Bayesian calibration using UQLab

WINDTRUE: WP2

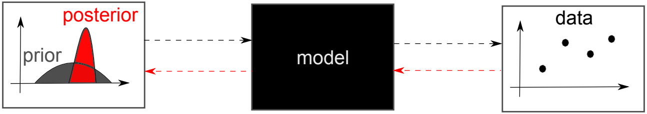
Vinit Dighe & Benjamin Sanderse

July 21, 2020

Motivation

Problem statement:

- For a given $\boxed{\text{computational model}}$, learn about the unknown (say θ) using data $\mathbf{y}$
- The **Bayesian model calibration** framework allows computation of the distribution of the unknown conditioned on data ($\pi(\theta|y)$)



Posterior computation is challenging!

Bayesian calibration framework

Given parameters $\theta \sim \pi(\theta)$ and measurement data $\mathbf{y}$, the Bayesian calibration (inverse) problem reads:

$$\pi(\theta|\mathbf{y}) = \frac{\pi(\mathbf{y}|\theta) \times \pi(\theta)}{Z} \quad \text{where} \quad Z = \int \pi(\mathbf{y}|\theta) \times \pi(\theta) d\theta$$

with:

- $\pi(\mathbf{y}|\theta)$: likelihood function (measure of how well the model fits the data)
- $\pi(\theta|\mathbf{y})$: posterior density function

Z is usually hard to determine!

Solution

MCMC sampling (needn't be normalized!) combined with simulation-based forward propagation to estimate: QoI $\mathbb{E}[\theta|\mathbf{y}]$

Computationally expensive!

Under the assumption that the measurement errors ϵ are iid and normally distributed, likelihood function:

$$\pi(\mathbf{y}|\theta) = \frac{1}{(2\pi\sigma^2)^{\frac{n}{2}}} e^{SS_\theta/2\sigma^2} \quad with \quad SS_\theta = \sum_{i=1}^{n} [\mathbf{y}_i - f_i(\theta)]^2$$

is replaced by a surrogate likelihood function:

$$\widetilde{\pi}(\mathbf{y}|\theta) = \frac{1}{(2\pi\sigma^2)^{\frac{n}{2}}} e^{\widetilde{SS}_\theta/2\sigma^2} \quad with \quad \widetilde{SS}_\theta = \sum_{i=1}^{n} [\mathbf{y}_i - \widetilde{f}_i(\theta)]^2$$

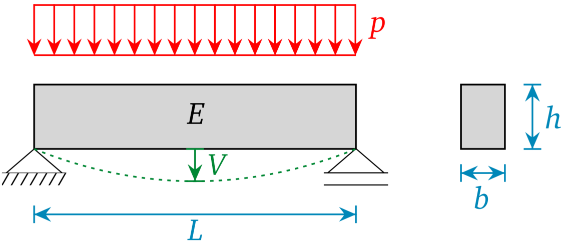
PCE using LARS

CWI

Outline

- Simple beam model calibration
- ECN AeroModule calibration
- Conclusions

Simple beam model calibration



Problem setup

Ingredients for Bayesian calibration

- Data ($\mathbf{y}$): {12.84, 13.12, 12.13, 12.19, 12.67} (mm)
- $\boxed{\text{Forward model}}$: $V = \frac{5}{32}\frac{pL^4}{Ebh^3}$
- Likelihood: $\pi(\mathbf{y}|\theta)$: $\sigma^2 = 10^{-6}$
- Prior distribution:

θ	$\pi(\theta)$
b (m)	0.15
h (m)	0.3
L (m)	5
p (kN/m)	$\mathcal{N}(0.012, 0.0006)$
E (MPa)	$\mathcal{LN}(30000, 4500)$

Prior and posterior distribution

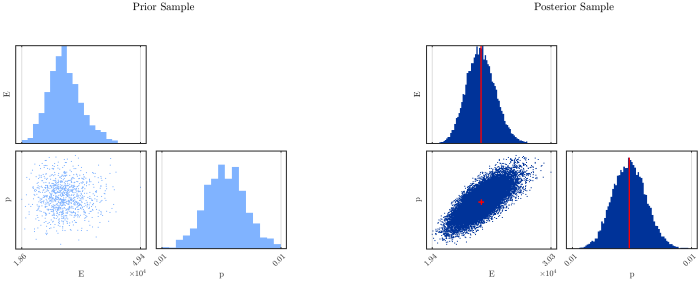


Figure: Prior and posterior samples.

