Surrogate Modeling and Uncertainty Quantification of Maximum Cauchy Stress in Solar Sail Membranes

— Project Report —

Fallstudien II / Case Studies II

Lecturers:

Prof. Dr. Katja Ickstadt
Dr. Henrike Weinert
Yassine Talleb
Carmen van Meegen

Author:

Rahul Ramesh Vishwkarma
Matriculation number: 236862

Group number: 1

Group members:

Anirudh Parameswaran
Harsh Yadav

January 12, 2026

TU Dortmund University

Contents

1 Introduction **1**

2 Problem definition **1**
2.1 Dataset Overview 1
2.2 Variable description 2
2.3 Data quality 3

3 Statistical Methods **3**
3.1 Optimized Latin Hypercube Sampling 3
3.2 Surrogate Models 5
3.2.1 Support Vector Regression 5
3.2.2 Polynomial Regression 6
3.2.3 Gaussian process regression 7

4 Statistical Analysis **10**
4.1 How accurately do surrogate models approximate the maximal Cauchy stress in a sun-sail membrane? 10
4.1.1 Residual behavior of surrogate models 10
4.1.2 Predictive accuracy of surrogate models (MSE) 12
4.2 How does uncertainty in the inputs induce uncertainty in the output? . . 12
4.3 Findings 13

5 Conclusion **14**

6 Appendix **16**
6.1 Input parameters for the structural membrane system 16
6.2 Summary statistics of Maximal cauchy stress 16

1 Introduction

In modern engineering and applied sciences, high-fidelity numerical simulations are key tools for studying complex structures and materials under realistic conditions. These computer experiments map uncertain input parameters, such as material properties, loads, and boundary conditions, to output quantities of interest. However, their high computational cost makes large uncertainty or design studies costly when many model evaluations are required. Surrogate modeling addresses this challenge by replacing the original computational model with a cheaper statistical or machine-learning approximation, which allows many evaluations for probabilistic analyses and sensitivity studies while keeping the accuracy sufficient for reliable engineering decisions.

The aim of this project is to develop and assess surrogate models for predicting the maximal membrane Cauchy stress in a sun-sail structure under uncertain material and loading conditions, using a synthetic dataset generated by optimized space-filling sampling and finite element simulations with *Kratos Multiphysics*.[Mataix Ferrándiz et al., 2023] Three surrogate approaches are considered: Support Vector Regression (SVR), Polynomial Regression (PLY), and Gaussian Process Regression (GPR). Their accuracy is evaluated using residual analysis and mean squared error. The best-performing model is then used in Monte Carlo simulations to propagate input uncertainty and to characterize the resulting distribution of maximal membrane Cauchy stress.

The second section describes the prestressed sun-sail membrane system, defines all random and deterministic input variables, and explains how the finite element model and the dataset are constructed. The third section presents the optimized sampling design and outlines the surrogate modeling techniques, including their mathematical formulations. The fourth section details the split of the data into training and test sets, compares the surrogate models using residual plots and mean squared error, performs Monte Carlo uncertainty propagation with the best surrogate, and analyzes both the empirical and parametric distributions of the maximal membrane Cauchy stress.

2 Problem definition

2.1 Dataset Overview

The dataset used in this study is generated from a planned computational experiment rather than from physical measurements. Each data point corresponds to a single simulation performed with the *Kratos Multiphysics* software, which uses the finite element method (FEM) to compute the mechanical response of a sun-sail structure under specified loading and material conditions. Because these simulations are computationally expensive, it is not feasible to sample all possible input combinations exhaustively.

Instead, an optimized experimental design is adopted, namely an **Optimized Latin Hypercube** (OLH) sampling scheme based on genetic algorithms, to select a representative set of input samples. The study uses a total of $n_{\max} = 200$ simulations as the available sample size. This corresponds to a planned computer experiment with strati-

fied coverage over the six independent input variables, ensuring a nearly uniform spread across the multidimensional parameter space while preserving statistical independence between variables.

Each simulation produces a single scalar output, the maximal membrane Cauchy stress, together with a six-dimensional input vector. The resulting dataset therefore consists of 200 input–output pairs, each combining six input variables with one corresponding value of the maximal Cauchy stress in the membrane.

2.2 Variable description

The structural membrane system is defined by several material and loading parameters associated with the membrane, truss, edge cables, and support cables. All quantities listed in Table 1 are required as input to the *Kratos Multiphysics* finite element model, even though only a subset is treated as uncertain in the surrogate modeling and uncertainty quantification.

Six membrane-related parameters, together with the pre-stress values in the edge and support cables, are modeled as random variables and drive the probabilistic characterization of the structural response. The rupture stress, denoted by $\sigma_{\text{mem},y}$ (in MPa), represents the material strength limit used in failure assessment. Although it does not enter the finite element computation directly, it is used in post-processing to compare against the predicted maximal Cauchy stress, denoted by $\sigma_{\text{mem,max}}$ (in kPa) and to classify each simulation as safe or failed.

The input parameters, which are given as random input parameters, are as follows:

Random Variables Considered in the Model:

- **Membrane Young's modulus** E_{mem} [GPa] — quantifies the elastic stiffness of the membrane and governs its overall rigidity under tensile loading.
- **Membrane Poisson's ratio** ν_{mem} [–] — defines the ratio of transverse to longitudinal strain; controls coupling between orthogonal in-plane deformation modes.
- **Membrane pre-stress** σ_{mem} [MPa] — isotropic initial tension applied prior to external loading; stabilizes membrane geometry and influences its deformation response.
- **Surface loading** f_{mem} [kPa] — external distributed pressure acting on the membrane surface; represents the primary excitation driving structural response.
- **Edge cable pre-stress** σ_{edg} [MPa] — initial tension within edge cables that determines boundary stiffness and facilitates load transfer along the perimeter.
- **Support cable pre-stress** σ_{sup} [MPa] — initial tension within support cables contributing to global equilibrium and maintaining structural stability of the sail configuration.

