Towards Efficient Ways of Estimating Failure Probability of Mechanical Structures Under Interval Uncertainty

Michael Beer\textsuperscript{1}, Marco De Angelis\textsuperscript{1}, and Vladik Kreinovich\textsuperscript{2}

\textsuperscript{1}Institute for Risk and Uncertainty, School of Engineering University of Liverpool, UK
mbeer@liverpool.ac.uk, Marco.De-Angelis@liverpool.ac.uk
\textsuperscript{2}Department of Computer Science, University of Texas at El Paso
El Paso, TX 79968, USA, vladik@utep.edu
1. Case of Full Knowledge: Reminder

- The textbook approach to estimating the failure probability of mechanical structures assumes that:
  - we know the pdf $f(\theta)$ of quantities $\theta = (\theta_1, \ldots, \theta_n)$ describing the structure and its environment;
  - we know a limit function $g(\theta)$ such that:
    * stable states correspond to $g(\theta) > 0$, while
    * failures correspond to $g(\theta) < 0$.

- Once we know this information, we can find the desired failure probability $P$ as the integral $P = \int_{g(\theta)<0} f(\theta) \, d\theta$.

- $P$ can be computed by a (somewhat time-consuming) Monte-Carlo algorithm (MCA).
2. Case of Interval Uncertainty: A Description

- We often do not know the exact probabilities:
  - usually, we know that the distribution belongs to a certain family (e.g., that it is normal),
  - but we only know the bounds $\underline{p}_i \leq p_i \leq \overline{p}_i$ on the corresponding parameters $p_i$.

- Similarly, we know the general parametric expression for the limit function (e.g., that $g(\theta)$ is linear or quadratic).

- However, we only know the bounds on the actual (unknown) values of the corresponding parameters $p_j$.

- For each combination of parameters $p_i$, we can compute the failure probability $P_f(p_1, \ldots, p_m)$.

- We want to compute the range

$$[\underline{P}, \overline{P}] \overset{\text{def}}{=} \left\{ P_f(p_1, \ldots, p_n) : p_i \in [\underline{p}_i, \overline{p}_i] \right\}.$$
3. Linearization is Usually Possible

- Each interval can be represented as $[\tilde{p}_i - \Delta_i, \tilde{p}_i - \Delta_i]$, where $\tilde{p}_i$ is a midpoint and $\Delta_i$ is half-width.

- Possible values $p_i$ are $p_i = \tilde{p}_i + \Delta p_i$, with $|\Delta p_i| \leq \Delta_i$, so $P_f(p_1, \ldots, p_m) = P_f(\tilde{p}_1 + \Delta p_1, \ldots, \tilde{p}_m + \Delta p_m)$.

- The values $\Delta_i$ are usually reasonable small, hence the values $\Delta p_i$ are also small; thus:
  - we can expand $P_f$ into Taylor series
  - and keep only linear terms in this expansion:

$$P_f(\tilde{p}_1 + \Delta p_1, \ldots) = \tilde{P} + \sum_{i=1}^{m} c_i \cdot \Delta p_i, \quad \tilde{P} \overset{\text{def}}{=} P_f(\tilde{p}_1, \ldots), \quad c_i \overset{\text{def}}{=} \frac{\partial P}{\partial p_i}.$$

- Here, $\max(c_i \cdot \Delta p_i) = |c_i| \cdot \Delta_i$, so the range of $P_f$ is

$$\left[\tilde{P} - \Delta, \tilde{P} + \Delta\right], \quad \text{where} \quad \Delta = \sum_{i=1}^{m} |c_i| \cdot \Delta_i.$$
4. Towards an Algorithm

• To compute $\Delta = \sum_{i=1}^{m} |c_i| \cdot \Delta_i$, we need to find $c_i$.

• If we replace one of $\tilde{p}_i$ with $\tilde{p}_i + \Delta_i$, then, due to linearization, we get

$$P_i \overset{\text{def}}{=} P_f(\tilde{p}_1, \ldots, \tilde{p}_{i-1}, \tilde{p}_i + \Delta_i, \tilde{p}_{i+1}, \ldots, \tilde{p}_m) = \tilde{P} + c_i \cdot \Delta_i.$$ 

• Thus, $|c_i| \cdot \Delta_i = \left| P_i - \tilde{P} \right|$ and hence $\Delta = \sum_{i=1}^{m} \left| P_i - \tilde{P} \right|$.

