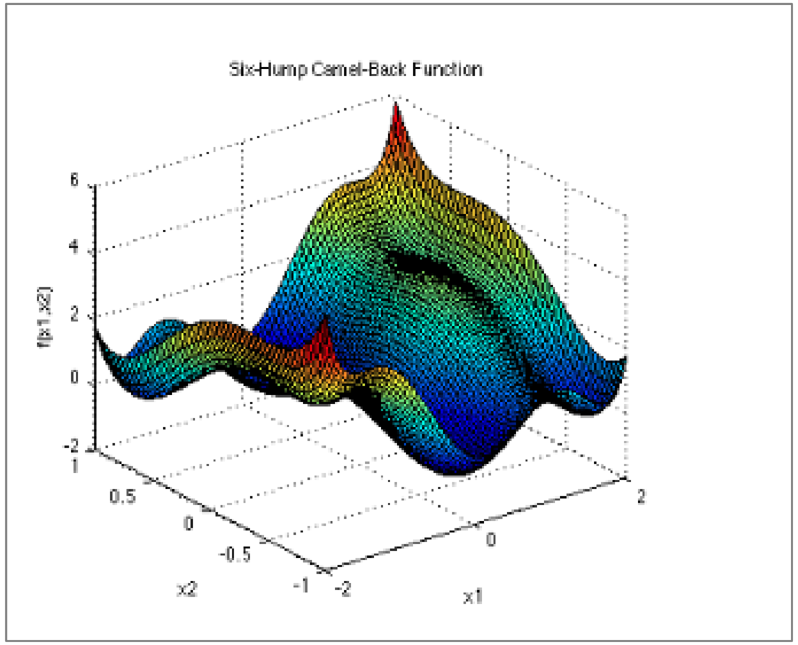
Workshop – Prediction Analysis, Introduction to Gaussian Process Regression

AN MSC NASTRAN MACHINE LEARNING WEB APP TUTORIAL

Questions? Email: christian@ the-engineering-lab.com

Goal: Use Gaussian process regression to predict the true function

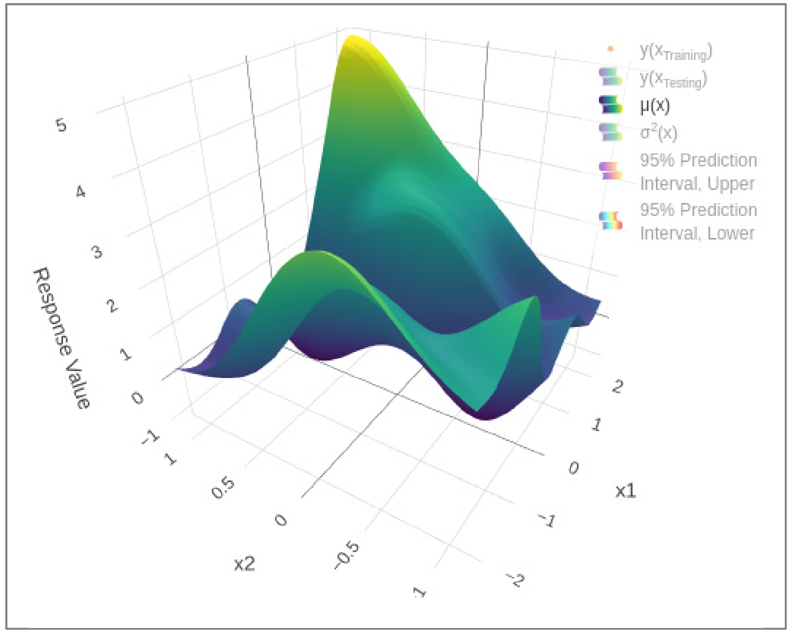
True Function



$$f(\mathbf{x}) = \left(4 - 2.1x_1^2 + \frac{x_1^4}{3}\right)x_1^2 + x_1x_2 + (-4 + 4x_2^2)x_2^2$$

Source: https://www.sfu.ca/~ssurjano/camel6.html

Predicted Function



Contact me

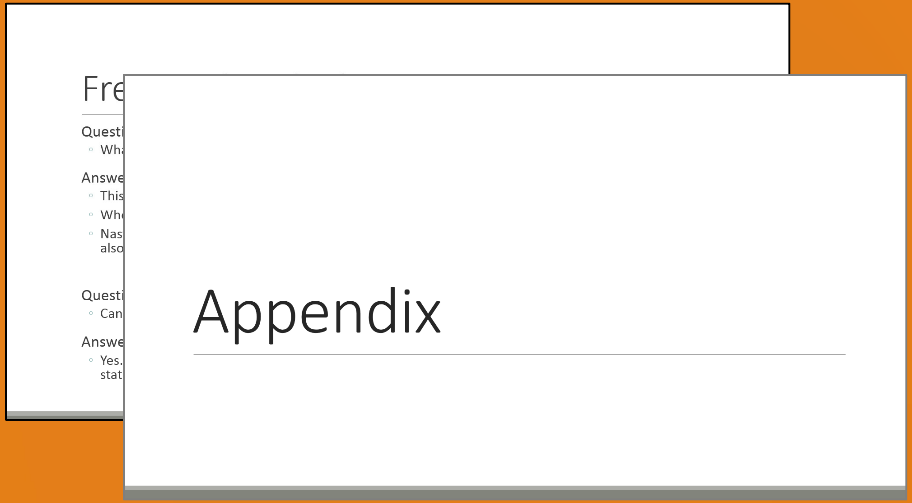
- Nastran SOL 200 training
- Nastran SOL 200 questions
- Structural or mechanical optimization questions
- Access to the SOL 200 Web App

christian@ the-engineering-lab.com

More Information Available in the Appendix

The Appendix includes information regarding the following:

- What is Gaussian Process Regression?



Tutorial

Tutorial Overview

Use the Prediction Analysis web app to:

1. Part 1
    1. Perform a regression
    2. Perform a Prediction
    3. View the response surface of the surrogate model
2. Part 2
    1. Import training data from a structural analysis
    2. Perform a regression
    3. View the response surface of the surrogate model

Special Topics Covered

Gaussian process (GP) regression – This tutorial introduces users to GP regression. Three different kernels, alternatively called covariance functions, are used. The three kernels are Matern52, Exponential and RBF.

The Prediction Analysis web app uses the multivariate normal (MVN) conditioning equations to calculate the mean and variance functions. The mean function, or surrogate model, is used to make predictions and the variance function is used to gauge the uncertainty of the predicted values. The MVN conditioning equations have multiple names including: MVN conditioning identities, Gaussian process regression, kriging and kriging equations.

mean	$\mu(\mathcal{X}) = \Sigma(\mathcal{X}, X_n)\Sigma_n^{-1}Y_n$	Predicted values
variance	$\Sigma(\mathcal{X}) = \Sigma(\mathcal{X}, \mathcal{X}) - \Sigma(\mathcal{X}, X_n)\Sigma_n^{-1}\Sigma(\mathcal{X}, X_n)^\top$	Prediction Uncertainty

SOL 200 Web App Capabilities

The Post-processor Web App and HDF5 Explorer are free to MSC Nastran users.

