Case 3a: Adding a second requirement

Case 2 described the set-up of a simple probabilistic computation using Hunt's formula. Now it is time to add some real-world trouble to this case. As has been explained, FORM is a fast method, but cannot cope well with discontinuities in the failure domain. A possible cause of these discontinuities is the existence of multiple requirements.

We use the probabilistic computation from Case 2 as starting point. In Case 2 we computed the probability of overtopping of a dike (Figure 1). The dike protects agricultural land. This land has a need of water that can only be met in case the water level is at least about NAP+1m. In this case we will compute the probability that both the overtopping requirement form Case 2 and the requirement for a minimum water level are met.

First, we will alter the necessary elements in a probabilistic computation: the stochastic variables and the limit state function. Next, we will compute the probability of failure using both the FORM and Crude Monte Carlo method. Finally, we will visualize the results.

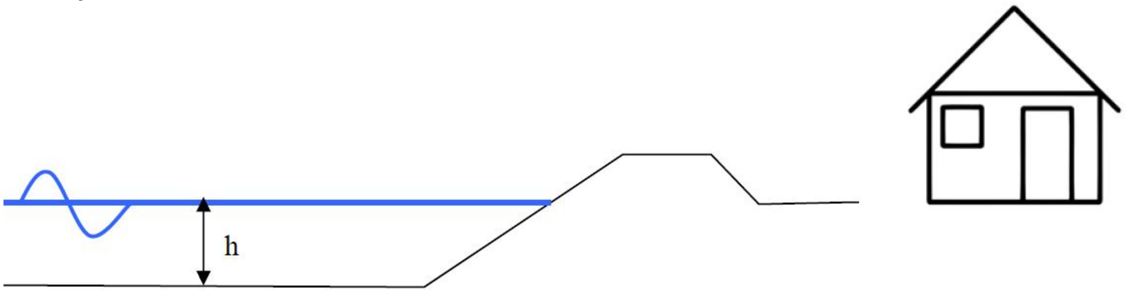

Figure 1 Situation sketch

In this simple case, we consider the need for water to be met in case the mean water level is larger than a certain critical value; so the unfavorable situation is defined as:

$$MWL < h_{critical}$$

Step-by-step description

1. Add an additional (stochastic) variable that can be defined by the following distribution:

Variable name	*Distribution*	*Par. #1*	*Par. #2*
$h_{critical}$	Normal	NAP+1m	0.1

2. Copy the M-file from Case 2 and alter the code to create the necessary stochastic variable structure.

3. Copy the M-file with the limit state function description from Case 2 and alter the code to implement the additional requirement.

   Be aware that it is possible that the input variables will be vectors 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. Make sure that the limit state function description should be smaller than zero in case *one or both* of the requirements are not met.

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

3. Did the computations seriously change with respect to Case 2 in terms of: computational effort, reliability, failure probability?

4. The *plotFORMresult* function visualizes the convergence of FORM in terms of beta. Can you visualize the convergence of both the FORM and Monte Carlo computation in terms of probability? What result is more reliable?