In addition to the membrane parameters, the global structural model implemented in *Kratos Multiphysics* requires material and geometric properties for the truss, edge cables, support cables, and the membrane thickness. These quantities are essential for defining the overall structural stiffness and load-transfer behavior but are treated as deterministic within the uncertainty analysis. The truss is characterized by its Young's modulus E_{tru} (GPa) and cross-sectional area A_{tru} (cm^2), which together determine its axial rigidity and resistance to deformation. The membrane thickness t_{mem} (mm) is specified as a fixed design value that affects both the size of the stresses and how they are distributed in the membrane. The edge and support cables are defined by their Young's moduli (E_{edg}, E_{sup} in GPa) and diameters (d_{edg}, d_{sup} in mm), which set their tensile stiffness and help keep the membrane properly tensioned under load.

It should be noted that not all variables listed in Table 1 are uncertain in the surrogate-modeling or uncertainty-quantification sense. A subset, primarily the membrane properties and cable pre-stresses, is modeled as random to capture variability in material behavior and loading conditions. The remaining quantities such as elastic moduli, cross-sectional areas, and diameters are treated as deterministic constants because they represent fixed design attributes of the structure. Nevertheless, all parameters are essential inputs for each finite element simulation carried out in *Kratos Multiphysics*. The primary model output analyzed in this study is the maximum Cauchy stress within the membrane $\sigma_{\text{mem,max}}$. This quantity is computed for every simulation case and serves as the key response variable for developing surrogate models and performing uncertainty propagation.

2.3 Data quality

All data in this study are obtained from deterministic numerical simulations rather than physical experiments. Consequently, there are no missing values or direct measurement errors, and identical input parameters always produce the same outputs in the *Kratos Multiphysics* finite element solver. Small numerical variations may occur due to mesh discretization or solver tolerances, but these effects are minor compared to the overall response levels. The dataset can therefore be treated as essentially noise-free and is well suited for surrogate modeling without additional preprocessing steps such as denoising or data imputation.

3 Statistical Methods

3.1 Optimized Latin Hypercube Sampling

In this section, the input designs used to train and evaluate the surrogate models are generated using an optimized Latin hypercube sampling (LHS) strategy. The aim is to obtain a space-filling design in the p-dimensional input space while limiting the total number of model evaluations to $n_{\max} = 200$, since each evaluation requires running the computationally expensive *Kratos Multiphysics* finite element model [Mataix Ferrándiz

et al., 2023].

Latin Hypercube (LH) design

A Latin hypercube (LH) design can be represented by a $n \times p$ matrix L with n rows 1, where each column is a permutation of the integers $1, \ldots, n$. Each row of L corresponds to one discrete sample point, and we write

$$L = \begin{bmatrix} u_1 \\ \vdots \\ u_n \end{bmatrix} = \begin{bmatrix} u_{11} & \cdots & u_{1p} \\ \vdots & \ddots & \vdots \\ u_{n1} & \cdots & u_{np} \end{bmatrix}, \tag{1}$$

where $u_i \in [0, 1]$ and corresponds to the i-th sample point. The goal is to obtain n points in a p-dimensional unit cube, $[0, 1]^p \subseteq \mathbb{R}^p$ and the main challenge is to construct an LH design that is well optimized in this space.

Genetic algorithm for LH optimization

To optimize Latin hypercubes, the following force-based criterion is used [Liefvendahl & Stocki, 2005],

$$G(L) = \sum_{i=1}^{n} \sum_{i'=i+1}^{n} \frac{1}{\|u_i - u_{i'}\|^2}, \tag{2}$$

which can be interpreted as the total repulsive force between sample points when they are viewed as electrically charged particles. This objective, $G(L)$ as defined in equation 2, serves as the fitness function in the genetic algorithm. The genetic algorithm is a general-purpose optimization method that iteratively improves a population of candidate LH designs. Its steps are implemented as follows.

Step 1: Initial population: An initial population $\{L_k\}_{k=1}^{N_{\text{pop}}}$ of random Latin hypercubes is generated, where N_{pop} is chosen to be even.

Step 2: Selection: After evaluating all Latin hypercubes in the population, the best $N_{\text{pop}}/2$ individuals, i.e. those L_k with the lowest values of $G(L_k)$, are retained as survivors, and the remaining $N_{\text{pop}}/2$ Latin hypercubes are discarded.

Step 3: Crossover: From the $N_{\text{pop}}/2$ survivors, a new population of N_{pop} children is generated as follows. Let L_{best} denote the current best Latin hypercube among the survivors. This Latin hypercube is copied as child 1 and as child $N_{\text{pop}}/2 + 1$. For each survivor with index $k \in \{2, \ldots, N_{\text{pop}}/2\}$, two children are then constructed:

- Child k is obtained by taking L_{best} and replacing one randomly chosen column with the corresponding column of L_k.
- Child $N_{\text{pop}}/2 + k$ is obtained by taking L_k and replacing one randomly chosen column with the corresponding column of L_{best}.