• Algorithm: compute $\tilde{P} = P_f(\tilde{p}_1, \ldots, \tilde{p}_m)$, $m$ values $P_i = P_f(\tilde{p}_1, \ldots, \tilde{p}_{i-1}, \tilde{p}_i + \Delta_i, \tilde{p}_{i+1}, \ldots, \tilde{p}_m)$, then

$$\Delta = \sum_{i=1}^{m} \left| P_i - \tilde{P} \right| \quad \text{and} \quad \left[ \tilde{P} - \Delta, \tilde{P} + \Delta \right].$$

• This algorithm requires $m+1$ calls to MCA: to compute $\tilde{P}$ and $m$ values $P_i$. 
5. Towards a Faster Algorithm

- When the number of parameters $m$ is large, $m + 1$ calls may be too long.

- We can use the property of Cauchy distribution

  \[
  \rho_{\delta}(x) = \frac{\delta}{\pi} \cdot \frac{1}{1 + \frac{x^2}{\delta^2}}
  \]

  - if $\eta_i$ are independently Cauchy-distributed with parameters $\Delta_i$,

  - then $\eta \overset{\text{def}}{=} \sum_{i=1}^{m} c_i \cdot \eta_i$ is Cauchy-distributed with parameter $\Delta = \sum_{i=1}^{m} |c_i| \cdot \Delta_i$.

- Once we get simulated Cauchy-distributed values $\eta$, we can estimate $\Delta$ by the Maximum Likelihood method.

- We also need to scale $\eta_i$ to the interval $[-\Delta_i, \Delta_i]$ on which the linear approximation is applicable.
6. Faster Algorithm

- First, we compute \( \tilde{P} = P_f(\tilde{p}_1, \ldots, \tilde{p}_m) \).
- For some \( N \) (e.g., 200), for \( k = 1, \ldots, N \), repeat:
  - use the random number generator to compute \( r_i^{(k)} \), \( i = 1, 2, \ldots, m \), uniformly distributed on \([0, 1]\);
  - compute Cauchy distributed values as \( c_i^{(k)} = \tan(\pi \cdot (r_i^{(k)} - 0.5)) \);
  - compute the largest value \( K \) of the values \( |c_i^{(k)}| \);
  - compute simulated “actual values” \( p_i^{(k)} = \tilde{p}_i + \frac{\Delta_i \cdot c_i^{(k)}}{K} \);
  - apply MCA and compute
  \[
  \Delta P^{(k)} = K \cdot \left( P_f\left( p_1^{(k)}, \ldots, p_i^{(k)}, \ldots, p_m^{(k)} \right) - \tilde{P} \right).
  \]
7. Faster Algorithm (cont-d)

• We have computed

\[ \Delta P^{(k)} = K \cdot \left( P_f \left( p_1^{(k)}, \ldots, p_i^{(k)}, \ldots, p_m^{(k)} \right) - \bar{P} \right). \]

• Then, we compute \( \Delta \in \left[ 0, \max_k |\Delta P^{(k)}| \right] \) by applying the bisection method to the equation

\[
\frac{1}{1 + \left( \frac{\Delta P^{(1)}}{\Delta} \right)^2} + \ldots + \frac{1}{1 + \left( \frac{\Delta P^{(N)}}{\Delta} \right)^2} = \frac{N}{2}.
\]

• We stop when we get \( \Delta \) with accuracy \( \approx 20\% \) (accuracy 1\% and 1.2\% is approximately the same).

• The Cauchy-variate algorithm requires \( N \approx 200 \) calls to MCA.

• So, when \( m \gg 200 \), it is much faster than the above linearization-based algorithm.

\[ \Delta P^{(k)} = K \cdot \left( P_f \left( p_1^{(k)}, \ldots, p_i^{(k)}, \ldots, p_m^{(k)} \right) - \bar{P} \right). \]
8. Need to Take Model Inaccuracy Into Account

• Often, the given finite-parametric family of distributions is only an approximation.

• Similarly, the given family of limit functions is only an approximation.

