Bayesian methods for the control-based continuation of multiple-timescale systems

Mark Blyth

Plan de jour

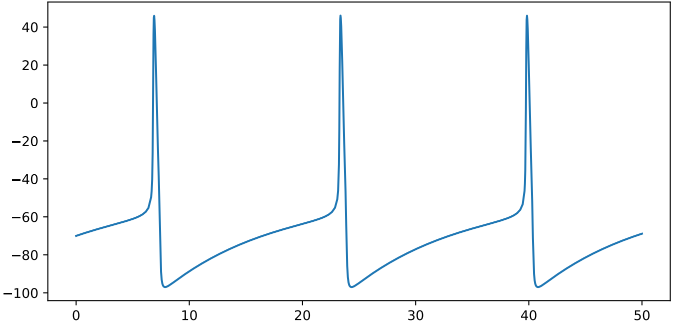
- CBC maths
- Surrogate modelling
- Novel discretisations

Plan de jour

- CBC maths
- Surrogate modelling
- Novel discretisations

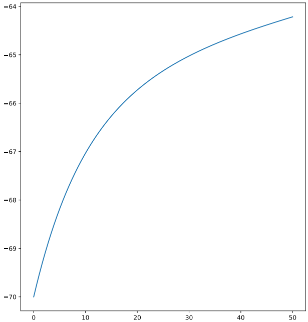
What is CBC?

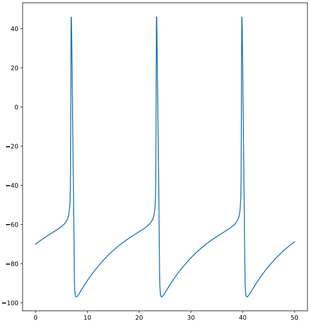
Dynamics are ‘what something does’



What is CBC?

A bifurcation is a change in dynamics





What is CBC?

Bifurcation analysis:

1. Find a feature

What is CBC?

Bifurcation analysis:

1. Find a feature
2. Change a parameter slightly

What is CBC?

Bifurcation analysis:

1. Find a feature
2. Change a parameter slightly
3. Find where the feature moved to

What is CBC?

Bifurcation analysis:

1. Find a feature
2. Change a parameter slightly
3. Find where the feature moved to
4. Bifurcations occur when features change, appear, or disappear

What is CBC?

- Numerical continuation:
  - Features x defined given by $f(x, \lambda) = 0$
  - Change λ, see how x changes

George Box

All models are wrong, but some are useful

What is CBC?

Control-based continuation; model-free bifurcation analysis:

1. Build a system controller

What is CBC?

Control-based continuation; model-free bifurcation analysis:

1. Build a system controller
   - Put in target $u^*(t)$

What is CBC?

Control-based continuation; model-free bifurcation analysis:

1. Build a system controller
   - Put in target $u^*(t)$
   - Controller makes system follow $u^*(t)$

What is CBC?

Control-based continuation; model-free bifurcation analysis:

1. Build a system controller
   - Put in target $u^*(t)$
   - Controller makes system follow $u^*(t)$
2. Find noninvasive $u^*(t)$

What is CBC?

Control-based continuation; model-free bifurcation analysis:

1. Build a system controller
   - Put in target $u^*(t)$
   - Controller makes system follow $u^*(t)$
2. Find noninvasive $u^*(t)$
   - Noninvasiveness := no control action applied

What is CBC?

Control-based continuation; model-free bifurcation analysis:

1. Build a system controller
   - Put in target $u^*(t)$
   - Controller makes system follow $u^*(t)$
2. Find noninvasive $u^*(t)$
   - Noninvasiveness := no control action applied
   - No control action = system behaves naturally

What is CBC?

Control-based continuation; model-free bifurcation analysis:

1. Build a system controller
   - Put in target $u^*(t)$
   - Controller makes system follow $u^*(t)$
2. Find noninvasive $u^*(t)$
   - Noninvasiveness := no control action applied
   - No control action = system behaves naturally
3. Change a parameter

What is CBC?