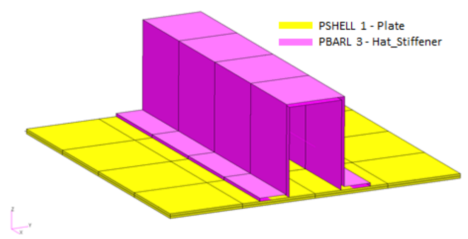
Compatibility

- Google Chrome, Mozilla Firefox or Microsoft Edge
- Windows and Red Hat Linux
- Installable on a company laptop, workstation or server. All data remains within your company.

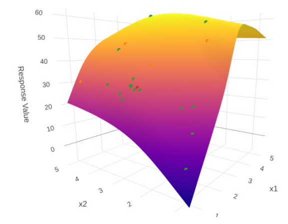
Benefits

- REAL TIME error detection. 200+ error validations.
- REALT TIME creation of bulk data entries.
- Web browser accessible
- Free Post-processor web apps
- +80 tutorials

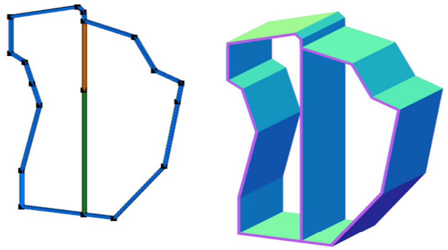
Web Apps



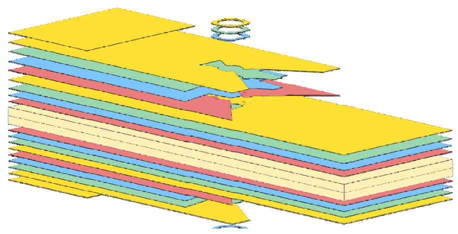
Web Apps for MSC Nastran SOL 200
Pre/post for MSC Nastran SOL 200. Support for size, topology, topometry, topography, multi-model optimization.



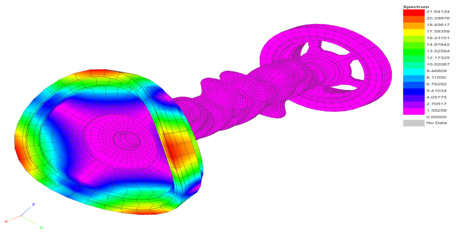
Machine Learning Web App
Bayesian Optimization for nonlinear response optimization (SOL 400)



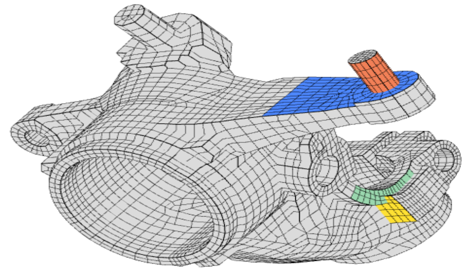
PBMSECT Web App
Generate PBMSECT and PBRSECT entries graphically



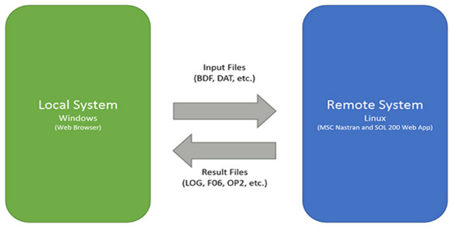
Ply Shape Optimization Web App
Optimize composite ply drop-off locations, and generate new PCOMPG entries



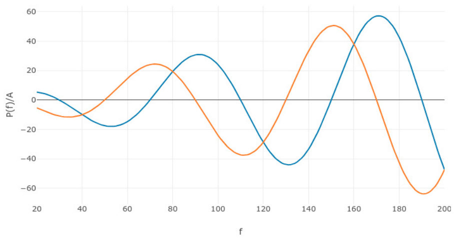
Post-processor Web App
View MSC Nastran results in a web browser on Windows and Linux



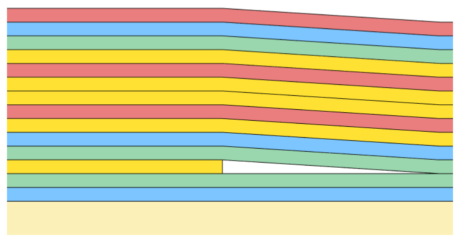
Shape Optimization Web App
Use a web application to configure and perform shape optimization.



Remote Execution Web App
Run MSC Nastran jobs on remote Linux or Windows systems available on the local network



Dynamic Loads Web App
Generate RLOAD1, RLOAD2 and DLOAD entries graphically



Stacking Sequence Web App
Optimize the stacking sequence of composite laminate plies