Step 4: Mutation: Mutation is applied to all children except the best Latin hypercube L_{best} (child 1). For each child Latin hypercube and for each column, a random number uniformly distributed in $[0, 1]$ is drawn; if this number is less than the prescribed mutation probability p_{mut}, two randomly chosen elements within that column are swapped. This operation preserves the Latin hypercube structure while introducing additional diversity into the population.

Step 5: Stopping criterion: The selection, crossover, and mutation steps are repeated over successive generations. After every fixed number of generations, the accumulated improvement in the best force criterion value $G(L_k)$ over this interval is evaluated; if this improvement falls below a prescribed tolerance, the algorithm is terminated. Moreover, the procedure is stopped once a predefined maximum number of generations has been reached.

Mapping to the physical input space

The optimized LHS design L yields n points in the unit hypercube $[0, 1]^p$. However, the physical input variables are modeled as random variables with prescribed marginal distributions. To transform the optimized design into physically meaningful input samples, the inverse cumulative distribution function (CDF) is applied as in (3). Specifically, for each design point $i = 1 \dots n$ and each input variable $j = 1 \dots p$, the physical realization x_{ij} is obtained as

$$x_{ij} = F_j^{-1}(u_{ij}) \,, \tag{3}$$

where $u_{ij} \in (0, 1)$ denotes the corresponding normalized LHS coordinate and F_j is the CDF of the j-th input variable. This transformation ensures that each column of the resulting design matrix $X \in \mathbb{R}^{n \times p}$ follows the prescribed marginal distribution while preserving the optimized space-filling properties of the LHS design.

3.2 Surrogate Models

3.2.1 Support Vector Regression

Support Vector Regression (SVR) seeks a function that balances prediction accuracy and model flatness by minimizing a regularized objective function. SVR employs an ε-insensitive loss, which ignores errors with absolute value below ε and penalizes only larger deviations. The parameter ε defines the width of an ε-tube around the regression function: decreasing ε narrows the tube, increases the number of support vectors, and reduces sparsity, whereas increasing ε widens the tube and typically yields fewer support vectors. Because small perturbations inside the tube are not penalized, the ε-insensitive region enhances robustness to noise. Alternative convex loss functions, such as linear, quadratic, and Huber losses, may also be used; the Huber loss is smoother and combines robustness for small errors with stronger penalties for large deviations. The choice of loss depends on prior knowledge of the noise affecting the data, the desired sparsity of the solution, and computational considerations [Awad & Khanna, 2015].

In the linear ε-insensitive SVR formulation, assuming that the intercept term is already included in the input vector $x_i \in \mathbb{R}^{p+1}$, the optimization problem is

$$\min_{w,\xi_i,\xi_i^*} \quad \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}(\xi_i + \xi_i^*), \tag{4}$$

subject to

$$\begin{aligned} y_i - w^\top x_i &\le \varepsilon + \xi_i, \quad i = 1,\dots,n, \\ w^\top x_i - y_i &\le \varepsilon + \xi_i^*, \quad i = 1,\dots,n, \\ \xi_i, \xi_i^* &\ge 0, \qquad\quad i = 1,\dots,n, \end{aligned} \tag{5}$$

where $w \in \mathbb{R}^{p+1}$ is the weight vector of the linear regression function. Here, w defines the regression function, ξ_i and ξ_i^* are slack variables measuring violations of the ε-tube, and $C > 0$ is a regularization parameter that controls the trade-off between minimizing the norm of w (model flatness) and penalizing prediction errors.

The Kernel Trick in Support Vector Regression

The linear SVR formulation in (4)–(5) assumes that the regression function $f(x)$ is linear in the input space. SVR can be extended to nonlinear relationships by mapping the inputs into a (typically higher-dimensional) feature space via a transformation $\varphi(x)$ and seeking a function of the form [Awad & Khanna, 2015]

$$f(x) = w^\top \varphi(x), \tag{6}$$

where the intercept term is included in $\varphi(x)$. Directly working with $\varphi(x)$ is usually impractical, so SVR employs the kernel trick: inner products $\varphi(x_i)^\top \varphi(x_{i'})$ are replaced by a positive definite kernel function $k(x_i, x_{i'})$ satisfying Mercer's condition, leading to a dual formulation that depends on the training data only through kernel evaluations.

In the kernel SVR setting, the regression function can be written as

$$f(x) = \sum_{i=1}^{n}(\alpha_i - \alpha_i^*)\, k(x_i, x), \tag{7}$$

where $k(\cdot,\cdot)$ is the chosen kernel, and α_i, α_i^* are dual variables obtained from the SVR dual optimization problem. Only data points with nonzero values $\alpha_i - \alpha_i^*$ (the support vectors) contribute to the prediction, and typical choices for $k(\cdot,\cdot)$ include linear, polynomial, and radial basis function (RBF) kernels, whose hyperparameters, together with C and ε, are usually selected by cross-validation.

3.2.2 Polynomial Regression

The polynomial regression model generalizes linear regression by augmenting it with polynomial functions of the predictors while remaining linear in the coefficients [Brenndoerfer, 2025]. For $x \in \mathbb{R}$, a polynomial model of degree d is written as

$$y = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3 + \cdots + \beta_d x^d + \varepsilon, \tag{8}$$

where y is the response variable, x is the independent variable, β_0 is the intercept, $\beta_1, \ldots, \beta_d$ are the coefficients of the polynomial terms, ε is a random error term (typically assumed to have mean zero and constant variance σ^2), and d denotes the degree of the polynomial.