Control-based continuation; model-free bifurcation analysis:

1. Build a system controller
   - Put in target $u^*(t)$
   - Controller makes system follow $u^*(t)$
2. Find noninvasive $u^*(t)$
   - Noninvasiveness := no control action applied
   - No control action = system behaves naturally
3. Change a parameter
4. Find how noninvasive $u^*(t)$ changed

What is CBC?

Control-based continuation; model-free bifurcation analysis:

1. Build a system controller
   - Put in target $u^*(t)$
   - Controller makes system follow $u^*(t)$
2. Find noninvasive $u^*(t)$
   - Noninvasiveness := no control action applied
   - No control action = system behaves naturally
3. Change a parameter
4. Find how noninvasive $u^*(t)$ changed
   - Tracks system features, bifurcations without ever needing a model

CBC

Control-based continuation

A model-free bifurcation analysis method. Uses a controller to stabilise a system, and continuation to track features.

My project: use CBC to analyse the bifurcations that make neurons fire

What is discretisation?

Recent work: improving CBC discretisation

- Periodic orbits are functions satisfying $f(t) = f(t + T)$

What is discretisation?

Recent work: improving CBC discretisation

- Periodic orbits are functions satisfying $f(t) = f(t+T)$
- Tracking these means solving the functional equation $I\left[u^*\right] = \int_0^T \left[u(u^*, t)\right]^2 \mathrm{d}t = 0$ for function $u^*(t)$

What is discretisation?

Recent work: improving CBC discretisation

- Periodic orbits are functions satisfying $f(t) = f(t+T)$
- Tracking these means solving the functional equation $I\left[u^*\right] = \int_0^T \left[u(u^*, t)\right]^2 \mathrm{d}t = 0$ for function $u^*(t)$
  - Basically, trying to solve for a function

What is discretisation?

Recent work: improving CBC discretisation

- Periodic orbits are functions satisfying $f(t) = f(t+T)$
- Tracking these means solving the functional equation $I\left[u^*\right] = \int_0^T \left[u(u^*, t)\right]^2 \mathrm{d}t = 0$ for function $u^*(t)$
  - Basically, trying to solve for a function
- Discretisation lets us approximately solve the problem by solving a finite set of equations

What is discretisation?

Goal: solve $I\left[u^{*}\right]=0$

1. Translate problem to system of vector-valued equations
2. Solve system numerically
3. Translate solution back to a continuous function

Translation between continuous and vector-valued systems is discretisation

What is discretisation?

Definition (Discretisation)

The act of representing a continuous signal by a discrete counterpart

We want a discretisation that

- Has minimal discretisation error
- Is low-dimensional

How do we discretise?

- Let $\mathbf{u}^*$ be some vector ‘representing’ the signal $u^*(t)$

How do we discretise?

- Let $\mathbf{u}^*$ be some vector 'representing' the signal $u^*(t)$
  - Eg. Fourier: let our periodic target be $u^*(t) = a_0 + \sum a_i \cos i\omega t + \sum b_i \sin i\omega t$

How do we discretise?

- Let $\mathbf{u}^*$ be some vector 'representing' the signal $u^*(t)$
  - Eg. Fourier: let our periodic target be $u^*(t) = a_0 + \sum a_i \cos i\omega t + \sum b_i \sin i\omega t$
- We can represent the signal by its Fourier harmonics $\mathbf{u}^* = \{a_0, a_i, b_i\}$

How do we discretise?

- Let $\mathbf{u}^*$ be some vector ‘representing’ the signal $u^*(t)$
  - Eg. Fourier: let our periodic target be $u^*(t) = a_0 + \sum a_i \cos i\omega t + \sum b_i \sin i\omega t$
- We can represent the signal by its Fourier harmonics $\mathbf{u}^* = \{a_0, a_i, b_i\}$
- $u^*(t)$ can be represented by $\mathbf{u}^*$ with minimal error

How do we discretise?