HDF5 Explorer Web App
Create graphs (XY plots) using data from the H5 file

Open the Correct Page

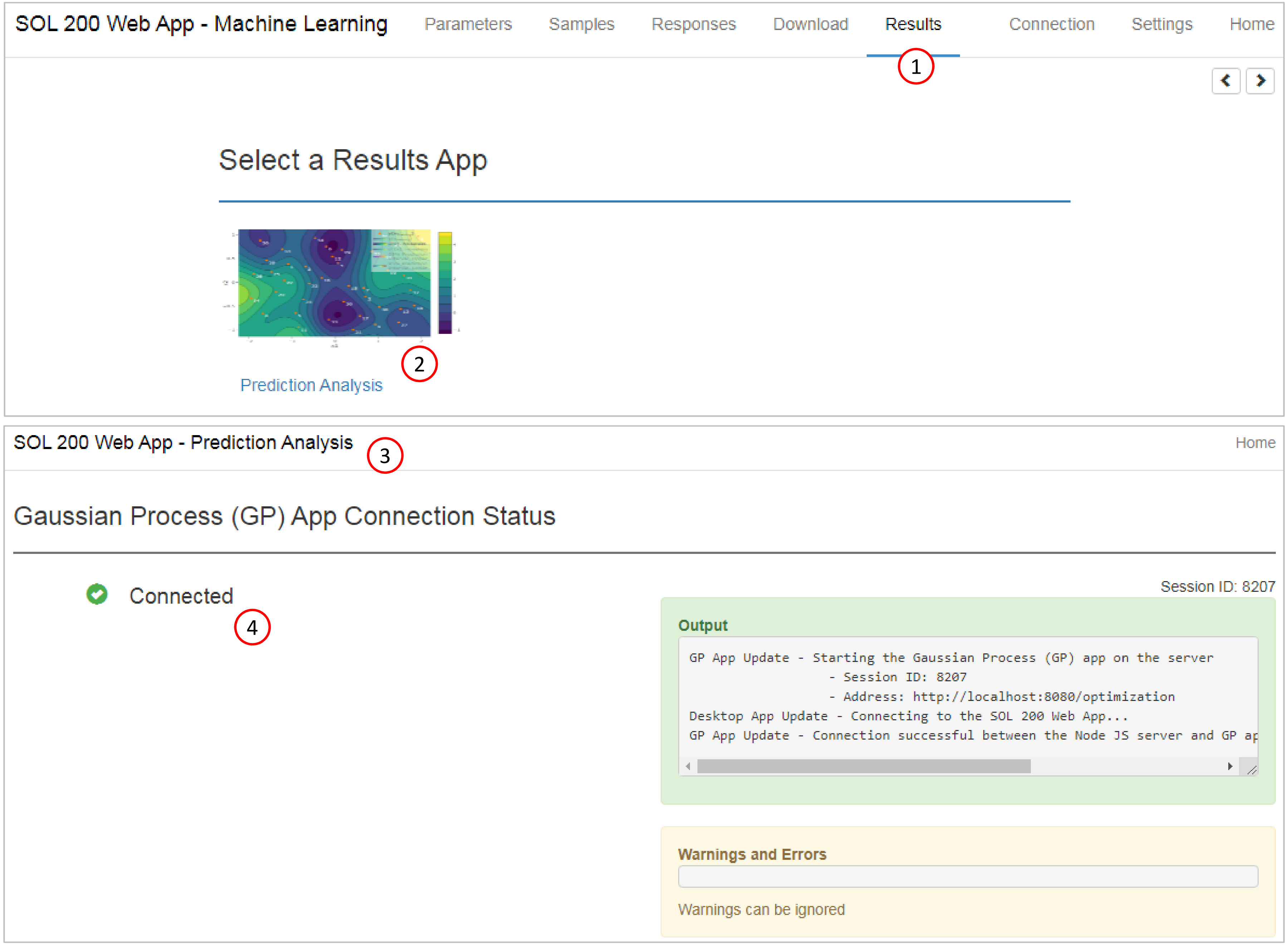
1. Click on the indicated link

- MSC Nastran can perform many optimization types. The SOL 200 Web App includes dedicated web apps for the following:
  - Optimization for SOL 200 (Size, Topology, Topometry, Topography, Local Optimization, Sensitivity Analysis and Global Optimization)
  - Multi Model Optimization
  - Machine Learning
- The web app also features the HDF5 Explorer, a web application to extract results from the H5 file type.

The Engineering Lab

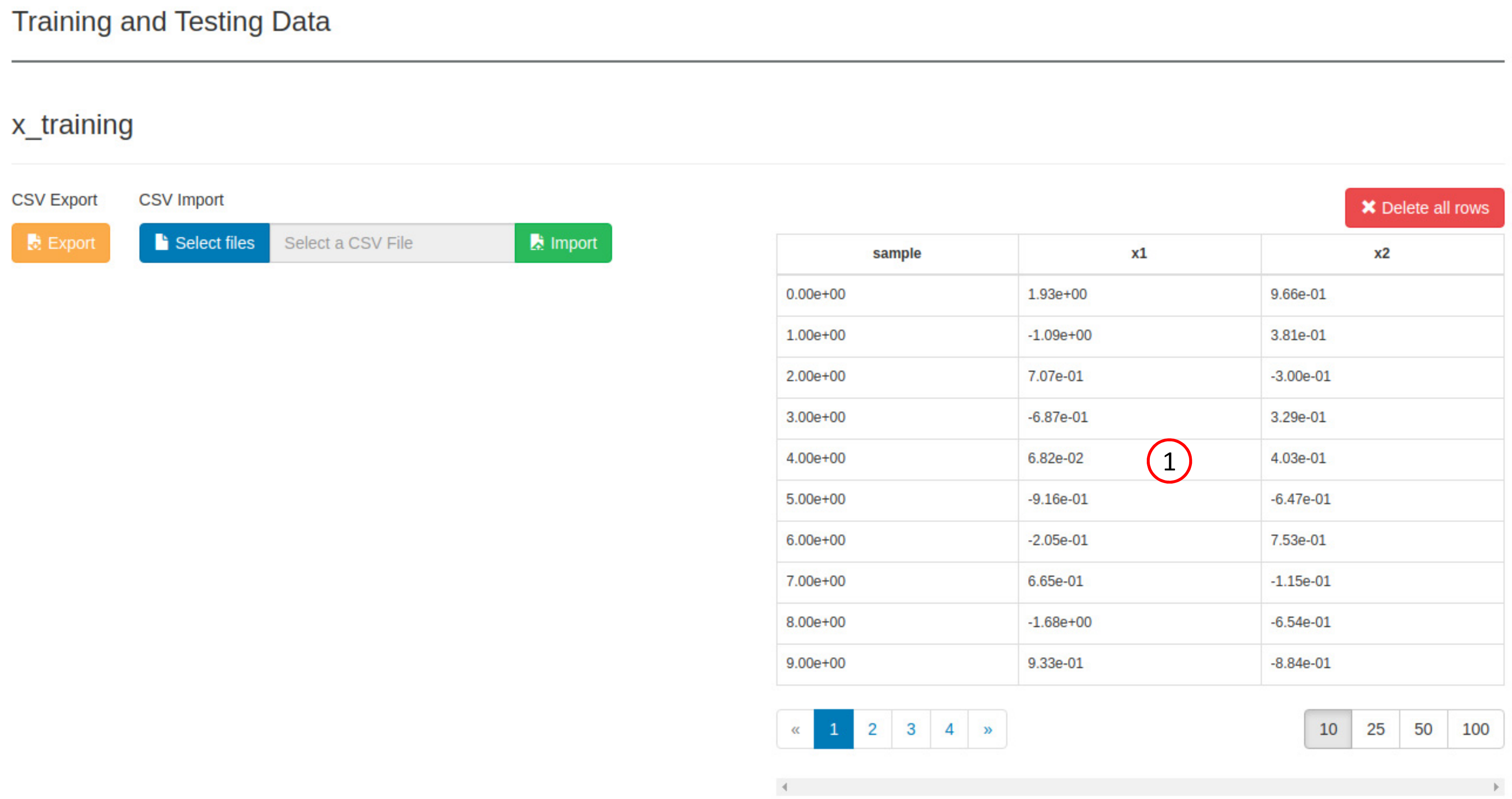
Open the Correct Page

1. Click Results
2. Click Prediction Analysis
3. You are navigated to the Prediction Analysis web app
4. Ensure it says Connected



Training and Testing Data

1. When the Prediction Analysis web app is first opened, training and testing data is already preloaded. For this part of the tutorial you do not have to upload training or testing data.