In the multivariate setting with p predictors $x_1, x_2, \ldots, x_p$, polynomial regression includes higher powers of each variable as well as interaction terms (products of different predictors) up to the chosen degree d. A general polynomial model of degree d can be expressed schematically as

$$y = \beta_0 + \sum_{i=1}^{p} \beta_i x_i + \sum_{i=1}^{p} \sum_{j=i}^{p} \beta_{ij} x_i x_j + \sum_{i=1}^{p} \sum_{j=i}^{p} \sum_{k=j}^{p} \beta_{ijk} x_i x_j x_k + \cdots + \beta_{12\ldots p} x_1 x_2 \cdots x_p + \varepsilon, \quad (9)$$

where β_0 is the intercept, β_i are coefficients for the linear terms, β_{ij} correspond to quadratic terms and two-way interactions, β_{ijk} to cubic terms and three-way interactions, and so forth up to degree d. The error term ε again represents random noise. This construction allows the model to capture complex nonlinear relationships among the predictors and the response while preserving the linear-in-parameters structure needed for standard least-squares estimation.

3.2.3 Gaussian process regression

Gaussian process (GP) regression admits different but equivalent interpretations, most notably the weight-space and function-space views. In this work, the *function-space* perspective is adopted.

Function-space view of Gaussian processes

In the function-space view, a Gaussian process defines a prior distribution directly over functions. A Gaussian process is a collection of random variables such that any finite subset has a joint multivariate Gaussian distribution and is fully characterized by a mean function and a covariance (kernel) function.[Rasmussen & Williams, 2006]

For $x \in \mathbb{R}^p$ and a real-valued stochastic process $f(x)$, the mean function $m(x)$ and covariance function $k(x, x')$ are given by

$$m(x) = \mathbb{E}[f(x)], \qquad k(x, x') = \mathbb{E}\big[(f(x) - m(x))(f(x') - m(x'))\big], \quad (10)$$

and the process is denoted compactly as

$$f(x) \sim \mathcal{GP}\big(m(x), k(x, x')\big). \quad (11)$$

For notational convenience, the mean function $m(x)$ is often set to zero in applications, although nonzero mean functions can also be used when appropriate.

Posterior prediction with noise-free observations

In practice, the goal of Gaussian process regression is not merely to draw sample functions from the prior, but to update this prior using information contained in the training data. In the simplest case of noise-free observations, we assume $y_i = f_i$ that the pairs $\{(x_i, f_i)\}_{i=1}^n$ are known exactly. Collecting the corresponding function values into a vector f, and denoting by f_* the function values at a set of test inputs X_*, the joint prior distribution of f and f_* is multivariate Gaussian,

$$\begin{bmatrix} f \\ f_* \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} K(X,X) & K(X,X_*) \\ K(X_*,X) & K(X_*,X_*) \end{bmatrix} \right), \tag{12}$$

where $K(X,X)$ is the $n \times n$ covariance matrix over the training inputs, $K(X,X_*)$ is the $n \times n_*$ matrix of covariances between training and test inputs, and $K(X_*,X_*)$ is the $n_* \times n_*$ covariance matrix over the test inputs.

To obtain the posterior distribution over function values at the test points, we condition this joint Gaussian prior on the observed training outputs f. The resulting conditional distribution is

$$f_* \mid X_*, X, f \sim \mathcal{N}\big(\mu_*,\ \Sigma_*\big), \tag{13}$$

with posterior mean and covariance given by

$$\mu_* = K(X_*,X)\,K(X,X)^{-1} f, \tag{14}$$

$$\Sigma_* = K(X_*,X_*) - K(X_*,X)\,K(X,X)^{-1} K(X,X_*). \tag{15}$$

Sampling function values at the test inputs then amounts to evaluating μ_* and Σ_* drawing from this multivariate normal distribution. This formulation extends directly to multidimensional inputs by evaluating the covariance function on vector-valued arguments; the algebra is unchanged, although visualizing the resulting sample functions becomes more challenging in higher dimensions.

Posterior Prediction with noisy observations

In many practical modeling problems, the true function values are not directly observed; instead, only noisy measurements of the form $y = f(x) + \varepsilon$ are available, where ε denotes additive noise [Rasmussen & Williams, 2006]. Assuming that this noise is independent, identically distributed Gaussian with variance σ_n^2, the prior over the noisy observations has covariance

$$\operatorname{cov}(y_p, y_q) = k(x_p, x_q) + \sigma_n^2 \delta_{pq},$$

or, in matrix form,

$$\operatorname{cov}(y) = K(X,X) + \sigma_n^2 I,$$

where δ_{pq} is the Kronecker delta, equal to one if $p = q$ and zero otherwise, and I is the identity matrix.

Under this noise model, the joint prior distribution of the noisy training outputs and the latent function values at test inputs can be written as

$$\begin{bmatrix} y \\ f_* \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} K(X,X) + \sigma_n^2 I & K(X,X_*) \\ K(X_*,X) & K(X_*,X_*) \end{bmatrix} \right). \tag{16}$$

Conditioning on the observed data yields the Gaussian process regression predictive distribution

$$f_* \mid X, y, X_* \sim \mathcal{N}\big(\bar{f}_*,\ \mathrm{cov}(f_*)\big), \tag{17}$$

with mean

$$\bar{f}_* \coloneqq \mathbb{E}[f_* \mid X, y, X_*] = K(X_*,X)\big[K(X,X) + \sigma_n^2 I\big]^{-1} y, \tag{18}$$

and covariance

$$\mathrm{cov}(f_*) = K(X_*,X_*) - K(X_*,X)\big[K(X,X) + \sigma_n^2 I\big]^{-1} K(X,X_*). \tag{19}$$