- Let $\mathbf{u}^*$ be some vector 'representing' the signal $u^*(t)$
  - Eg. Fourier: let our periodic target be $u^*(t) = a_0 + \sum a_i \cos i\omega t + \sum b_i \sin i\omega t$
- We can represent the signal by its Fourier harmonics $\mathbf{u}^* = \{a_0, a_i, b_i\}$
- $u^*(t)$ can be represented by $\mathbf{u}^*$ with minimal error
- The functional problem can be rewritten as $I\left(\mathbf{u}^*\right) = 0$

How do we discretise?

- Let $\mathbf{u}^*$ be some vector 'representing' the signal $u^*(t)$
  - Eg. Fourier: let our periodic target be $u^*(t) = a_0 + \sum a_i \cos i\omega t + \sum b_i \sin i\omega t$
- We can represent the signal by its Fourier harmonics $\mathbf{u}^* = \{a_0, a_i, b_i\}$
- $u^*(t)$ can be represented by $\mathbf{u}^*$ with minimal error
- The functional problem can be rewritten as $I\left(\mathbf{u}^*\right) = 0$
  - Finite-vector equation, solvable!

How do we discretise?

- Let $\mathbf{u}^*$ be some vector 'representing' the signal $u^*(t)$
  - Eg. Fourier: let our periodic target be $u^*(t) = a_0 + \sum a_i \cos i\omega t + \sum b_i \sin i\omega t$
- We can represent the signal by its Fourier harmonics $\mathbf{u}^* = \{a_0, a_i, b_i\}$
- $u^*(t)$ can be represented by $\mathbf{u}^*$ with minimal error
- The functional problem can be rewritten as $I\left(\mathbf{u}^*\right) = 0$
  - Finite-vector equation, solvable!
- This is how we track dynamical features

Issues with discretisation

- Solving the discretised system takes a long time when it is high-dimensional
- Neuron signals require lots of Fourier harmonics to discretise
- Higher-order harmonics are harder to get *[Nyquist cap]* and less accurate *[SNR]*

Plan de jour

- CBC maths
- Surrogate modelling
- Novel discretisations

The need for surrogates

- Recent work: local surrogate models for experimental data

The need for surrogates

- Recent work: local surrogate models for experimental data

Definition (Surrogate models)

A local model for data, that can be used in place of experimental recordings

The need for surrogates

- Recent work: local surrogate models for experimental data

Definition (Surrogate models)

A local model for data, that can be used in place of experimental recordings

- Record experimental data

The need for surrogates

- Recent work: local surrogate models for experimental data

Definition (Surrogate models)

A local model for data, that can be used in place of experimental recordings

- Record experimental data
- Fit a surrogate model

The need for surrogates

- Recent work: local surrogate models for experimental data

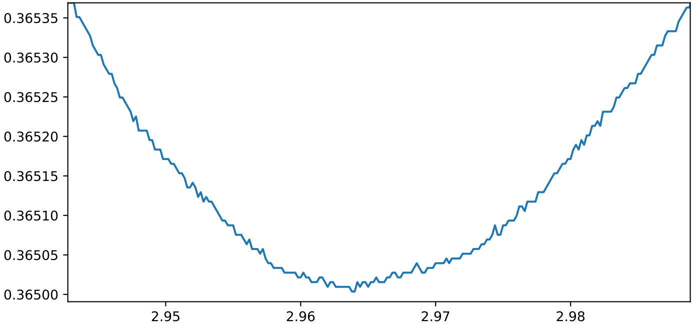
Definition (Surrogate models)

A local model for data, that can be used in place of experimental recordings

- Record experimental data
- Fit a surrogate model
- Perform analysis on model

Why surrogates?