MAP estimate

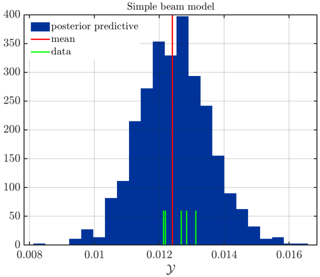
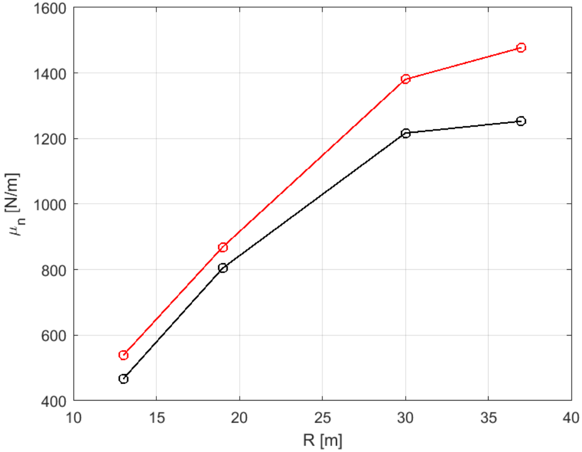


Figure: Bayesian estimate using posterior distribution against the experimental data.

MAP estimate: $\mathbb{E}[\theta|\mathbf{y}]$

p	E
2.4×10^4	0.0012

ECN AeroModule calibration



Problem setup

Ingredients for Bayesian calibration

- Axial force data ($\mathbf{y}$) $= \{\mathbf{y}_1, \mathbf{y}_2, \mathbf{y}_3, \mathbf{y}_4\}$
- $\boxed{\text{Forward model}}$: ECN Aero-Module
- Likelihood: $\pi(\mathbf{y}|\theta) = \prod_{i=1}^{n} \mathcal{N}[\mathbf{y}_i - f_i(\theta)]^2$
- Prior distribution:

θ	$\pi(\theta)$
P	Constant
C_L	Uniform
σ^2	Uniform

Problem setup

Name of Matlab file representing the turbine data for calibration

```
turbineName = 'NM80_calibrate'; % 'NM80', 'AVATAR'
% check |NM80_calibrate.m| or |(TurbineName_calibrate).m| for turbine-
specific
% settings and definition of uncertainties
```

Marco's (ECN) script for reading the data in N/m

```
filename_exp = ('../../../Experimental/WINDTRUE/raw.dat');
output_raw = read_exp_data(filename_exp, 2);
% Because the model has different discrepancy options at different
 radial locations,
% the measurement data is stored in four different data structures:
Data(1).y = mean(output_raw.Fy03); % [N/m]
Data(1).Name = 'Fy03';
Data(1).MOMap = 1; % Model Output Map 1
```

Name of Matlab file representing the model

```
Model.mHandle = @aero_module_calibration;
% Optionally, one can pass parameters to model stored in the cell
 array P
P = getParameterAeroModule(turbineName);
Model.Parameters = P;
Model.isVectorized = false;
```

```
DiscrepancyPriorOpts1.Name = 'Prior of sigma 1';
DiscrepancyPriorOpts1.Marginals(1).Name = 'Sigma1';
DiscrepancyPriorOpts1.Marginals(1).Type = 'Uniform';
DiscrepancyPriorOpts1.Marginals(1).Parameters =
 [0.5*std(output_raw.Fy03), 1.5*std(output_raw.Fy03)];
DiscrepancyPrior1 = uq_createInput(DiscrepancyPriorOpts1);

DiscrepancyOpts(1).Type = 'Gaussian';
DiscrepancyOpts(1).Prior = DiscrepancyPrior1;
```

Problem setup

Switch for Bayesian analysis with the AeroModule or with the surrogate model

```
Bayes_full = 0; % 0: use surrogate model (PCE); 1: run full model for
 Bayes (Computationally expensive!)

% If Bayes_full = 0, we need to specify options for loading a
 surrogate model
Surrogate_model_type = 0; % 0: Uses a stored PCE surrogate model, 1:
 create surrogate model

% Options for loading a surrogate model
Surrogate_model_filename = 'surrogate/PCE_60.mat'; % Specify the
 surrogate model file to be used

% Options for creating a surrogate model
% These are used if Bayes_full = 0 and Surrogate_model_type = 1
MetaOpts.Type = 'Metamodel';
MetaOpts.MetaType = 'PCE';
MetaOpts.Method = 'LARS'; % Quadrature, OLS, LARS

MetaOpts.ExpDesign.Sampling = 'LHS';
MetaOpts.ExpDesign.NSamples = 60;
MetaOpts.Degree = 1:4;
MetaOpts.TruncOptions.qNorm = 0.75;
```