Radial basis function kernel

The radial basis function (RBF) kernel, also known as the Gaussian kernel, is a popular choice for nonlinear support vector machines because it can model complex, smooth decision boundaries in the input space [Chang et al., 2010]. The kernel between two input vectors x_i and $x_{i'}$ is defined as

$$K(x_i, x_{i'}) = \exp\left(-\frac{\|x_i - x_{i'}\|^2}{2\sigma^2} \right), \tag{20}$$

where $\sigma > 0$ is a scale parameter that controls how quickly similarity decays as the distance between x_i and $x_{i'}$ increases. Small values of σ lead to highly localized kernels and flexible decision boundaries, while larger values of σ produce smoother, more global decision functions that resemble those of a linear classifier.

Resources

The analysis in this project was implemented in Python 3.12.10 [Python Software Foundation, 2025], using standard scientific computing and machine learning libraries. In particular, the following packages were employed: `pandas` and `numpy` for data handling and numerical computations, `matplotlib.pyplot` for visualization, and submodules of `scikit-learn` for model construction, training, and evaluation. In addition, a large language model (Perplexity) was used to refine the written text, improve grammar, and polish the presentation of explanations and results.

4 Statistical Analysis

In this study, a space-filling design is used to explore the input space efficiently. An optimized Latin hypercube of size 200×6 is first constructed in the unit cube $[0,1]^6$, providing a well-distributed set of points in $[0,1]^6$. Each column of this matrix is then transformed to the corresponding physical input variable by applying the appropriate inverse CDF element-wise. The resulting input samples are passed to the Kratos finite element model, which is run once per sample to compute the maximal membrane Cauchy stress $\sigma_{\text{mem,max}}$. The resulting dataset $\mathcal{D}$ of 200 input–output pairs is finally split at random into a training set $\mathcal{D}_{\text{train}}$ (80% of the data) for fitting the surrogate models and a test set $\mathcal{D}_{\text{test}}$ (20% of the data) for evaluating their predictive performance.

4.1 How accurately do surrogate models approximate the maximal Cauchy stress in a sun-sail membrane?

Three surrogate models are used to approximate the Kratos finite element model of the sun-sail membrane: support vector regression (SVR), polynomial regression (PLY), and Gaussian process regression (GPR). Each model is trained on $\mathcal{D}_{\text{train}}$ and assessed on $\mathcal{D}_{\text{test}}$ using residual plots for visual inspection and the mean squared error (MSE) as a quantitative measure of predictive accuracy.

4.1.1 Residual behavior of surrogate models

For the SVR model, the residual plot (see Figure 1, top panel) for the test data shows strong systematic errors over the full range of predicted values $\hat{\sigma}_{\text{mem,max}}$. The residuals are large almost everywhere, which points to low prediction accuracy.

At low predicted stresses, most residuals are negative, meaning that SVR tends to overestimate the true stresses in this range. At high predicted stresses, the residuals become positive and very large, so the model clearly underestimates the true response there.

In addition, the spread of the residuals grows with $\hat{\sigma}_{\text{mem,max}}$, so the errors are much larger at high stress levels than at low ones. This pattern indicates heteroscedasticity and that the SVR model is not robust in the upper tail of the response distribution.

For the polynomial regression model, the residual plot for the test data looks much more homogeneous than for SVR. The residuals stay close to zero over most of the range of predicted values $\hat{\sigma}_{\text{mem,max}}$, with no clear trend of systematic under- or overestimation.

However, two clearly negative residuals appear at higher predicted stresses, which means the polynomial model strongly overestimates the true response at these points. As shown in the middle panel of Figure 1, these two samples have much larger errors than the rest of the test data, indicating isolated outliers rather than a general lack of fit.

For the Gaussian process regression model, the residual-prediction plot for the test data (see Figure 1, bottom panel) shows the smallest residuals of all three models, tightly clustered around zero over the range of predicted values $\hat{\sigma}_{\text{mem,max}}$. This indicates that the

surrogate predictions match the Kratos model's responses very closely and do not show any systematic bias.