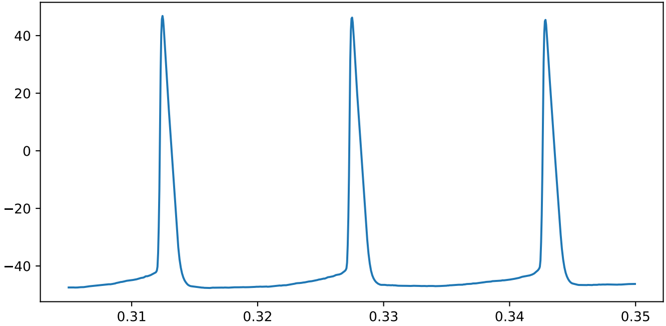
Real data are noisy



[Thanks to LR for the data]

Why surrogates?

Real data are ‘fast’



[Thanks to KTA for the data]

Why surrogates?

- We want to get rid of noise to get the best possible discretisation

Why surrogates?

- We want to get rid of noise to get the best possible discretisation
  - Fourier should encode only signal, not signal + noise

Why surrogates?

- We want to get rid of noise to get the best possible discretisation
  - Fourier should encode only signal, not signal + noise
- Fast signals mean lots of high-frequency energy

Why surrogates?

- We want to get rid of noise to get the best possible discretisation
  - Fourier should encode only signal, not signal + noise
- Fast signals mean lots of high-frequency energy
  - High signal-to-noise ratio on the harmonics that give sharp spikes

Why surrogates?

- We want to get rid of noise to get the best possible discretisation
  - Fourier should encode only signal, not signal + noise
- Fast signals mean lots of high-frequency energy
  - High signal-to-noise ratio on the harmonics that give sharp spikes
  - Simple low-pass filters would remove both noise *and* signal

Why surrogates?

- We want to get rid of noise to get the best possible discretisation
  - Fourier should encode only signal, not signal + noise
- Fast signals mean lots of high-frequency energy
  - High signal-to-noise ratio on the harmonics that give sharp spikes
  - Simple low-pass filters would remove both noise *and* signal
- A good surrogate lets us remove noise in a statistically optimal way

Why surrogates?

- We want to get rid of noise to get the best possible discretisation
  - Fourier should encode only signal, not signal + noise
- Fast signals mean lots of high-frequency energy
  - High signal-to-noise ratio on the harmonics that give sharp spikes
  - Simple low-pass filters would remove both noise *and* signal
- A good surrogate lets us remove noise in a statistically optimal way
  - Less noise = better discretisation

Bayesian surrogates

- We have a 'true' signal $f(t)$, but we can only see noise-corrupted samples $y_i = f(t_i) + \varepsilon$

Bayesian surrogates

- We have a 'true' signal $f(t)$, but we can only see noise-corrupted samples $y_i = f(t_i) + \varepsilon$
  - $f(t)$ is unknown, but we can reason about it with Bayes

Bayesian surrogates

- We have a ‘true’ signal $f(t)$, but we can only see noise-corrupted samples $y_i = f(t_i) + \varepsilon$
  - $f(t)$ is unknown, but we can reason about it with Bayes
  - Assume $\varepsilon \sim \mathcal{N}(0, \sigma^2)$

Bayesian surrogates

- We have a 'true' signal $f(t)$, but we can only see noise-corrupted samples $y_i = f(t_i) + \varepsilon$
  - $f(t)$ is unknown, but we can reason about it with Bayes
  - Assume $\varepsilon \sim \mathcal{N}(0, \sigma^2)$
- Let's estimate $y^* = f(t^*)$ at unseen data t^*

Bayesian surrogates

- We have a ‘true’ signal $f(t)$, but we can only see noise-corrupted samples $y_i = f(t_i) + \varepsilon$
  - $f(t)$ is unknown, but we can reason about it with Bayes
  - Assume $\varepsilon \sim \mathcal{N}(0, \sigma^2)$
- Let’s estimate $y^* = f(t^*)$ at unseen data t^*
  - Joint distribution: $p(f(t^*), t^*, y, t) \sim \mathcal{N}(\mu, \Sigma_k^2)$