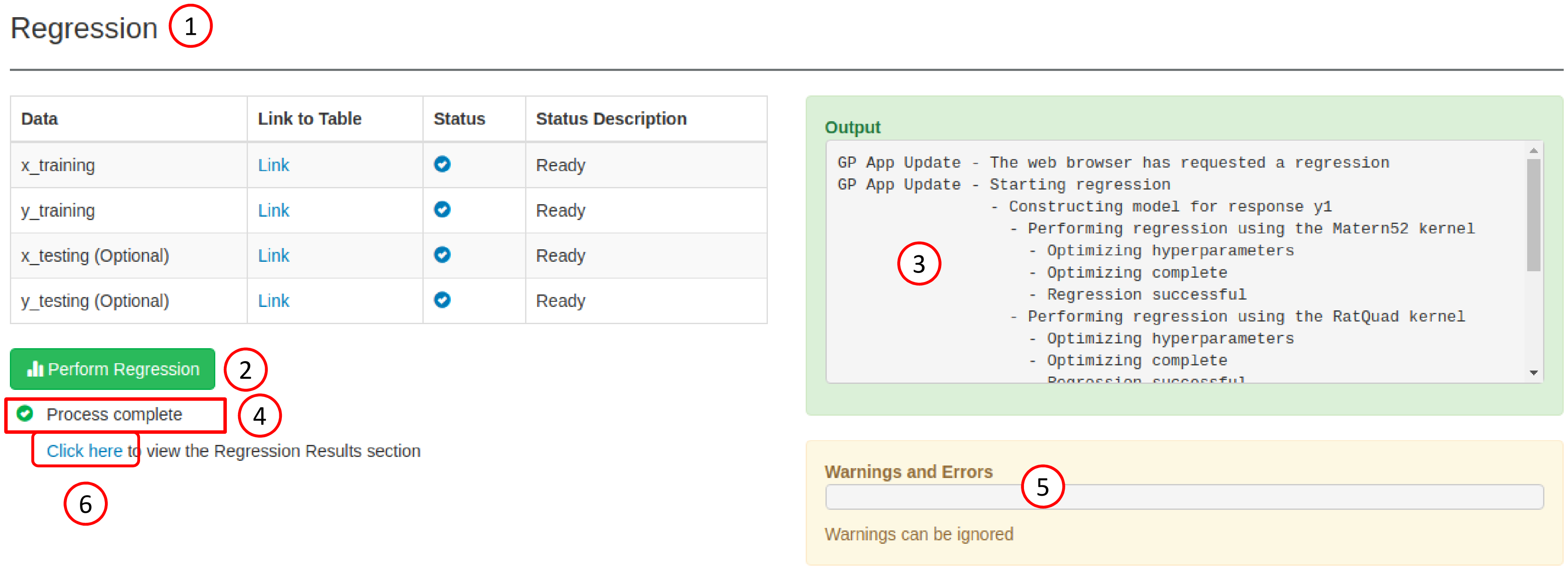


sample	x1	x2
0.00e+00	1.93e+00	9.66e-01
1.00e+00	-1.09e+00	3.81e-01
2.00e+00	7.07e-01	-3.00e-01
3.00e+00	-6.87e-01	3.29e-01
4.00e+00	6.82e-02	4.03e-01
5.00e+00	-9.16e-01	-6.47e-01
6.00e+00	-2.05e-01	7.53e-01
7.00e+00	6.65e-01	-1.15e-01
8.00e+00	-1.68e+00	-6.54e-01
9.00e+00	9.33e-01	-8.84e-01

- Training Data – The training data is a collection of inputs and outputs of a black box function. The training data is used in Gaussian process regression.
- Testing Data – The testing data is a collection of inputs and outputs of a black box function. The testing data is used to validate the predictions, i.e. the predictions are compared to the testing data's outputs.

Regression

1. Navigate to the Regression section
2. To start the regression, click Perform Regression
3. Output of the regression process is displayed
4. The regression is complete when the following status message is visible:
    - Process complete
5. Any warnings can be ignored
6. Click the indicated link to move to the Regression Results section



Parameter Relevance for Variable Screening

Automatic Relevance Determination (ARD) is one of many methods to screen parameters. After the regression, the ARD values are calculated and displayed in the output.

1. Scroll through the output until this section is visible: Summary of Automatic Relevance Determination (ARD)

2. A table is listed with the ARD values for each response with respect to each parameter. High values indicate the variable is relevant.

3. When there are multiple responses, the table will have multiple ARD values. Deciding which parameters are relevant across all responses is difficult. An additional output is available as shown and lists the parameters in decreasing order of relevance across all responses. Parameters listed at the start are more relevant than parameters towards the end.

Output

```
GP App Update - Summary of Automatic Relevance Determination (ARD)
                ARD measures predictive relevance of parameters and is used for parameter selection.
                - High Value: The parameter xi is relevant
                - Low Value:  The parameter xi is irrelevant and could potentially be screened out
Kernel: Matern52
| Response   |      x1 |      x2 |
|:-----------|--------:|--------:|
| y1         | 29.3195 | 9.51006 |
    Parameters listed in decreasing order of relevance: x1, x2
Kernel: Exponential
| Response   |      x1 |       x2 |
```

Regression Results

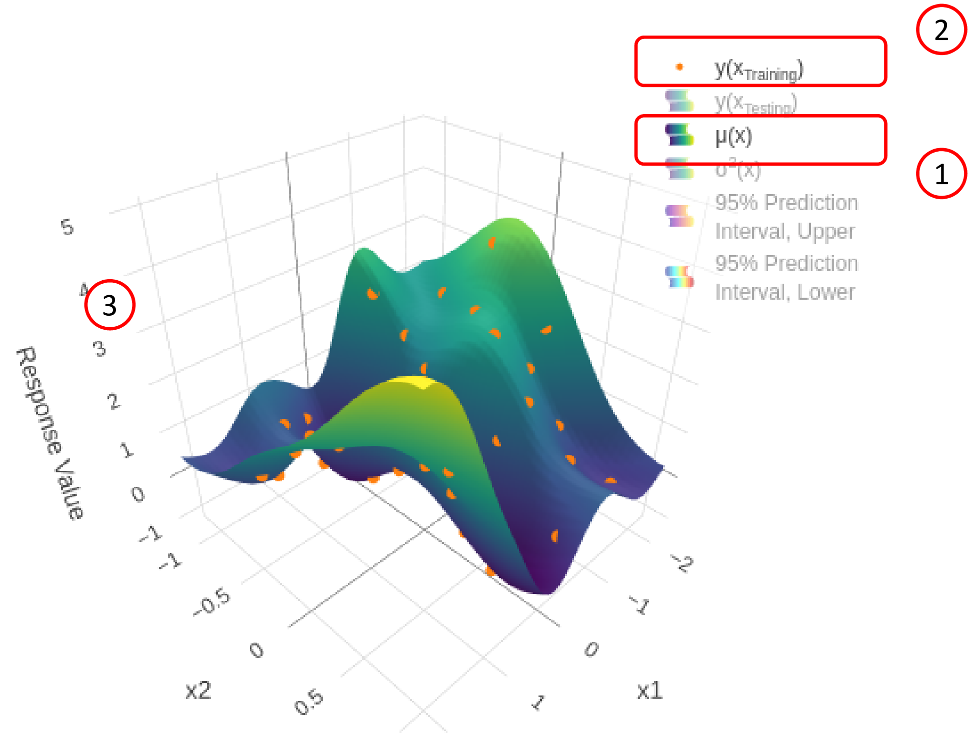
1. Double click μ(x)
2. Single click $y(x_{training})$
3. The predicted values (μ(x)) and training data are displayed. This plot is only visible if the training data has 1 or 2 parameters (variables).