• As a result of the model uncertainty:
  
  – the value \( C = C_f(p_1, \ldots, p_m) \) produced by the approximate model
  
  – is, in general, different from the actual failure probability \( P = P_f(p_1, \ldots, p_m) \).

• Often, the only available information about the model inaccuracy \( C - P \) is the upper bound \( \delta: |C - P| \leq \delta \).

• Let us analyze how this inaccuracy affects our estimations.
9. What If We Use the Linearized Algorithm

- In the linearized case, the desired upper endpoint $\overline{C}$ has the form $\overline{C} = \tilde{C} + \sum_{i=1}^{m} |\tilde{C} - C_i|$.  

- Each of the values $\tilde{C}$ and $\tilde{C}_i$ is accurate only with accuracy $\delta$.  

- As a result, $|\overline{C} - \overline{P}| \leq (2m + 1) \cdot \delta$.  

- Similarly, $|\overline{C} - \overline{P}| \leq (2m + 1) \cdot \delta$.  

- As an interval which is guaranteed to contain the actual failure probability $\overline{P}$, we can thus take  

  $$\left[\overline{C} - (2m + 1) \cdot \delta, \overline{C} + (2m + 1) \cdot \delta\right].$$

- When the number $m$ of parameters is large, the approximation error $(2m + 1) \cdot \delta$ becomes significant.  

- How can we decrease this approximation error?
10. Analysis of the Problem

- The actual maximum \( P \) is attained when \( \Delta p_i = \varepsilon_i \cdot \Delta_i \), where \( \varepsilon_i \overset{\text{def}}{=} \text{sign}(c_i) \).
- For these \( \varepsilon_i \), we have \( C_f(\tilde{p}_1 + \varepsilon_1 \cdot \Delta_1, \ldots) \geq \bar{P} - \delta \).
- Thus, for \( \bar{C} \overset{\text{def}}{=} \max_{\varepsilon} C_f(\tilde{p}_1 + \varepsilon_1 \cdot \Delta_1, \ldots) \), we get \( \bar{C} \geq \bar{P} - \delta \).
- On the other hand, for each \( \varepsilon \), we have
  \[
  C_f(\tilde{p}_1 + \varepsilon_1 \cdot \Delta_1, \ldots) \leq P_f(\tilde{p}_1 + \varepsilon_1 \cdot \Delta_1, \ldots) + \delta.
  \]
- Since \( P_f(\tilde{p}_1 + \varepsilon_1 \cdot \Delta_1, \ldots) \leq \bar{P} \), we conclude that
  \[
  C_f(\tilde{p}_1 + \varepsilon_1 \cdot \Delta_1, \ldots) \leq \bar{P} + \delta.
  \]
- Thus, for \( \bar{C} = \max_{\varepsilon} C_f(\tilde{p}_1 + \varepsilon_1 \cdot \Delta_1, \ldots) \), we have
  \[
  \bar{C} \leq \bar{P} + \delta.
  \]
- So, the maximum \( \bar{C} \) provides a \( \delta \)-approximation to \( \bar{P} \).
11. First New Algorithm

- For all $2^m$ possible combinations of values $\varepsilon_1 \in \{-1, 1\}$, $\ldots$, $\varepsilon_m \in \{-1, 1\}$, we estimate $C_f(\tilde{p}_1 + \varepsilon_1 \cdot \Delta_1, \ldots)$.
- Then, we compute the largest of these estimates $\overline{C}$.
- We can then guarantee that $|\overline{C} - \overline{P}| \leq \delta$.
- Similarly, the smallest of the values $C_f(\tilde{p}_1 + \varepsilon_1 \cdot \Delta_1, \ldots)$ is $\delta$-close to $\overline{P}$: $|C - P| \leq \delta$.
- The above algorithm requires at least $2^m$ estimates, which for large $m$ is unrealistically large.
- It is known that if we want to find $\tilde{P}$ with accuracy $\delta$, we cannot use fewer than exponentially many calls.
- How can we decrease the uncertainty in estimating without increasing the number of calls too much?
12. Towards the Second New Algorithm

- Let us compute a new difference
  \[ P_f(\ldots, \tilde{p}_i + \Delta_i, \ldots) - P_f(\ldots, \tilde{p}_i - \Delta_i, \ldots) = 2c_i \cdot \Delta_i. \]

- When \( P_f(\ldots) \) is known with accuracy \( \delta \), we have \( 2c_i \cdot \Delta_i \) with accuracy \( 2\delta \), hence \( |c_i| \cdot \Delta_i \) with accuracy \( \delta < 2\delta \).