Bayesian surrogates

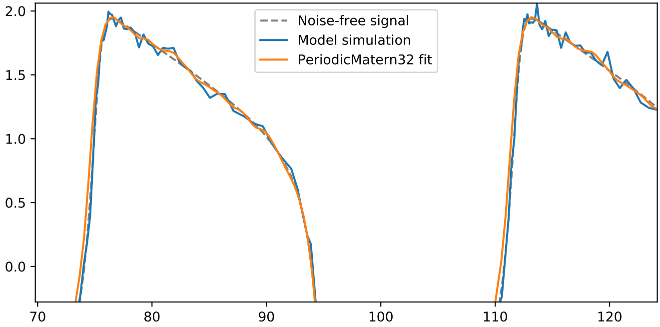
- We have a 'true' signal $f(t)$, but we can only see noise-corrupted samples $y_i = f(t_i) + \varepsilon$
  - $f(t)$ is unknown, but we can reason about it with Bayes
  - Assume $\varepsilon \sim \mathcal{N}(0, \sigma^2)$
- Let's estimate $y^* = f(t^*)$ at unseen data t^*
  - Joint distribution: $p(f(t^*), t^*, y, t) \sim \mathcal{N}(\mu, \Sigma_k^2)$
  - Conditional distribution: $p(f(t^*)|t^*, y, t)$

Bayesian surrogates

- We have a 'true' signal $f(t)$, but we can only see noise-corrupted samples $y_i = f(t_i) + \varepsilon$
  - $f(t)$ is unknown, but we can reason about it with Bayes
  - Assume $\varepsilon \sim \mathcal{N}(0, \sigma^2)$
- Let's estimate $y^* = f(t^*)$ at unseen data t^*
  - Joint distribution: $p(f(t^*), t^*, y, t) \sim \mathcal{N}(\mu, \Sigma_k^2)$
  - Conditional distribution: $p(f(t^*)|t^*, y, t)$
- This is Gaussian process regression!

Gaussian process regression surrogates

Build a statistically optimal regression model from noisy observations



GPR results

- GPR is Bayesian

GPR results

- GPR is Bayesian
  - Covariance function specifies our initial belief about the data

GPR results

- GPR is Bayesian
  - Covariance function specifies our initial belief about the data
- Covariance functions generally assume stationarity

GPR results

- GPR is Bayesian
  - Covariance function specifies our initial belief about the data
- Covariance functions generally assume stationarity
  - Assume smooth, nice signals

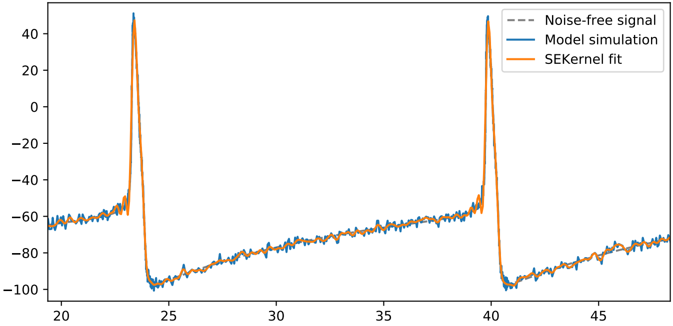
GPR results

- GPR is Bayesian
  - Covariance function specifies our initial belief about the data
- Covariance functions generally assume stationarity
  - Assume smooth, nice signals
- Stationary covariance = poorly encoded beliefs = low belief in posterior

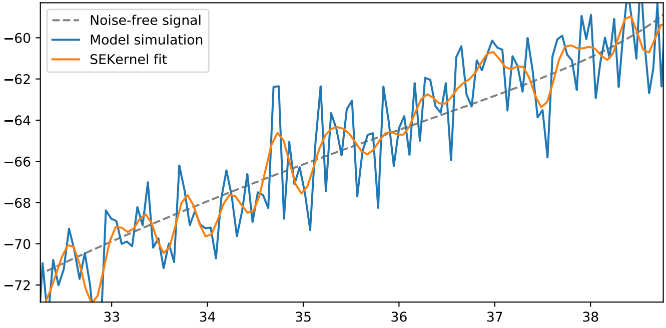
GPR results

- GPR is Bayesian
  - Covariance function specifies our initial belief about the data
- Covariance functions generally assume stationarity
  - Assume smooth, nice signals
- Stationary covariance = poorly encoded beliefs = low belief in posterior
  - Bayes with bad priors = bad results!