Regression Results

Response Surface

Matern52

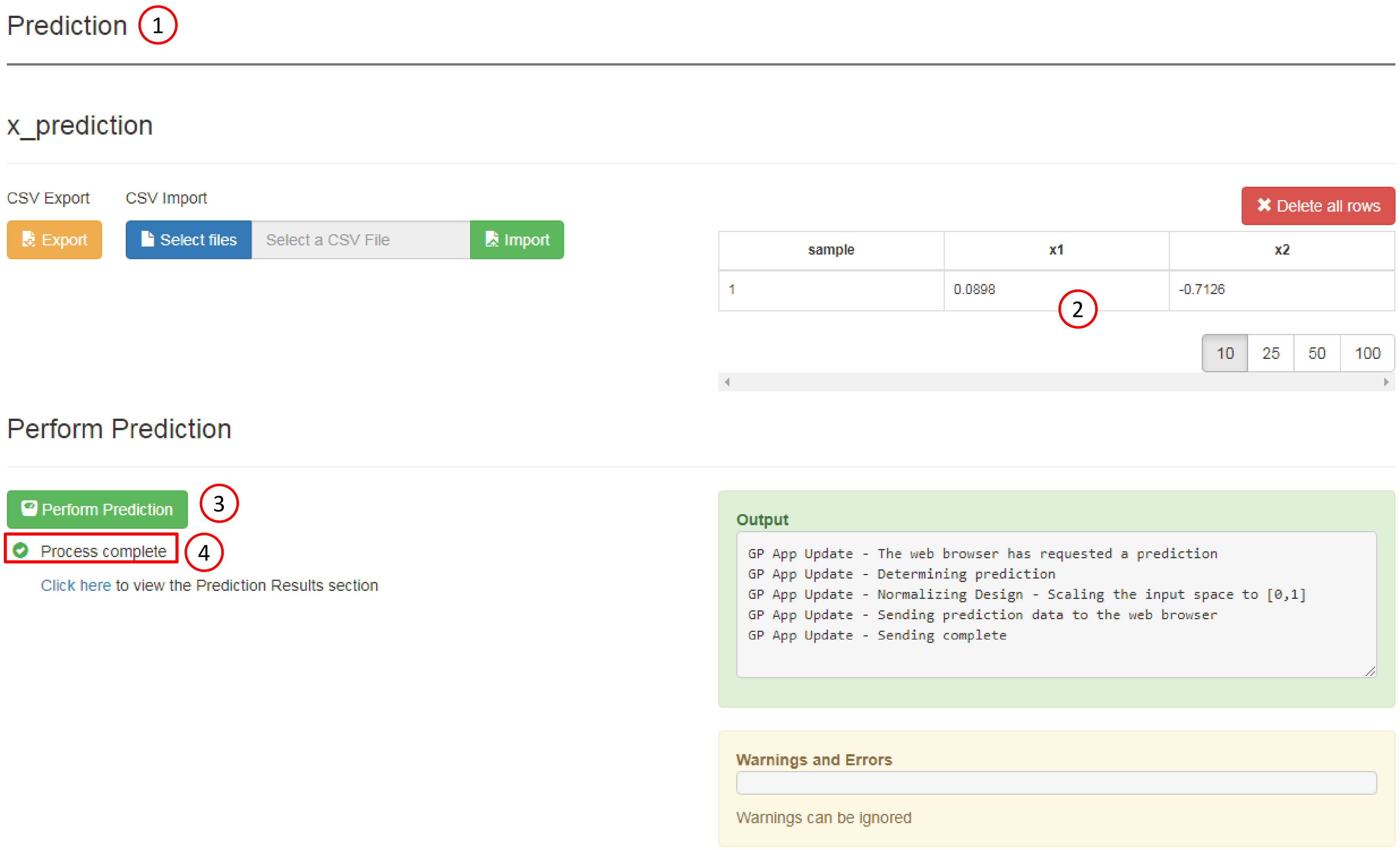
View Contour Plot



Prediction

This example used the Six-Hump Camel function to generate the training data. The global minimum exists at (.0898, -.7126) and (-.0898, .7126). A prediction is made at (.0898, -.7126).

1. Navigate to the Prediction section
2. Inputs for the parameters have been provided, a prediction will be made at this point
3. Click Perform Prediction
4. The prediction is complete when the following status message is visible:
    - Process complete



Response Surface

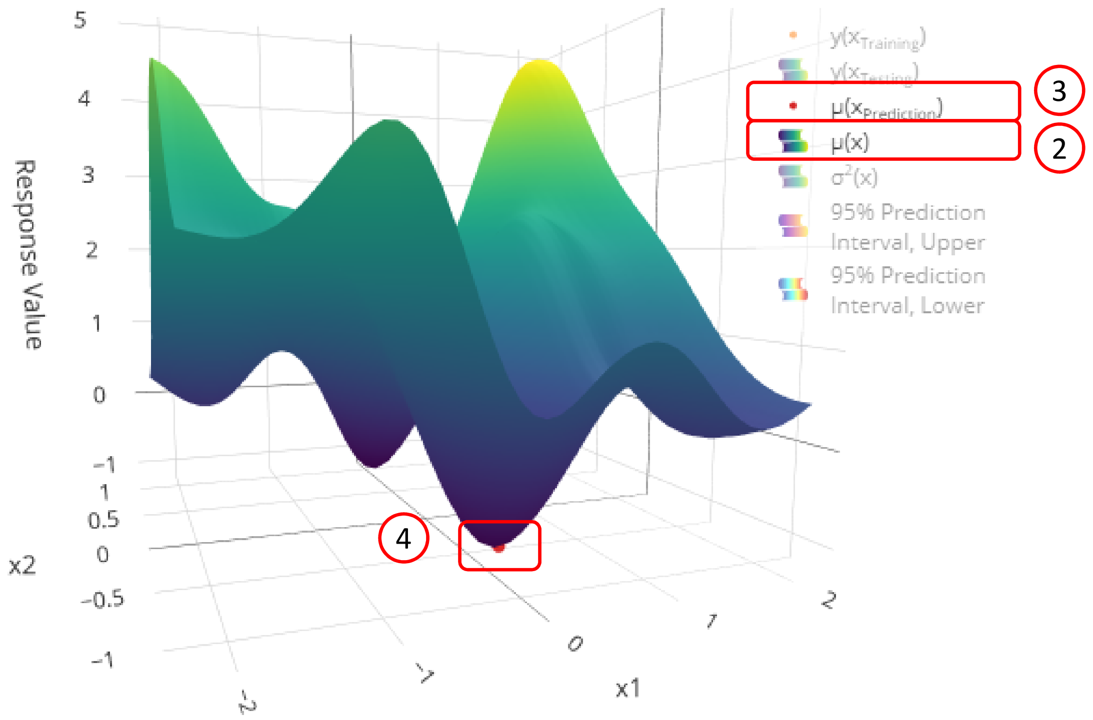
1. Navigate to the Regression Results section
2. Double click μ(x)
3. Single click $\mu(x_{Prediction})$
4. Rotate the model until the prediction is visible

