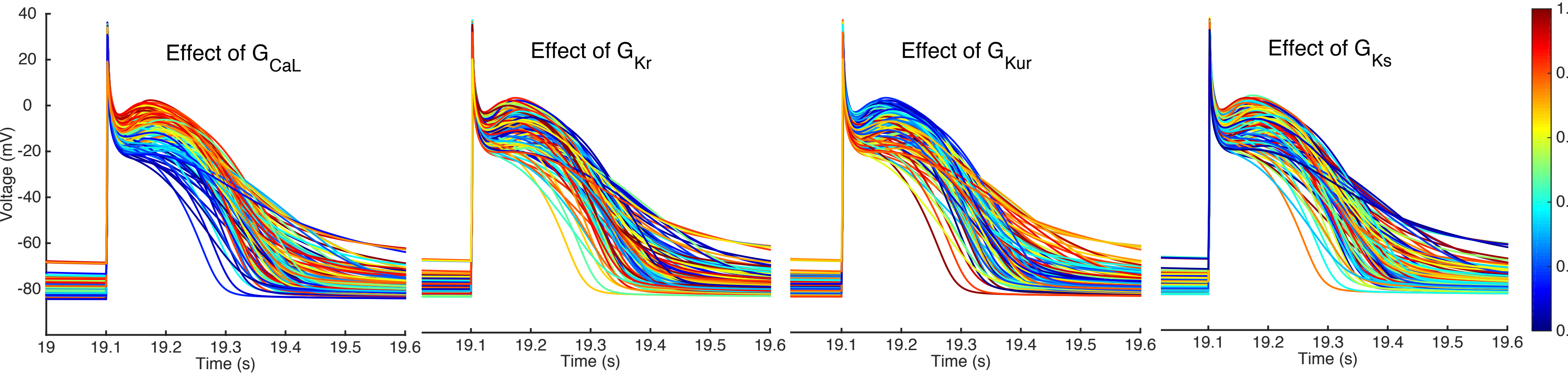


LR1 Sensitivity Analysis



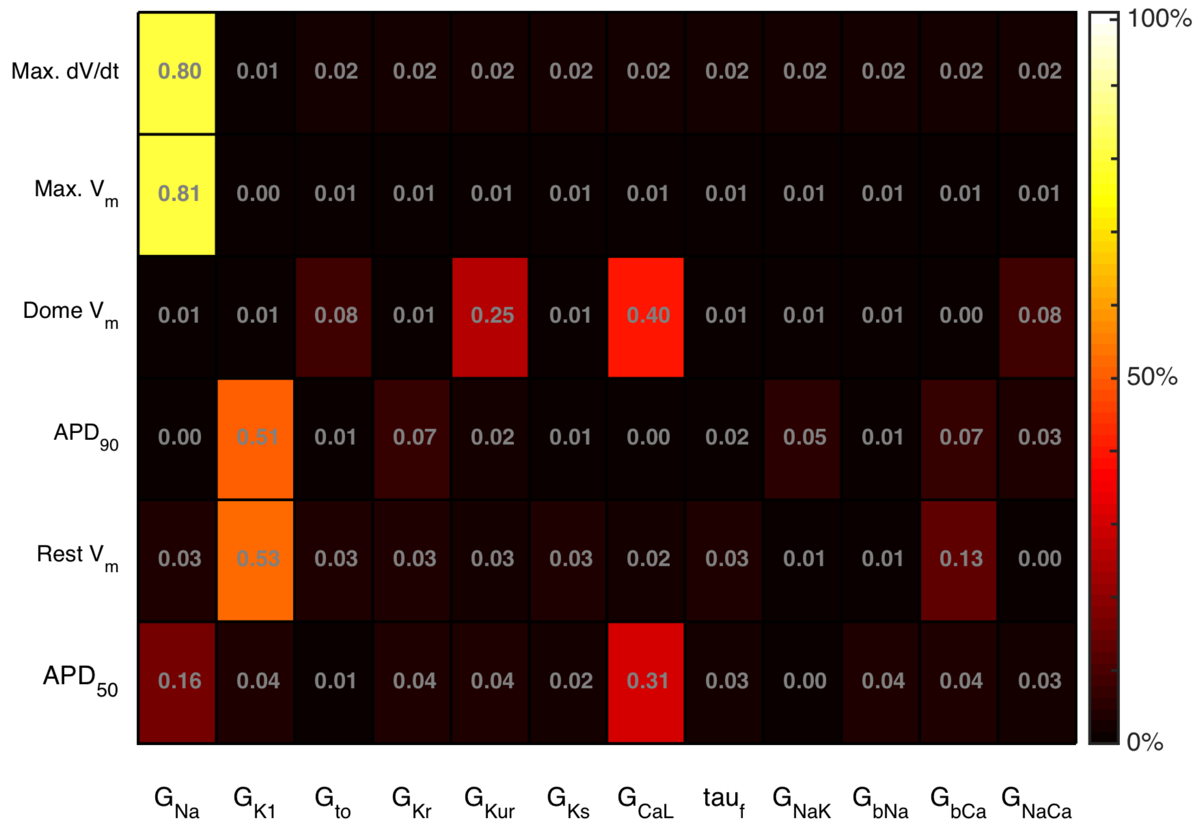
- Most outputs show strong sensitivity to one or two inputs
- APDR slope and min DI show weak sensitivity to inputs
- Sum of sensitivity indices for APDR slope and Min DI < 1

