Case 2: Probability of Overtopping using Hunt's Formula

This case involves the set-up of a simple probabilistic computation of the probability of overtopping of a dike (Figure 1). First, we will set-up the necessary elements in a probabilistic computation: the stochastic variables and the limit state function. Next, we will compute the probability of overtopping using both the FORM and Crude Monte Carlo method. Finally, we will visualize the results.

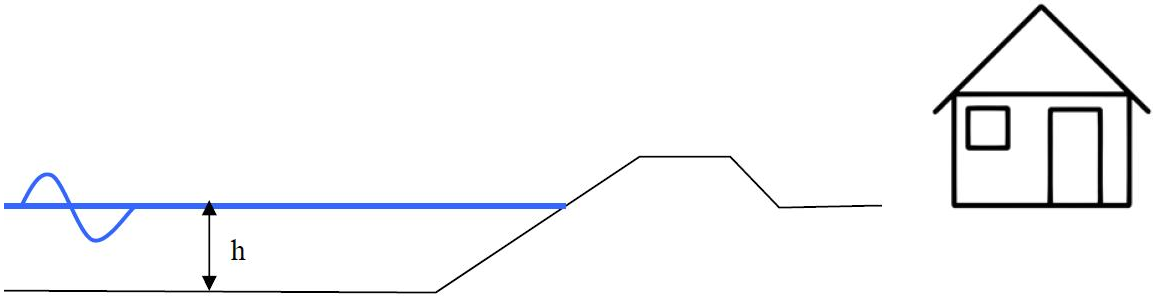

Figure 1 Situation sketch

We will use Hunt's formula to compute the run-up height above the mean water level:

$$\frac{R_u}{H_s} = \xi$$

$$\xi = \frac{\tan(\alpha)}{\sqrt{\frac{H_s}{L_0}}} = \frac{\tan(\alpha)}{\sqrt{\frac{2\pi H_s}{g T_p^2}}}$$

In this simple case, we consider overtopping to occur in case the mean water level plus the run-up height is larger than the dike crest:

$$MWL + R_u > h_{crest}$$

All heights and levels are with respect to a certain reference. In this example we use NAP.

Step-by-step description

1. Determine the (stochastic) variables in Hunt's formula and choose the following distributions for them:

Variable name	*Distribution*	*Par. #1*	*Par. #2*
MWL	Exponential	NAP+0.5m	1
H_s	Normal	3m	1m
T_p	Normal	6s	2s
h_{crest}		NAP+5m	
tan(α)		1/3	

2. Create an M-file (case2.m) to create the necessary stochastic variable structure. It has the following layout:

```
stochast =

1xN struct array with fields:
    Name
    Distr
    Params
```

With for example:

```
stochast(1) =

      Name: 'WL'
     Distr: @exp_inv
    Params: {[1]  [0]}
```

3. Create another M-file in the same folder with the limit state function description. It has the following layout:

```
function Z = x2z_HuntsFormula(varargin)

% load sample values
samples = struct(varargin{:});

...

% z value
Z = R - S;
```

The variable *samples* contains the samples of the stochastic variables you defined in the previous step. This is a similar construction as used in Case 1 with the *OPT* variable, but without default values. You can access the sample for the water level, for example, as follows:

```
samples.WL
```

Be aware that it is possible that this variable will be a vector containing several samples at once. Your code should be able to process these vectors item-by-item. Therefore, use dot-operators (.* and ./ etc.) in your formulations.

4. Add a call to the FORM routine specifying both the newly created stochastic variable structure and the z-function handle.

5. Do the same using the MC routine. Choose a sensible number of samples.

6. Execute your script and visualize your results using the *plotFORMresult* and *plotMCResult* functions.

Additional questions

If you succeeded in running the probabilistic computations above, you can ask yourself the following questions in order to explore the results. The results are stored in the *Output* field of the result variable.

1. Has the FORM computation converged? How many iterations were needed? And how many limit state function realizations?

2. How many limit state function realizations were needed in the Monte Carlo simulation? How many of these realizations are in the failure domain? Is that enough?

3. What is the difference in probability computed by FORM and Monte Carlo? Is this a large difference? What causes the difference?

4. What are the value *Beta* and *alpha* in the FORM result? What is the value *Beta* in the Monte Carlo results?

5. What parameter contributes most to the computed probability?

6. The *plotFORMresult* function visualizes the convergence of FORM in terms of beta. Can you visualize the convergence of the Monte Carlo computation in terms of probability?

   Hint: the probability of failure in a Monte Carlo simulation is defined by the number of failure points divided by the total number of samples (so far). The Matlab function *cumsum* computes the cumulative sum of an array. For example:

   ```
   cumsum([0 0 1 1 0 1 0 1 1 0]) = [0 0 1 2 2 3 3 4 5 5]
   ```

7. There are several vectors and matrices present in the result structures for FORM and Monte Carlo. What is the difference between vectors and matrices? Why does the length of some vectors (or first dimension of some matrices) differ in the FORM results from others?

8. Did you use deterministic variables in your stochastic variable structure? If not, where could you use those?