Regression Results 1

Response Surface

Matern52

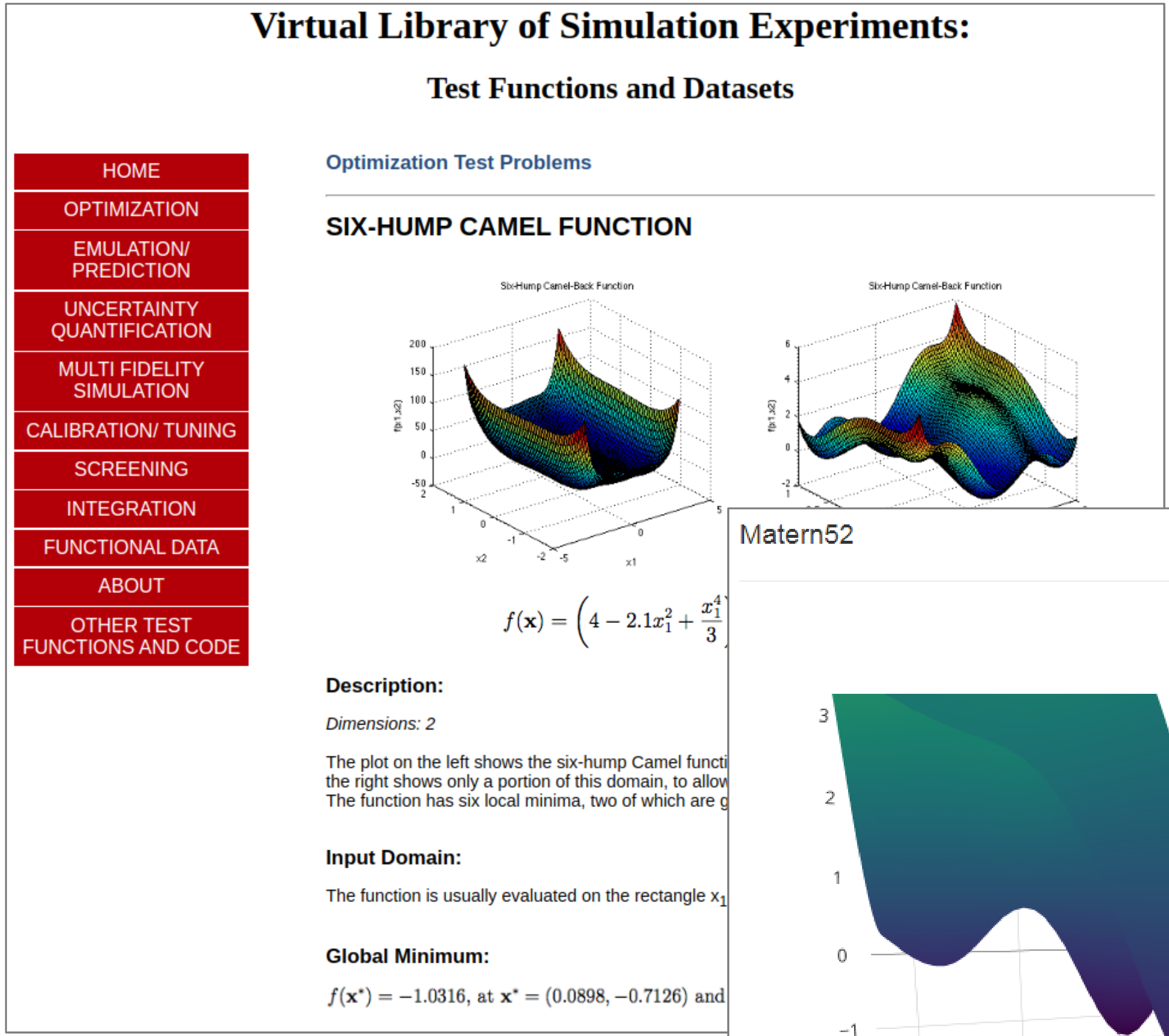
View Contour Plot



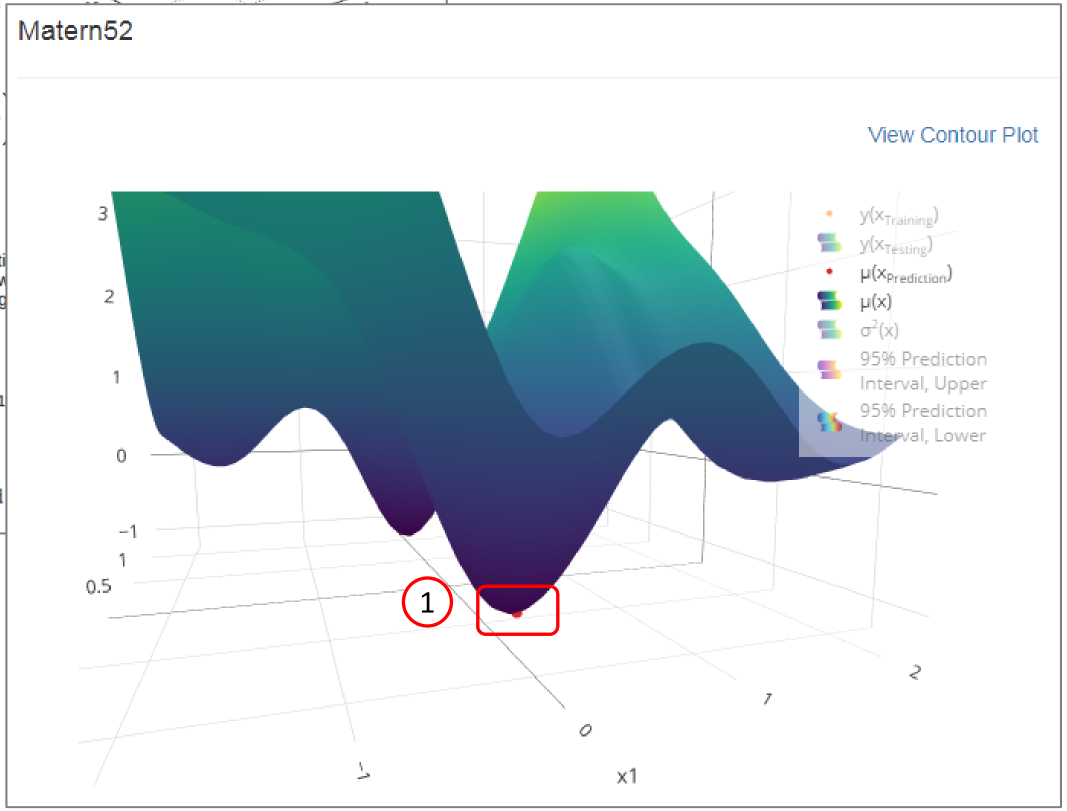
Comparison of True Function and Predicted Response Surface

This example used the Six-Hump Camel function to generate the training data. The global minimum exists at (.0898, -.7126) and (-.0898, .7126).

1. A prediction was made at (.0898, -.7126). The predicted response surface (μ(x)) does show a minimum close to this point. The predicted surface does well in capturing the behavior of the original Six-Hump Camel function.



Source: https://www.sfu.ca/~ssurjano/camel6.html



Part 2

Training and Testing Data

This part of the tutorial demonstrates how you can upload your own training and testing data.