CRN: Thirteen inputs, six outputs



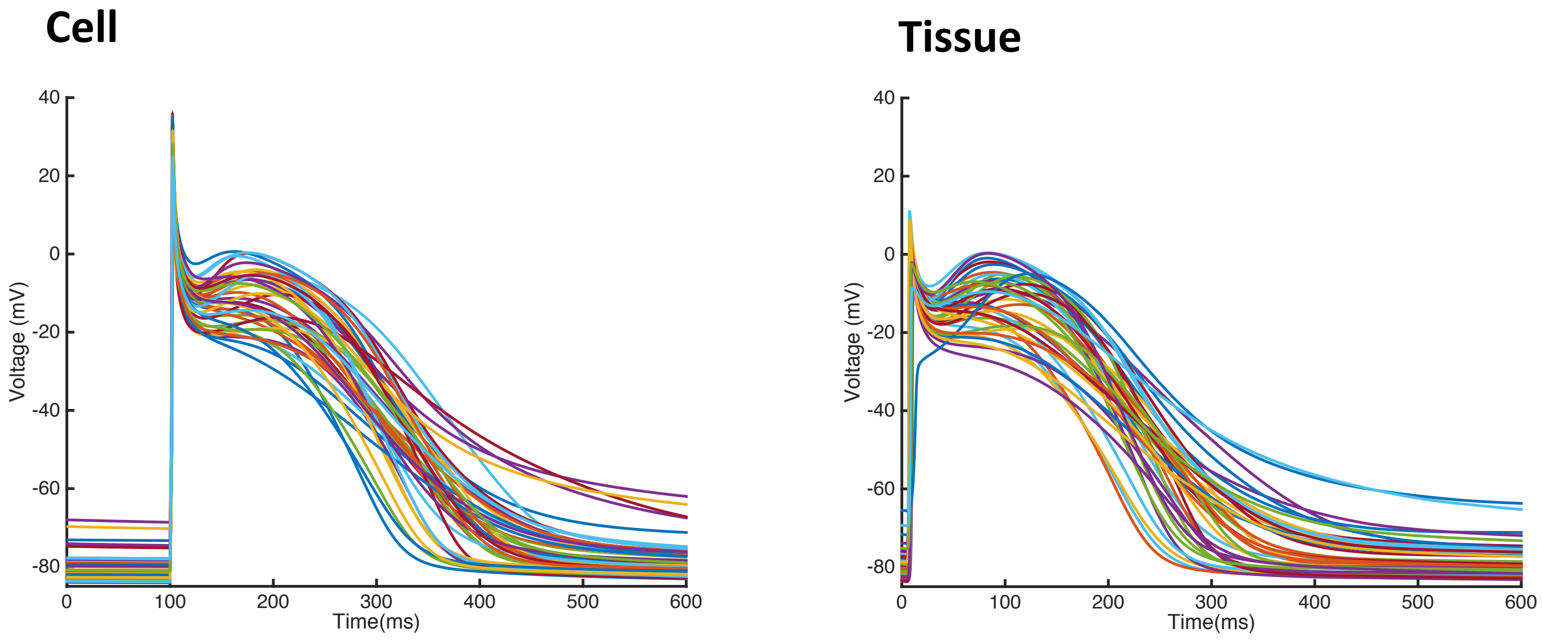
- In the CRN model we would expect increasing G_{CaL} to prolong APD, and increasing G_K to shorten APD
- The figure shows 100 design data action potentials coloured by G_{CaL} (left) and G_K (right 3), showing that the effects are harder to dissect than in the LR1991 model.

CRN: Sensitivity Analysis



- Sensitivity analysis for emulators trained with 150 design data samples
- GK1 has the largest effect on APD90, while GCaL influences dome voltage and hence APD50

Single Cell vs Tissue simulations



- Build emulators of cell and tissue models, so that the effect of uncertainty in cell scale parameters on tissue scale behaviour can be examined
- Simulations for a 2d tissue strip using CardiacEPSolver in Nektar++
- Preliminary results with the CRN models show that action potentials in cells and tissues are substantially different when parameters vary

Conclusions and next steps

- Emulators provide a promising framework for UQ in models of AF
- It is possible to examine contribution of variances on inputs to overall variability of the output, i.e. variance based sensitivity analysis.
- Very simple analysis, other GP variants may be better suited to this problem (Multivariate emulators, dynamic emulators)
- Need to investigate tissue parameters as well as ODE parameters
- Need to generate training data of tissue simulations for GP emulator
- Future extension for direct UQ using Nektar++
- Other forms of UQ for PDEs (e.g. Polynomial Chaos)?

- <SOLVERINFO>
- <I PROPERTY="EQTYPE" VALUE="Monodomain" />
- <I PROPERTY="CellModel” VALUE="CourtemancheRamirezNattel98" />
- <I PROPERTY="CellModelVariant" VALUE="Var" /> <!-- Set to 'Original', 'AF' or 'Var'. If 'Var' then need to set parameters-->
- <I PROPERTY="Projection" VALUE="Continuous"/>
- <I PROPERTY="DiffusionAdvancement" VALUE="Implicit"/>
- <I PROPERTY="TimeIntegrationMethod" VALUE="IMEXOrder1"/>
- <I PROPERTY="GlobalSysSoln" VALUE="IterativeStaticCond" />
- </SOLVERINFO>