SAND2016-7668PE

Efficient Code Calibration with Gaussian Process Emulators

Nathan Bowman
Mentor: Matthew R. Denman

August 2, 2016

Outline

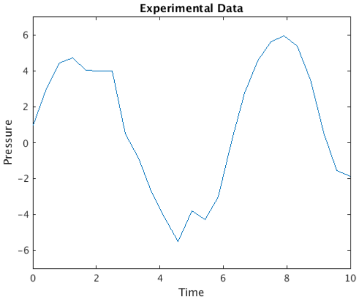
- Problem
- Approach
- Accomplishments
- Future Work
- About Me

Problem

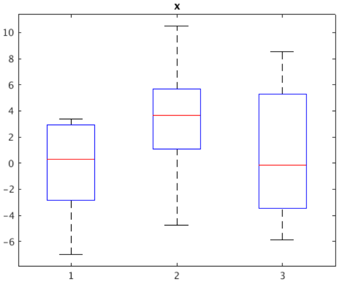
- Given long-running code (MELCOR, etc.) and experimental data, find input parameters $\widetilde{\boldsymbol{\theta}}$ to match experimental output
- Parameters need "error bars" to indicate confidence
- Given experimental output $\mathbf{y}$, want to determine posterior $P(\boldsymbol{\theta}|\mathbf{y})$
- Idea: choose prior $P(\boldsymbol{\theta})$ and use Markov-Chain Monte Carlo (MCMC) to determine posterior
- MCMC requires running simulator for each iteration
- Since thousands of MCMC iterations are required, this is often infeasible

Determining Simulation Inputs

$$simulator(x_1, x_2, x_3) =$$



$$\Longrightarrow$$



Approach

- Use an emulator in place of the simulation code
- Fast emulator will speed up inference by several orders of magnitude
- Gaussian Process Emulator:
  - Infinite-dimensional generalization of multivariate normal (MVN) distribution
  - Can choose different covariance functions to capture pattern of modeled function
  - Require solving linear system at $O(n^3)$ cost, where n is number of training inputs
- Use MCMC with emulator-supplied likelihood to draw samples from posterior distribution

Accomplishments

- Wrote `MATLAB` code for calibrating with Gaussian processes
- Wrote class for validating GP emulators against simulators to ensure validity of recovered parameters and help troubleshoot discrepancies
- Tested calibration on CONTAIN-LMR
  - Emulator matched simulator aside from known limitation
  - MCMC mixing was good
  - Small uncertainty in posterior parameter

CONTAIN-LMR Results

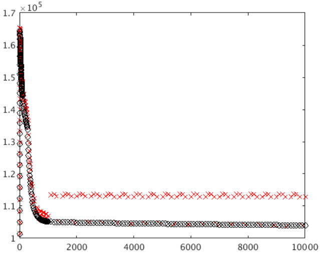


Figure: Emulator vs CONTAIN-LMR

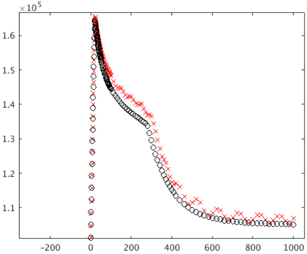


Figure: Emulator vs CONTAIN-LMR (zoomed)

Code output:

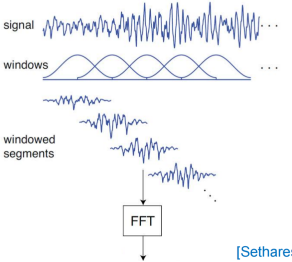
```
Mean of theta:  0.934167
Std of theta:  0.048463
Best value via optimization is:  0.947423
```

Future Work

- Function Factorization with Gaussian process Priors (FFGP)
  - Approximate matrix A by $A \approx uv^\top$
  - Higher rank approximations written $A \approx UV^\top = \sum_i u_i v_i^\top$, where u_i and v_i are columns of U and V
  - Gaussian processes used as priors for u_i, v_i
- Apply to more experiments
  - May require sparse approximations or FFGP for larger problems

About Me

- Current work involves signal processing and optimization
- Short-time Fourier Transform is essentially FFT on smaller chunks of a signal – results in time-frequency representation
- Trying to recover transformed signals with phase discarded
- Determining phase to maximize norm of recovered signal



[Sethares, 2007]