Training and Testing Data

x_training

CSV Export

Export

CSV Import

Select files | Select a CSV File | Import

Delete all rows

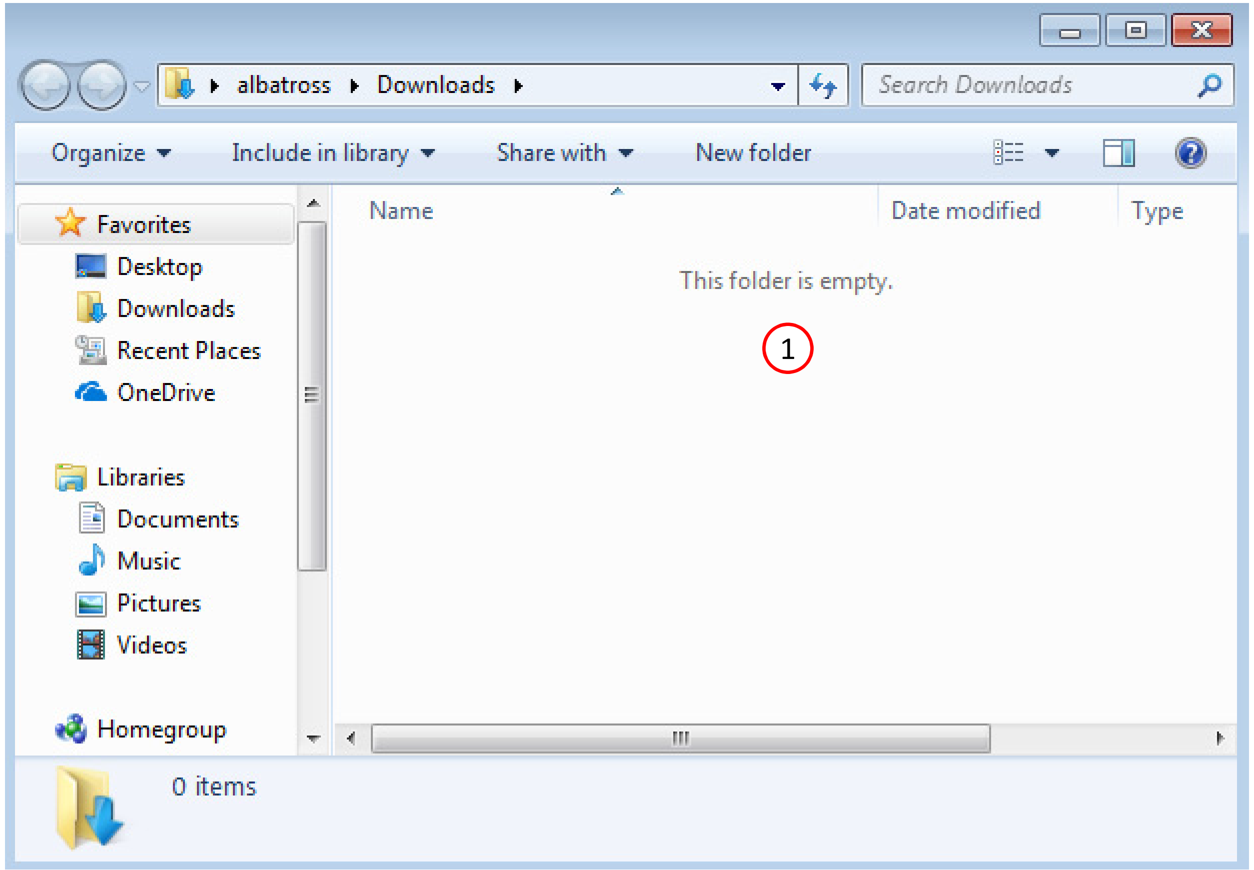
sample	x1	x2
0.00e+00	1.93e+00	9.66e-01
1.00e+00	-1.09e+00	3.81e-01
2.00e+00	7.07e-01	-3.00e-01
3.00e+00	-6.87e-01	3.29e-01
4.00e+00	6.82e-02	4.03e-01
5.00e+00	-9.16e-01	-6.47e-01
6.00e+00	-2.05e-01	7.53e-01
7.00e+00	6.65e-01	-1.15e-01
8.00e+00	-1.68e+00	-6.54e-01
9.00e+00	9.33e-01	-8.84e-01

« 1 2 3 4 »

10 25 50 100

Before Starting

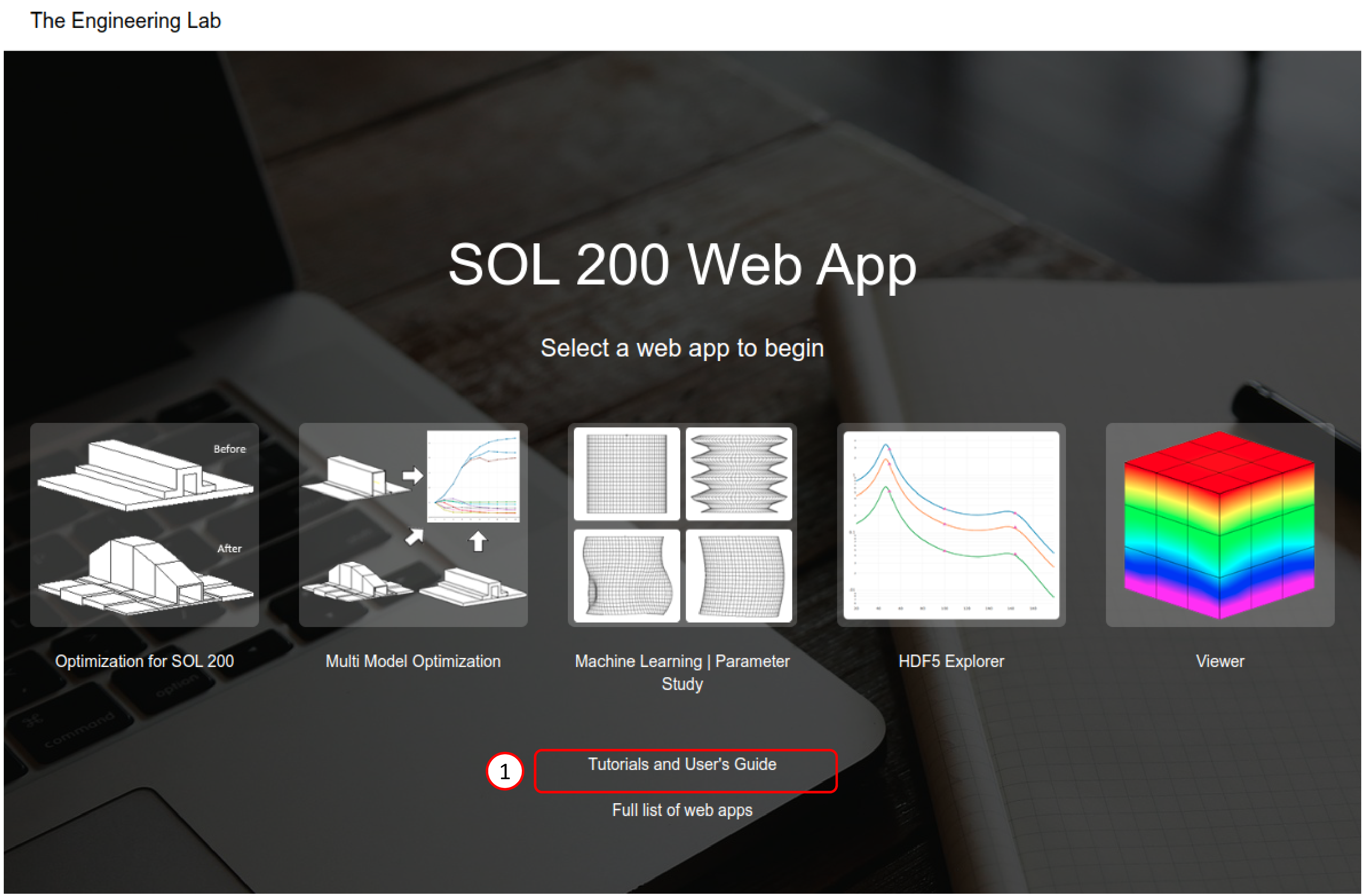
1. Ensure the Downloads directory is empty in order to prevent confusion with other files



Go to the User's Guide

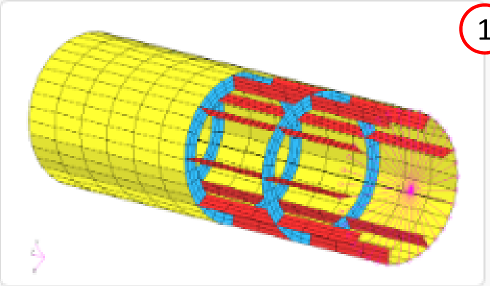
1. Click on the indicated link

- The necessary BDF files for this tutorial are available in the Tutorials section of the User's Guide.



