# Uncertainty Quantification Training Seminar 2021
# Bayesian Model Updating: 1-D Linear Static System

Facilitators: Adolphus Lye, Edoardo Patelli

## 1 Problem Description

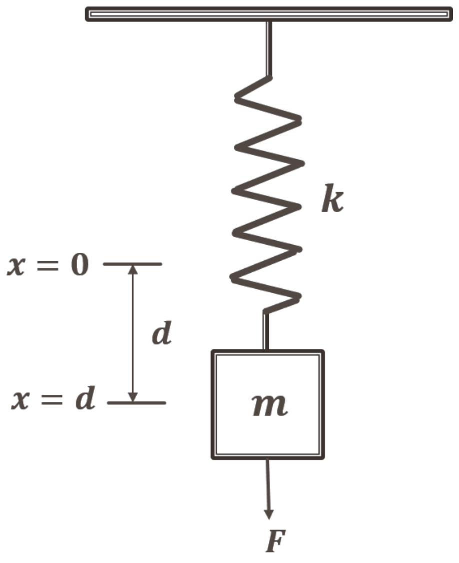

**Figure 1:** *Schematic set-up of the 1-dimensional linear static spring-mass system.*

.

In this problem we will be working on a simple 1-dimensional linear static spring-mass system as shown in Figure 1 whose spring has a constant stiffness, $k$. With a known displacement, $d$, we wish to estimate the value of $k$ through measuring its restoring force, $F$. Assuming the spring obeys the Hooke's Law relation, the quantity $F$ can be calculated via the following equation:

$$F = -k \cdot d \tag{1}$$

In essence, the model described by Eq. (1) is assumed to describe the system perfectly. However, in reality, the measurement of $F$ is usually corrupted by noise. Let the measurements of $F$ be $F_{noisy}$ and it is defined as follows:

$$F_{noisy} = -k \cdot d + \epsilon \tag{2}$$

Here, $\epsilon$ is a random noise variable which follows a Normal distribution with mean $0$ $N$ and standard deviation $1.0$ $N$. For this set-up, $d$ is set at $0.05$ $m$ while the true value of $k$ is $263$ $N/m$.

## 2 Assignment

The assignment consists of 5 main tasks. Each task involves the participants to write out segments of the code using the COSSAN objects that they have just learnt. The tasks should be completed sequentially and in order. Upon completion of all the required tasks, participants should have a working code which allows for the estimation of the epistemic parameter, $k$, as well as the updating/calibration of the Hooke's Law model through the Bayesian model updating technique.

### 2.1 Task 1: Setting up the model

a ) On a new MATLAB script, write and set-up the nominal model function handle with reference to Eq. (1).

b ) On a new MATLAB script, write and set-up the "noisy" model function handle with reference to Eq. (2).

### 2.2 Task 2: Generate synthetic data for $F_{noisy}$

You will need to do this on a new MATLAB script.

a ) Define the following fixed parameters and random variables:

- Define the true stiffness parameter, $k$, using the OpenCOSSAN 'Parameter' constructor.
- Define the displacement parameter, $d$, using the OpenCOSSAN 'Parameter' constructor.
- Define the noise random variable, $\epsilon$, using the OpenCOSSAN 'RandomVariable' constructor for a Normal distribution.
- Consolidate the random data using the OpenCOSSAN 'RandomVariableSet' constructor.

b ) Simulate $F_{noisy}$ data using Monte Carlo simulation:

- Consolidate the input parameters and random variables for the "noisy" model using the OpenCOSSAN 'Input' constructor.
- Call out the "noisy" model function handle using the OpenCOSSAN 'Mio' constructor.
- Set up the model evaluator by adding the 'Mio' constructor in the previous step to the OpenCOSSAN 'Evaluator' constructor.
- Define the "noisy" model using the OpenCOSSAN 'Model' constructor.
- Set up the Monte Carlo sampler using the OpenCOSSAN 'MonteCarlo' constructor. Set $Nsamples = 500$ (You may use a different sample size). Perform the simulation using the OpenCOSSAN 'Apply' constructor.

### 2.3 Task 3: Deterministic analysis set-up

Continue working on the current MATLAB script. The following sub-tasks below are similar to what you did in Task 2(b).

- Consolidate the input parameters for the nominal model using the OpenCOSSAN 'Input' constructor.
- Call out the nominal model function handle using the OpenCOSSAN 'Mio' constructor.
- Set up the model evaluator by adding the 'Mio' constructor in the previous step to the OpenCOSSAN 'Evaluator' constructor.
- Define the nominal model using the OpenCOSSAN 'Model' constructor.
- Execute the deterministic analysis using the OpenCOSSAN 'deterministicAnalysis' class.
- Obtain the result of the deterministic analysis using the OpenCOSSAN 'getValues' object. **Note:** As a check to see if you are on the right path, the results should simply yield $-13.15$ $N$ (i.e. $F_{deterministic} = -263 \times 0.05$).

### 2.4 Task 4: Bayesian analysis set-up

Continue working on the current MATLAB script.

For the Bayesian analysis, a non-informative Uniform prior is used whose lower and upper limits are 1 $N/m$ and 1000 $N/m$ respectively. The likelihood function used in this problem would be a Normal distribution with standard deviation, $\sigma = 1.0$ $N$. To generate samples from the posterior, the TMCMC sampling technique will be used for this problem.

a ) Define the Prior and the Likelihood functions:

- Define the prior random variable using the OpenCOSSAN 'RandomVariable' constructor for a Uniform distribution.
- Consolidate the random data from the prior using the OpenCOSSAN 'RandomVariable-Set' constructor.
- Consolidate the input prior random variable for the nominal model using the OpenCOSSAN 'Input' constructor.
- Call out the nominal model function handle using the OpenCOSSAN 'Mio' constructor.
- Set up the model evaluator by adding the 'Mio' constructor in the previous step to the OpenCOSSAN 'Evaluator' constructor.
- Define the nominal model using the OpenCOSSAN 'Model' constructor.
- Define the loglikelihood function using the OpenCossan 'LogLikelihood' constructor.

b ) Set up the TMCMC sampler:

- Define $Nsamples = 600$ (You may use a different sample size).

- Consolidate the prior and the loglikelihood functions using the OpenCOSSAN 'Bayesian-ModelUpdating' object.
- Run the TMCMC sampler using the OpenCOSSAN 'applyTMCMC' class.
- Obtain the resulting samples using the OpenCOSSAN 'getValues' object.

### 2.5 Task 5: Analysis of results

Continue working on the current MATLAB script.

- Generate a histogram figure to observe the histogram profile of the samples obtained via the TMCMC sampler.
- Use samples generated from the TMCMC sampler as input of the nominal model. From there, obtain the resulting calibrated model output.
- Generate the histograms of both $F_{noisy}$ samples and that of the calibrated model output in one figure.

**Note:** Ensure that all the MATLAB scripts (3 of them) are placed in the same directory. Once this is checked, RUN the script you've just worked on!

## 3 Discussions

- What is your estimation of $k$? How precise is your estimation?
- How would your precision of the estimate change if you vary the number of samples generated from the posterior using TMCMC sampler?
- How would your precision of the estimate change if you vary the standard deviation of the noise parameter, $\epsilon$?
- How would your precision of the estimate change if you vary the standard deviation of the Normal likelihood function, $\sigma$?
- Compare the histogram profile of $F_{noisy}$ and that obtained from the updated/calibrated model using the samples obtained from TMCMC sampler as input. What do you notice?
- What would be the result of $k$ if we change the prior distribution to something different? Let's say a Normal distribution. Give it a try!