- So, for each \( i = 0, \ldots, m \), we compute
  \[ E_i = C_f(\tilde{p}_1 + \Delta_1, \ldots, \tilde{p}_i + \Delta_i, \tilde{p}_{i+1} - \Delta_{i+1}, \ldots, \tilde{p}_m - \Delta_m). \]

- We take the average \( \tilde{E} \) of all \( E_i \) as an estimate for \( \tilde{P} \), and \( \tilde{\Delta} \) as an estimate for \( \Delta \).

- This algorithm requires \( m + 1 \) calls to \( C_f \), but its accuracy is \((m + 1) \cdot \delta \) – twice better.
13. Towards the Third New Algorithm

- For $c_i \geq 0$, the maximum $\overline{P}$ is attained when $\Delta p_i = \Delta_i$.
- When can we conclude that $c_i > 0$?
- We know that $E_i - E_{i-1}$ is a $2\delta$-approximation to $2c_i \cdot \Delta_i$.
- So, if $E_i - E_{i-1} \geq 2\delta$, we can conclude that $c_i \geq 0$.
- Thus, after computing each $E_i$, we mark $i$ as $i \in S^+$ if $E_i - E_{i+1} \geq 2\delta$, and $i \in S^-$ if $E_i - E_{i+1} \leq -2\delta$.
- Let $s = \#(S^- \cup S^+)$ and $S_0 = -(S^- \cup S^+)$.
- Then, we compute $p_i^+ = \tilde{p}_i + \Delta_i$ when $i \in S^+$, $p_i^- = \tilde{p}_i - \Delta_i$ when $i \in S^-$, and $p_i^+ = \tilde{p}_i$ else.
- We then compute $C^+ = C_f(p_1^+, \ldots, p_m^+)$ and

$$
\overline{E} = C^+ + \frac{1}{2} \cdot \sum_{i \in S_0} |E_i - E_{i-1}| .
$$
14. Third New Algorithm (cont-d)

- Similarly, we compute \( p_i^- = \tilde{p}_i - \Delta_i \) when \( i \in S^+ \),
  \( p_i^- = \tilde{p}_i + \Delta_i \) when \( i \in S^- \), and \( p_i^- = \tilde{p}_i \) when \( i \in S_0 \).

- We then compute \( C^- = C_f(p_1^-, \ldots, p_m^-) \) and
  \[
  \overline{E} = C^- - \frac{1}{2} \cdot \sum_{i \in S_0} |E_i - E_{i-1}|. 
  \]

- The estimates \( \overline{E} \) and \( E \) approximate \( \overline{P} \) and \( P \) with accuracy \((m + 1 - s) \cdot \delta\).

- This algorithm requires \( m + 3 \) calls to \( C_f \).

- \( s \) is large if there are many parameters which significantly affect the failure probability.

- In this case, we get a drastic improvement in accuracy – at the expense of having only two more calls to \( C_f \).
15. Similar Modification of the First Algorithm

- First, we compute the differences
  \[ E_i = C_f(\tilde{p}_1 + \Delta_1, \ldots, \tilde{p}_i + \Delta_i, \tilde{p}_{i+1} - \Delta_{i+1}, \ldots, \tilde{p}_m - \Delta_m) \].

- We mark \( i \) as \( i \in S^+ \) if \( E_i - E_{i+1} \geq 2\delta \), as \( i \in S^- \) if \( E_i - E_{i+1} \leq -2\delta \), else as \( i \in S_0 \).

- To estimate \( \overline{P} \), we estimate \( 2^{m-s} \) values
  \[ C_f(\tilde{p}_1 + \varepsilon_1 \cdot \Delta_1, \ldots) \), where:
  - \( \varepsilon_i = 1 \) for \( i \in S^+ \),
  - \( \varepsilon_i = -1 \) for \( i \in S^- \), and
  - we take all possible combinations of the values \( \varepsilon_i \in \{-1, 1\} \) for the remaining \( m-s \) indices \( i \in S_0 \).