GPR results


Noise-free signal
Model simulation
SEKernel fit
40
20
0
−20
−40
−60
−80
−100
20
25
30
35
40
45

GPR results


Noise-free signal
Model simulation
SEKernel fit
−60
−62
−64
−66
−68
−70
−72
33
34
35
36
37
38

GPR results

- Stationary GPR, non-stationary data = overly flexible models
- Non-stationary would fix this
- Non-stationary GPR is hard!

Splines

- Less flexible alternative: splines

Splines

- Less flexible alternative: splines
- Choose some representative points

Splines

- Less flexible alternative: splines
- Choose some representative points
- Place a piece of cubic polynomial between each point

Splines

- Less flexible alternative: splines
- Choose some representative points
- Place a piece of cubic polynomial between each point
- Choose polynomials so that the function is smooth

Splines

- Less flexible alternative: splines
- Choose some representative points
- Place a piece of cubic polynomial between each point
- Choose polynomials so that the function is smooth
- Finite, low degree-of-freedom, forcibly averages out noise

Bayesian splines

- Choosing representative points is hard

Bayesian splines

- Choosing representative points is hard
- Alternative: don't!

Bayesian splines

- Choosing representative points is hard
- Alternative: don't!
  - Let ξ be a vector of representative points

Bayesian splines

- Choosing representative points is hard
- Alternative: don't!
  - Let ξ be a vector of representative points
  - Find $p(\xi|x, y)$

Bayesian splines

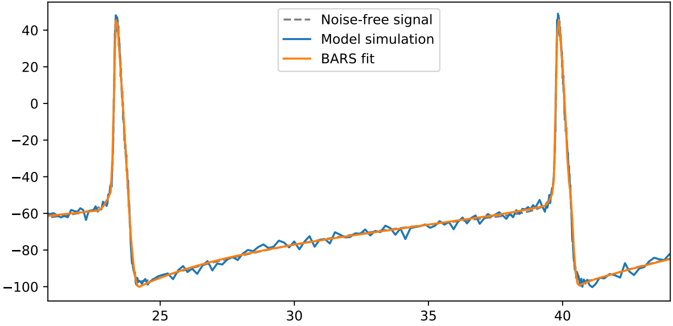
- Choosing representative points is hard
- Alternative: don't!
  - Let ξ be a vector of representative points
  - Find $p(\xi|x,y)$
  - Use that to estimate $p(f|\xi,x,y)$

Bayesian splines

- Choosing representative points is hard
- Alternative: don't!
  - Let ξ be a vector of representative points
  - Find $p(\xi|x, y)$
  - Use that to estimate $p(f|\xi, x, y)$
- This is Bayesian free-knot splines

Splines as a surrogate

Result 1: splines outperform stationary GPR as neuronal data surrogate



Plan de jour

- CBC maths
- Surrogate modelling
- Novel discretisations

The issue with surrogates

My current work. . .

- Bayesian free-knot splines gives a good noise-free surrogate model

The issue with surrogates

My current work…

- Bayesian free-knot splines gives a good noise-free surrogate model
  - More accurate discretisations

The issue with surrogates

My current work...

- Bayesian free-knot splines gives a good noise-free surrogate model
  - More accurate discretisations
- Issue: too many coefficients are needed to discretise the signal

The issue with surrogates

My current work...

- Bayesian free-knot splines gives a good noise-free surrogate model
  - More accurate discretisations
- Issue: too many coefficients are needed to discretise the signal
  - Too many = too slow

The issue with surrogates

My current work…

- Bayesian free-knot splines gives a good noise-free surrogate model
  - More accurate discretisations
- Issue: too many coefficients are needed to discretise the signal
  - Too many = too slow
- We can reconstruct signal from splines models

The issue with surrogates

My current work...