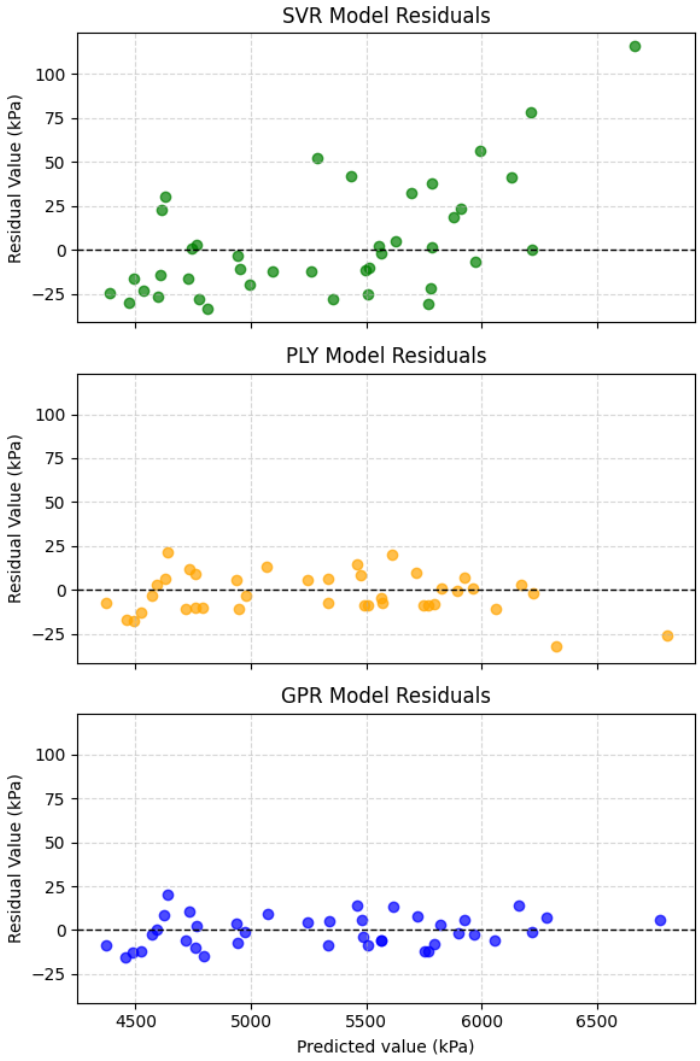

Figure 1: Residual plots on the test set for the SVR (top panel), polynomial regression (middle panel), and GPR models (bottom panel).

The two strong overestimations at high predicted stresses seen for the polynomial regression model do not appear in the GPR residuals, which shows that these errors are corrected by the Gaussian process model. Together with the lowest MSE values on both the training and test sets, this confirms GPR as the best-performing surrogate model in this study.

4.1.2 Predictive accuracy of surrogate models (MSE)

The MSE values for the three models are shown in Figure 2. SVR has the highest errors, with $\text{MSE}_{\text{train}} \approx 693.72$ and $\text{MSE}_{\text{test}} \approx 1080.81$, which means it approximates the Kratos response rather poorly. Using polynomial regression instead reduces the training MSE by about 583.47 (roughly 84%) and the test MSE by about 943.23 (about 87%).

Gaussian process regression improves the accuracy even more. It lowers the training MSE by about 626.08 compared to SVR (around 90% improvement) and reduces the test MSE by about 1000.53 (about 93% improvement). Overall, both PLY and GPR clearly outperform SVR on both training and test data, with GPR giving the most accurate approximation of the Kratos model among the three surrogates.

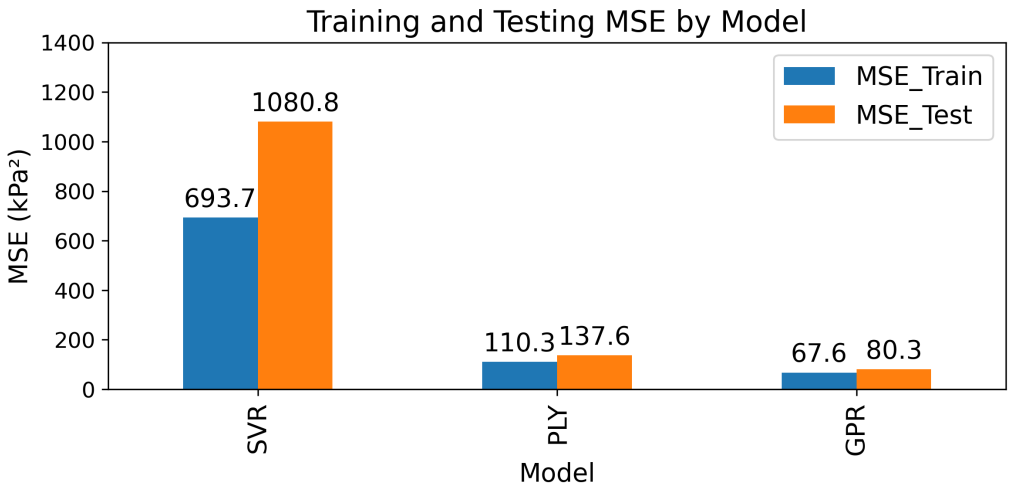

Figure 2: MSE comparison of SVR, polynomial regression, and GPR models on training and test data sets.

4.2 How does uncertainty in the inputs induce uncertainty in the output?

To study how uncertainty in the six input variables affects the maximal Cauchy stress $\sigma_{\text{mem,max}}$, a Monte Carlo (MC) [Sudret et al., 2017] simulation is performed. From the marginal distributions of the inputs, 10,000 independent samples are drawn and combined into 10,000 six-dimensional input vectors, each representing one sample from the six-dimensional input space.

For each of these 10,000 input vectors, the maximal membrane Cauchy stress $\sigma_{\text{mem,max}}$ is predicted with the best surrogate model, the Gaussian process regression (GPR) model, which yields 10,000 samples of $\sigma_{\text{mem,max}}$. These output samples are then used to approximate the probability distribution of $\sigma_{\text{mem,max}}$ and to compute summary statistics such as the mean, variance, and selected quantiles, describing how input uncertainty translates into variability of the structural response.