- The largest of these estimates is then returned as an estimate \( \overline{C} \) for \( \overline{P} \).
16. Modified First Algorithm (cont-d)

- To estimate $\underline{P}$, we estimate $2^{m-s}$ values

$$C_f(\tilde{p}_1 + \varepsilon_1 \cdot \Delta_1, \ldots),$$

where:

- $\varepsilon_i = -1$ for $i \in S^+$,
- $\varepsilon_i = 1$ for $i \in S^-$, and
- we take all possible combinations of the values $\varepsilon_i \in \{-1, 1\}$ for the remaining $m - s$ indices $i \in S_0$.

- The smallest of these estimates is then returned as an estimate $\underline{C}$ for $\underline{P}$.

- We get the same accuracy $\delta$ with $(m+1)+2 \cdot 2^{m-s} \ll 2^m$ calls to $C_f$. 

17. Towards a Fourth Algorithm

- To get the best accuracy $\delta$, we need all $2^{m-s}$ combinations of $m-s$ values $\varepsilon_i = \pm 1$.

- If we only have time for $\leq m + 3$ combinations, then we can gain accuracy $\leq (m - s + 1) \cdot \delta$.

- What if we have more computation time but still not enough to try all $2^{m-s}$ combinations?

- It then makes sense to select $g > 1$ and divide $m-s$ parameters $i \in S_0$ into $\frac{m-s}{g}$ groups $G_\ell$ of size $g$.

- Within each $G_\ell$, we try all combinations, to estimate:
  - $s_1^+ \overset{\text{def}}{=} \tilde{P} + \sum_{i \in G_1} |c_i| \cdot \Delta_i$ (for the first group) and
  - $s_\ell \overset{\text{def}}{=} \sum_{i \in G_\ell} |c_i| \cdot \Delta_i$ (for all other groups $\ell > 1$).

- Then, we add up the resulting estimates (same for $\overline{P}$).
18. Towards a Similar Modification of the Cauchy Variate Algorithm

- Due to model inaccuracy, we only know the values \( P(p_1^{(k)}, \ldots) \) and \( \tilde{P} \) with accuracy \( \delta \).

- Thus, the computed value \( \tilde{\Delta}^{(k)} = K \cdot \left( C_f \left( p_1^{(k)}, \ldots \right) - \tilde{C} \right) \) is \((2K \cdot \delta)\)-close to the desired values \( \Delta P^{(k)} \):

\[
\Delta P^{(k)} \in \left[ \tilde{\Delta}^{(k)} - 2K \cdot \delta, \tilde{\Delta}^{(k)} + 2K \cdot \delta \right].
\]

- In the formula for \( \Delta \), \( \Delta \) increases with each

\[
s_k \overset{\text{def}}{=} \left( \Delta P^{(k)} \right)^2.
\]

- Thus, to find the largest possible value of \( \Delta \), we need to take the largest possible value \( \bar{s}_k \) of \( \left( \Delta P^{(k)} \right)^2 \):

\[
\bar{s}_k = \left( \left| \tilde{\Delta}^{(k)} \right| + 2K \cdot \delta \right)^2.
\]
19. Resulting Algorithm

- First, we estimate \( \tilde{C} = C_f(\tilde{p}_1, \ldots, \tilde{p}_m) \).
- Then, we compute \( K \) and \( p_i^{(k)} \) as in the original Cauchy deviate algorithm, and estimate \( \tilde{\Delta}^{(k)} \) and
  \[
  \bar{s}_k = \left( |\tilde{\Delta}^{(k)}| + 2K \cdot \delta \right)^2.
  \]
- After that, we compute \( \Delta \in \left[ 0, \sqrt{\max_k \bar{s}_k} \right] \) by applying the bisection method to the equation
  \[
  \frac{1}{1 + \frac{\bar{s}_1}{\Delta^2}} + \ldots + \frac{1}{1 + \frac{\bar{s}_N}{\Delta^2}} = \frac{N}{2}.
  \]
- Finally, we return the range \( [\tilde{C} - \Delta, \tilde{C} + \Delta] \).
20. Parallelization and Acknowledgments

- One can easily see that the above algorithms are easily parallelizable.

- This work was supported in part by the National Science Foundation grants:
  - HRD-0734825,
  - HRD-1242122, and
  - DUE-0926721.