Interactive Uncertainty Quantification Visualization: Project Proposal

Matthew Isaac *

1. Introduction

Uncertainty quantification is a methodological framework used with some frequency in engineering analysis (Ewing et al., 2018). It is used to understand how variability within the parameters (i.e. inputs) to some system impact the end state of that system. Engineering analysts use uncertainty quantification to assess and find the balance between design sufficiency and design efficiency. This is particularly important in fields where large-scale prototypes, testing, and data collection are extremely expensive. Engineers in these types of applications are increasingly relying on computational simulations to assess system designs.

In the following paper, I propose to implement an interactive tool to visualize uncertainty quantification results. The outline of this proposal will proceed as follows: In Section 2, I will give a brief overview of uncertainty quantification. In Section 3, I will describe the features I propose to implement as part of this visualization tool. In Section 4, I will indicate which `R` packages and methods are anticipated as being necessary for the implementation.

2. Uncertainty Quantification Overview

The following description of the uncertainty quantification algorithm is adapted from Ewing et al. (2018). First, a few terms and definitions will be described.

system response quantity (SRQ): A parameter of particular interest directly related to the engineering system in question. The SRQ is the output (i.e. prediction) from the engineering model.

engineering model: A mathematical model that defines the relationship between the parameters (model inputs) and SRQ (model output).

aleatory uncertainty: Uncertainty resulting from randomness inherent to a given parameter. Gaining more knowledge about the parameter will not reduce the uncertainty of the parameter.

epistemic uncertainty: Uncertainty resulting from a lack of knowledge about a given parameter. Gaining more knowledge about the parameter could reduce the uncertainty of the parameter.

The first step of uncertainty quantification is to identify sources of uncertainty (model parameters) and classify them as either aleatory or epistemic. This classification process has been somewhate debated in literature (Kiureghian and Ditlevsen, 2009), and will not be discussed as it is outside the scope of this project. Once these uncertainties related to the

*Department of Mathematics and Statistics, Utah State University, Logan, UT 84322–3900, USA. E-mail: matt.isaac@aggiemail.usu.edu

model parameters have been identified and classified as aleatory or epistemic, the uncertainty for each parameter must somehow be described. Traditionally, epistemic uncertainties have been described by an interval over which any value in the interval is equally likely, while aleatory uncertainties have been assigned probability distributions. Some more recent publications (Ewing et al., 2018) propose that all uncertainties, aleatory or epistemic, be described using probability distributions. This debate will not be discussed here as it is, again, outside the scope of this project. Once these distributions and/or intervals have all been assigned, they are carried through the model using Monte Carlo techniques.

Let m denote the number of iterations of an outer for loop, and let n denote the number of iterations in an inner for loop. In the outer for loop, values for the epistemic uncertainty parameters are selected randomly from the intervals/distributions assigned. Upon entering the inner loop, values for the aleatory uncertainty parameters are randomly chosen from the distributions assigned. The values chosen for the parameters in both the outer loop and the inner loop are then used as inputs in the engineering model to calculate a value for the SRQ. This value is stored and the inner loop continues running for the rest of the $n - 1$ iterations. All of the n SRQ values calculated from the n iterations of the inner loop are used to calculate an empirical cumulative distribution function (CDF) of SRQ values and the CDF is stored. The outer loop then begins its second iteration, and new values for the empirical uncertainty parameters are chosen. The inner loop then runs another n iterations, producing another CDF. This process continues until the outer loop has run all of its m iterations.

At this point, there will be m empirical CDFs that have been calculated, representing various possible realized values of the SRQ. These CDFs can then be plotted and interpreted. Ewing et al. (2018) suggests constructing a "P-box". This P-box is found by calculating a lower percentile (e.g. the 5th percentile) and an upper percentile (e.g. the 95th percentile) of the CDF ensemble. This P-box can then be interpreted in several ways, including (1) selecting a SRQ value and extracting a probability interval, and (2) selecting a probability value and extracting an SRQ interval.

3. Implementation

I propose to implement an interactive visualization tool to assist analysts in visualizing and interpreting results from an uncertainty quantification analysis as described in Section 2. Since the actual uncertainty quantification analysis can likely be carried out with greater speed and computational power elsewhere, this implementation will not include the Monte Carlo portion of the implementation described in Section 2. Users will begin by uploading a `.csv` file, which should contain the ensemble of CDFs generated from the Monte Carlo process. I also plan to include several sample data sets that users can select and use to experiment with. The CDFs will be plotted with the x-axis displaying the SRQ values and the y-axis displaying the probability values. Once the ensemble of CDFs have been uploaded, or a sample data set selected, users will be able to interact with the visualization in various ways, including:

- toggle P-box on/off
- select percentiles to be used in P-box calculation
- toggle CDF ensemble on/off (so only P-box is displayed)
- select level of transparency for CDF ensemble
- input a probability value and extract (display) an SRQ interval

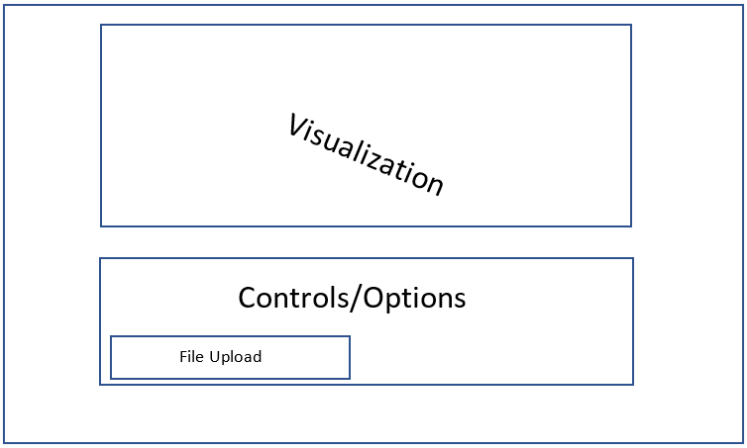

Figure 1: Proposed layout of web application.

- input a SRQ value and extract (display) a probability interval
- download and save the visualization in its current state

The prototype layout is shown in a diagram in Figure 1. The web application will consist of a main pane which will display the graphical visualization. A control panel below the plot will hold the controls (toggle buttons, check boxes, sliders, `.csv` upload) to be used when interacting with the visualization.

4. `R` Packages and Methods

I anticipate the following packages and methods being useful and/or necessary in carrying out this implementation.

4.1 `dplyr`

The `dplyr` package (Wickham et al., 2013) is a data-wrangling and manipulation package implemented in `R`. The methods in this package will be used to manage and manipulate the CDFs from the `.csv` file into a convenient format for plotting.

4.2 `shiny`

The `shiny` package and framework (Chang et al., 2018) will be the backbone of this project. `shiny` provides a way to create and deploy web applications through RStudio (RStudio Team, 2016). It also contains the implementations for all user-interface components (toggle buttons, check boxes, numeric inputs, sliders, etc.). A package such as `shinydashboard` (Chang and Borges Ribeiro, 2018) may also be used as a aesthetic wrapper around the `shiny` framework. This package will be used to create and deploy the actual web application. The controls in the we app will be made from the user-interface components in this package.

4.3 `ggplot2`

The plotting functionality of the `ggplot2` package (Wickham, 2016) will be used to generate the actual visualization and to add, remove, or adjust components on the plot.

4.4 plotly

The plotly package (Sievert, 2018) will be used to add additional interactive capabilities to the plot. plotly includes a method called ggplotly() which will be used to convert the ggplot plot object to a plotly plot object. The plotly plots contain options to zoom in and out on a plot, show plot values when hovering with mouse, download and save a .png version of the plot, and download and save an interactive html version of the plot.

Since the existing R packages already implement most of the tools I will need to generate the plot and implement user interface elements, my primary objective on this project will be to seamlessly combine elements from the packages above (particularly the shiny, ggplot2, and plotly packages) to create a user-friendly interactive visualization tool.

References

Chang, W., Borges Ribeiro, B., 2018. shinydashboard: Create Dashboards with 'Shiny'. R package version 0.7.1 (https://CRAN.R-project.org/package=shinydashboard).

Chang, W., Cheng, J., Allaire, J., Xie, Y., McPherson, J., 2018. shiny: Web Application Framework for R. R package version 1.2.0 (https://CRAN.R-project.org/package=shiny).

Ewing, M., Liechty, B. C., Black, D., 2018. A General Methodology for Uncertainty Quantification in Engineering Analyses Using a Credible Probability Box. Journal of Verification, Validation, and Uncertainty Quantification 3 (2), https://doi.org/10.1115/1.4041490.

Kiureghian, A. D., Ditlevsen, O., 2009. Aleatory or Epistemic? Does it Matter? Structural Safety 31 (2), http://www.sciencedirect.com/science/article/pii/S0167473008000556.

RStudio Team, 2016. RStudio: Integrated Development Environment for R. RStudio, Inc., Boston, MA, (http://www.rstudio.com/).

Sievert, C., 2018. plotly for R. (https://plotly-book.cpsievert.me).

Wickham, H., 2016. ggplot2: Elegant Graphics for Data Analysis. Springer-Verlag New York, (http://ggplot2.org).

Wickham, H., Francois, R., Henry, L., Muller, K., 2013. dplyr: A Grammer of Data Manipulation. R package version 0.7.6. (https://CRAN.R-project.org/package=dplyr).