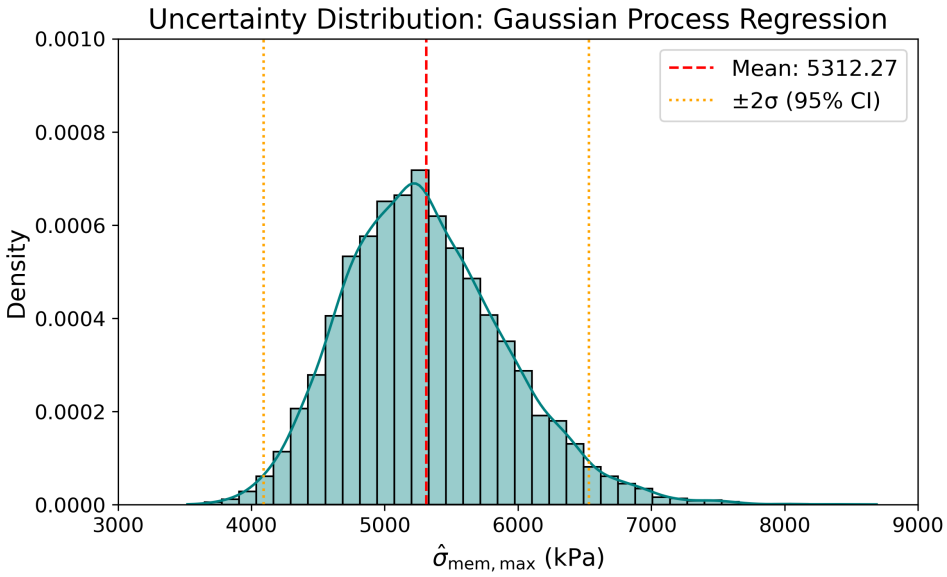

Figure 3: Empirical marginal distribution of $\sigma_{\text{mem,max}}$ from GPR predictions

To visualize the marginal distribution of the response, a histogram of the 10,000 samples of $\sigma_{\text{mem,max}}$ is plotted together with a kernel density estimate (KDE), as shown in Figure 3. In this figure, a red vertical line marks the empirical mean of $\sigma_{\text{mem,max}}$, and two yellow vertical lines show the bounds of the empirical 95% confidence interval, giving a clear graphical summary of the central tendency and spread of the maximal Cauchy stress under input uncertainty.

The summary statistics in Table 2 confirm this picture, with a mean of 5312.27 kPa, a median of 5256.46 kPa, and a standard deviation of 610.02 kPa. The 2.5% and 97.5% quantiles, 4267.02 kPa and 6648.41 kPa, match the visually identified 95% interval in Figure 3, while the interquartile range from 4876.66 kPa to 5688.45 kPa shows that most realizations lie in the central part of the distribution.

4.3 Findings

For a prestressed membrane, the maximal membrane Cauchy stress must satisfy $\sigma_{\text{mem,max}} > 0$, so only positive values are physically relevant. The empirical distribution of $\sigma_{\text{mem,max}}$ obtained from the GPR-based Monte Carlo simulation is clearly right-skewed. It is therefore reasonable to model this marginal response with a continuous distribution supported on positive values. In this analysis, the lognormal and gamma distributions are used as candidate models for $\sigma_{\text{mem,max}}$, since they are defined on $(0, \infty)$ and can flexibly represent right-skewed behavior.

A histogram of the 10,000 GPR predictions is plotted together with the fitted lognormal and gamma probability density functions (Figure 4). Both fitted curves follow the empirical distribution closely, capturing the right-skewness and overall spread, which suggests that each distribution provides a reasonable parametric description of the response

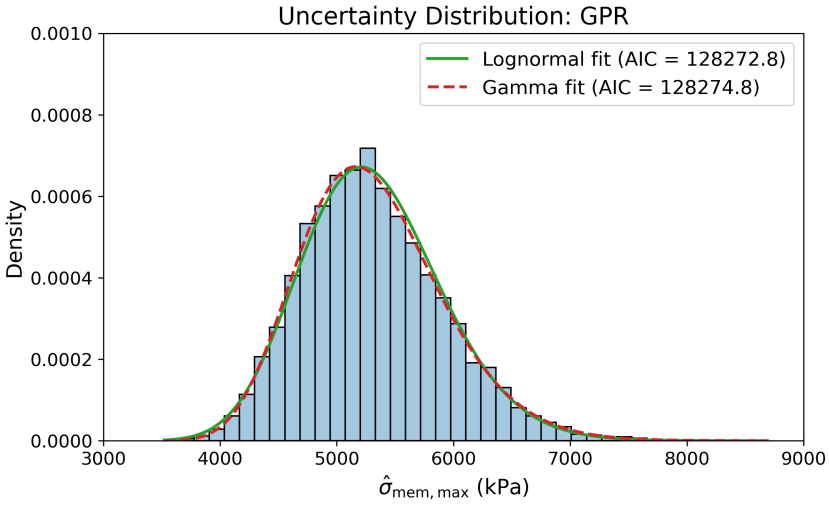

Figure 4: Empirical distribution of $\sigma_{\text{mem,max}}$ from GPR predictions with overlaid lognormal and gamma PDFs

variability.

To assess the fit more formally, the Akaike Information Criterion (AIC) is computed for both models, giving $\text{AIC}_{\text{lognormal}} = 128272.81$ and $\text{AIC}_{\text{gamma}} = 128274.81$, so that $\Delta\text{AIC} = \text{AIC}_{\text{gamma}} - \text{AIC}_{\text{lognormal}} \approx 2$. This small positive difference indicates a slight preference for the lognormal model, while at the same time confirming that both parametric families fit the data comparably well. It should be emphasized, however, that good visual agreement and similar AIC values do not prove that the samples are truly generated by either distribution; rather, they support the plausibility of using lognormal or gamma laws as convenient parametric approximations for the marginal behavior of $\sigma_{\text{mem,max}}$.

5 Conclusion

This project developed and evaluated surrogate models to approximate the maximal membrane Cauchy stress in a prestressed sun-sail structure under uncertain material and loading conditions. Using an optimized Latin hypercube design, 200 high-fidelity simulations with *Kratos Multiphysics* were carried out to generate a synthetic, effectively noise-free dataset that captures the main sources of variability in the structural response.