MCMC parameters

```
Solver.Type = 'MCMC';
% MCMC algorithms available in UQLab
MH = 0; % Metropolis-Hastings
AM = 0; % Adaptive Metropolis
AIES = 1; % Affine invariant ensemble
HMC = 0; % Hamilton Monte Carlo
```

Convergence diagnostics

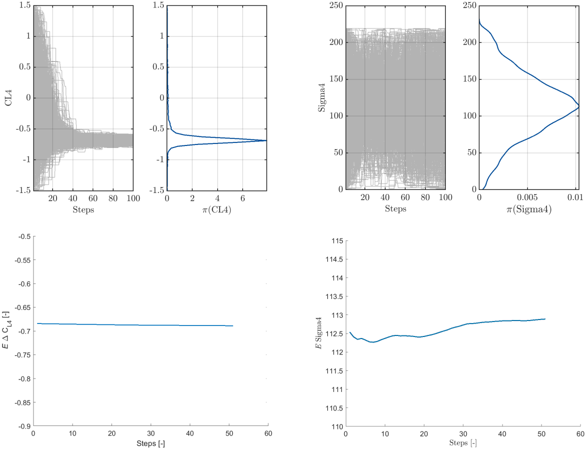


Figure: Trace plots and convergence for y4

Prior and posterior distribution

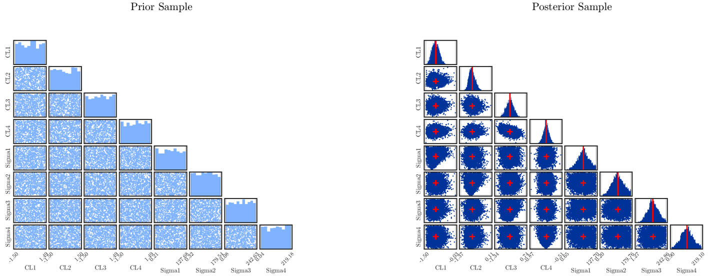


Figure: Prior and posterior samples.

MAP estimate

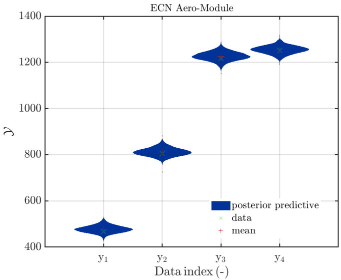


Figure: Violin plot showing distributions of Bayesian prediction against the DANAERO data.

MAP estimate: $\mathbb{E}[\theta|\mathbf{y}]$

ΔC_{L1}	ΔC_{L2}	ΔC_{L3}	ΔC_{L4}	σ_1^2	σ_2^2	σ_3^2	σ_4^2
-0.2025	-0.1604	-0.1087	-0.2067	73.7666	100.2012	138.4856	112.8861

Calibrated polars

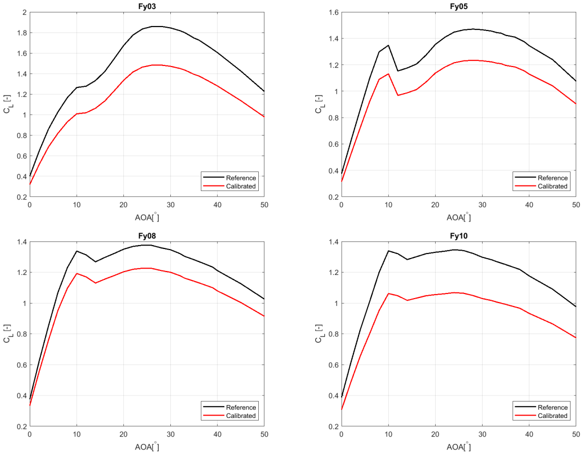


Figure: C_L polars comparison.

QoI validation

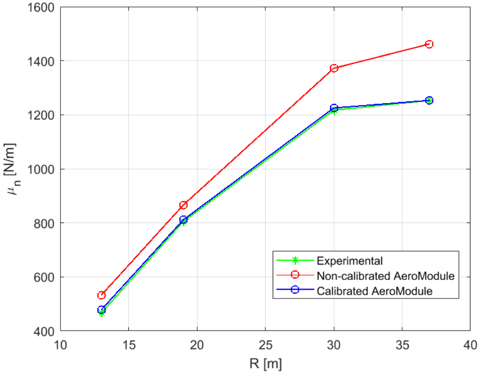


Figure: Comparison of axial force obtained using: experimental, non-calibrated Aero-Module run and calibrated Aero-Module run.

Conclusions

Conclusions

Bayesian model calibration with MCMC sampling and PCE suurogate model is successfully applied using the DANAERO experimental data.

Next steps

- Further validation
- Code testing
- Article preparation