- Bayesian free-knot splines gives a good noise-free surrogate model
  - More accurate discretisations
- Issue: too many coefficients are needed to discretise the signal
  - Too many = too slow
- We can reconstruct signal from splines models
  - Is this a discretisation?

Splines as a discretisation

- Splines models are of form $\hat{f}(x) = \sum \beta_i b_i(x)$

Splines as a discretisation

- Splines models are of form $\hat{f}(x) = \sum \beta_i b_i(x)$
  - $b_i(x)$ form a set of basis functions over splines models

Splines as a discretisation

- Splines models are of form $\hat{f}(x) = \sum \beta_i b_i(x)$
  - $b_i(x)$ form a set of basis functions over splines models

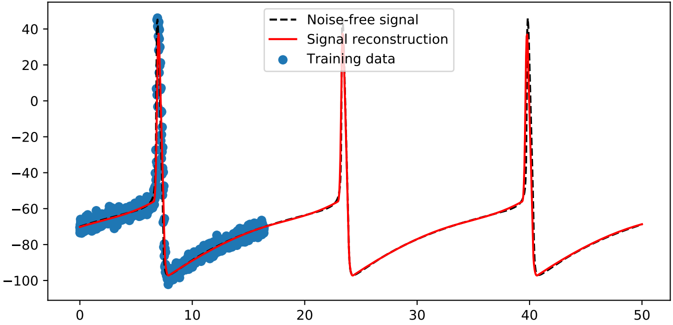
- For a basis set b_i, can the associated β_i discretise a signal?

Splines as a discretisation

- Splines models are of form $\hat{f}(x) = \sum \beta_i b_i(x)$
  - $b_i(x)$ form a set of basis functions over splines models
- For a basis set b_i, can the associated β_i discretise a signal?
  - Result 2: probably…

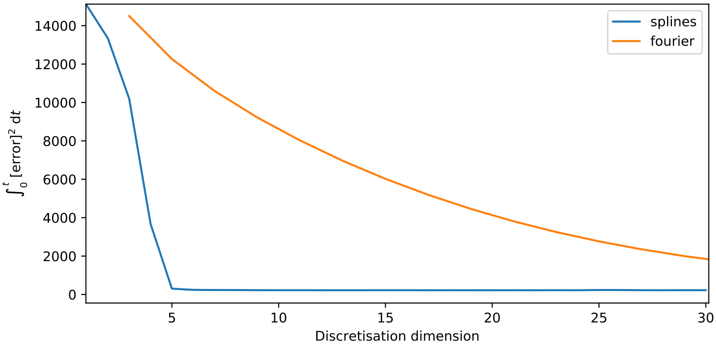
Spline discretisation

8-dimensional discretisation; but does it work with continuation?



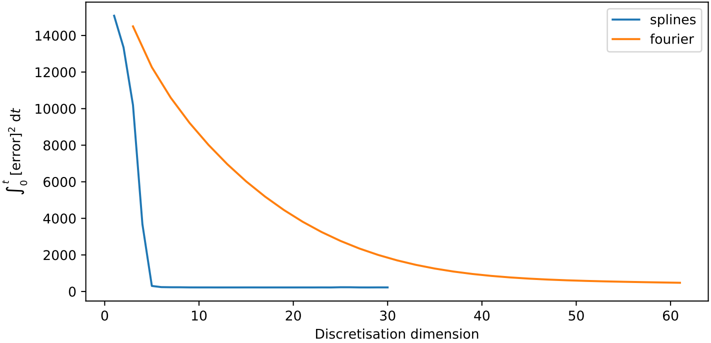
Splines vs Fourier

Hodgkin-Huxley neuron; error decays *significantly* faster with splines



Splines vs Fourier

Hodgkin-Huxley neuron; error decays *significantly* faster with splines



Where next?

- Test the robustness
- See if the discretisation breaks down with stochastic models
  - It probably will
- Test the discretisation with continuation
  - Splines discretisation is still only a local model
  - Need to ensure it can predict signals at other parameter values


That's all Folks