Among the three surrogate models, i.e., Support Vector Regression (SVR), Polynomial Regression (PLY), and Gaussian Process Regression (GPR), the GPR exhibited the best overall performance. Residual plots showed small, well-centered errors for GPR, and the mean squared error on both training and test sets was substantially lower than for SVR and PLY, indicating that GPR provides the most accurate and robust approximation of the finite element model within the available data budget.

Building on the GPR surrogate, Monte Carlo simulations were performed to propagate

input uncertainty and to study the distribution of the maximal membrane Cauchy stress. The resulting response distribution was strictly positive and right-skewed and was well represented by parametric models such as the lognormal and gamma distributions, with a slight preference for the lognormal law as a convenient approximation.

Overall, the results demonstrate that combining optimized sampling, Gaussian process surrogates, and Monte Carlo simulation offers an efficient and reliable framework for uncertainty quantification in membrane structures, while drastically reducing computational effort compared with direct finite element models. Future work could extend this framework by incorporating correlated input variables, adaptive or sequential sampling strategies, and more advanced surrogate formulations to further improve accuracy in critical regions of the input space.

6 Appendix

6.1 Input parameters for the structural membrane system

Table 1: Model input parameters for the structural membrane system.

Part	Quantity	Symbol	Unit	Distribution	Mean	Std. dev.
Membrane	Young's modulus	E_{mem}	[GPa]	Lognormal	0.60	0.09
Membrane	Poisson's ratio	ν_{mem}	[–]	Uniform	0.40	0.0115
Membrane	Thickness	t_{mem}	[mm]	Deterministic	1.00	–
Membrane	Pre-stress	σ_{mem}	[MPa]	Lognormal	4.00	0.80
Membrane	Surface loading	f_{mem}	[kPa]	Gumbel	0.40	0.12
Membrane	Rupture stress[1]	$\sigma_{\text{mem},y}$	[MPa]	Lognormal	11.00	1.65
Truss	Young's modulus	E_{tru}	[GPa]	Deterministic	205.00	–
Truss	Cross-sectional area	A_{tru}	[cm^2]	Deterministic	25.00	–
Edge cable	Young's modulus	E_{edg}	[GPa]	Deterministic	205.00	–
Edge cable	Diameter	d_{edg}	[mm]	Deterministic	12.00	–
Edge cable	Pre-stress	σ_{edg}	[MPa]	Lognormal	353.678	70.735
Support cable	Young's modulus	E_{sup}	[GPa]	Deterministic	205.00	–
Support cable	Diameter	d_{sup}	[mm]	Deterministic	12.00	–
Support cable	Pre-stress	σ_{sup}	[MPa]	Lognormal	400.834	80.166

[1] Used only for failure assessment; not included as a model parameter.

6.2 Summary statistics of Maximal cauchy stress

Table 2: Summary statistics of maximal Cauchy stress, $\hat{\sigma}_{\text{mem,max}}$

Statistic	Value
Mean	5312.27
Std. dev.	610.02
Min	3520.99
2.5% quantile	4267.02
25% quantile	4876.66
Median	5256.46
75% quantile	5688.45
97.5% quantile	6648.41
Max	8689.61

References

Liefvendahl, M., & Stocki, R. [2005]. A study on algorithms for optimization of latin hypercubes. *Journal of Statistical Planning and Inference*, *134*[1], 279–298. https://doi.org/10.1016/j.jspi.2005.01.007

Rasmussen, C. E., & Williams, C. K. I. [2006]. *Gaussian processes for machine learning*. The MIT Press. https://doi.org/10.7551/mitpress/3206.001.0001

Chang, Y.-W., Hsieh, C.-J., Chang, K.-W., Ringgaard, M., & Lin, C.-J. [2010]. Training and testing low-degree polynomial data mappings via linear svm. *Journal of Machine Learning Research*, *11*, 1471–1490.

Awad, M., & Khanna, R. [2015]. *Efficient learning machines: Theories, concepts, and applications for engineers and system designers* 1st ed. Apress Berkeley, CA. https://doi.org/10.1007/978-1-4302-5990-9

Sudret, B., Marelli, S., & Wiart, J. [2017]. Surrogate models for uncertainty quantification: An overview. *2017 11th European Conference on Antennas and Propagation (EUCAP)*, 793–797. https://doi.org/10.23919/EuCAP.2017.7928679

Mataix Ferrándiz, V., Bucher, P., Zorrilla, R., Rossi, R., Cornejo, A., Celigueta, M. A., Roig, C., Masó, M., Warnakulasuriya, S., Casas, G., Nuñez, M., Dadvand, P., Latorre, S., de Pouplana, I., Irazábal González, J., Arrufat, F., Chandra, B., Geiser, A., Sautter, K. B., ... Garáte, J. [2023]. *Kratosmultiphysics/kratos: Release 9.3* Version 9.3. Zenodo. https://doi.org/10.5281/zenodo.7681287

Brenndoerfer, M. [2025]. Polynomial regression: Complete guide with math, implementation best practices [Accessed: 2026-01-09]. *mbrenndoerfer.com*. https://mbrenndoerfer.com / writing / polynomial - regression - complete - guide - math - implementation - python-scikit-learn

Python Software Foundation. [2025]. *Python 3.12 documentation* [Last updated on Oct 10, 2025]. Python Software Foundation. https://docs.python.org/3.12/