Obtain Starting Files

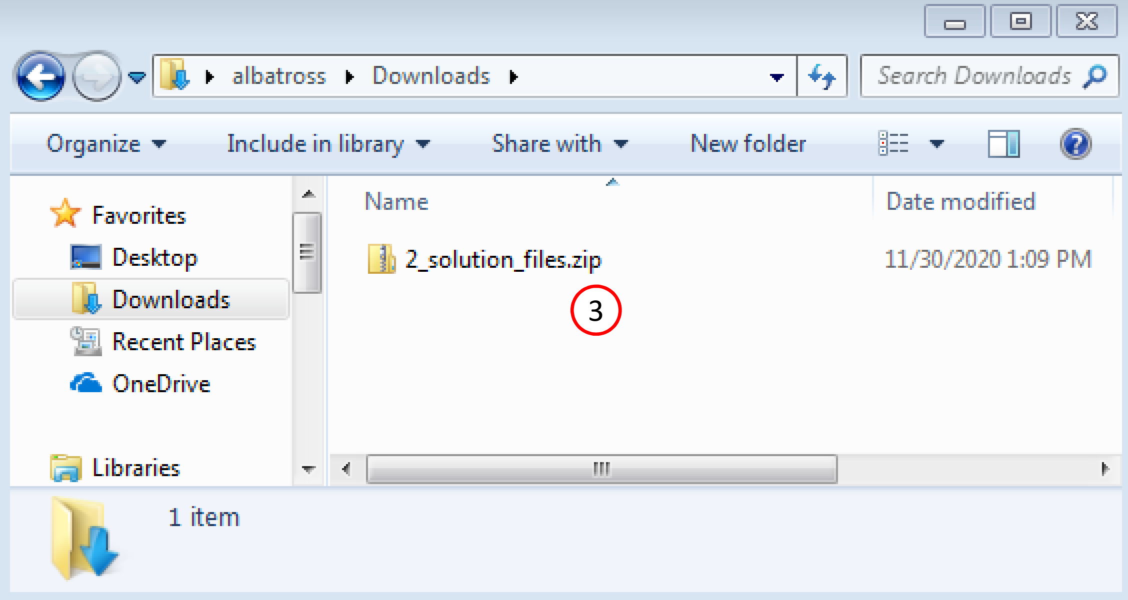
1. Find the indicated example
2. Click Link
3. The starting file has been downloaded



1 **Machine Learning, Nonlinear Buckling (Post-Buckling) Optimization of a Reinforced Cylinder**

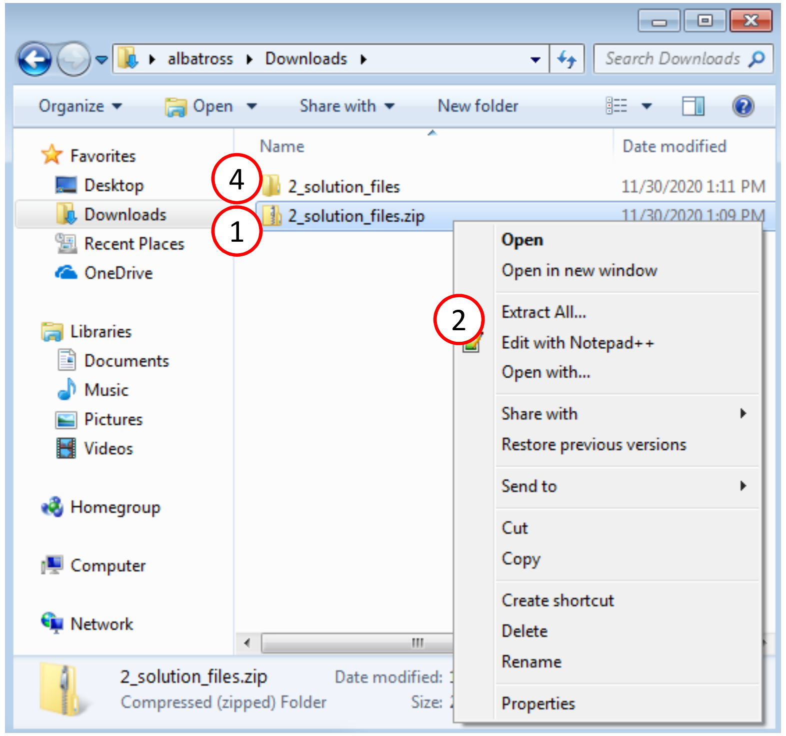
A reinforced cylinder is fixed at the base and a load is applied laterally at its top. Machine learning is performed to determine the optimal thicknesses of the reinforcements to achieve a minimum eigenvalue of 30 while minimizing the weight. This example features nonlinear buckling (post-buckling) analysis.

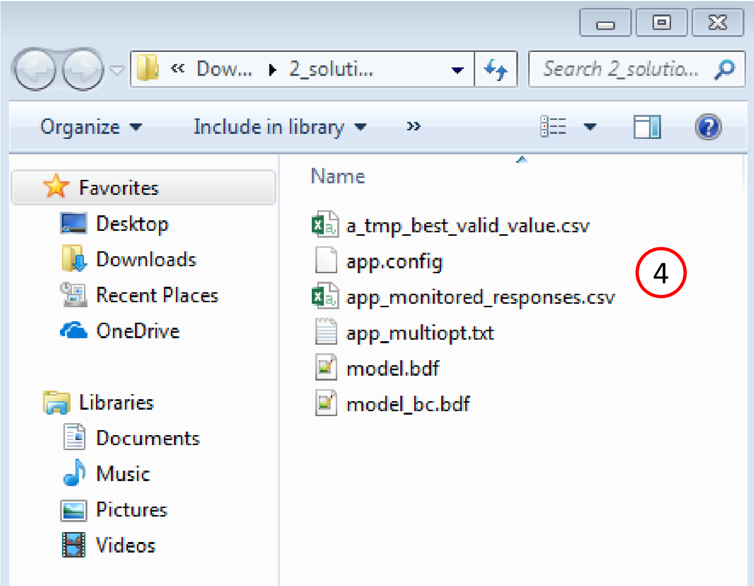
Starting Files: Link
Solution BDF Files: Link 2

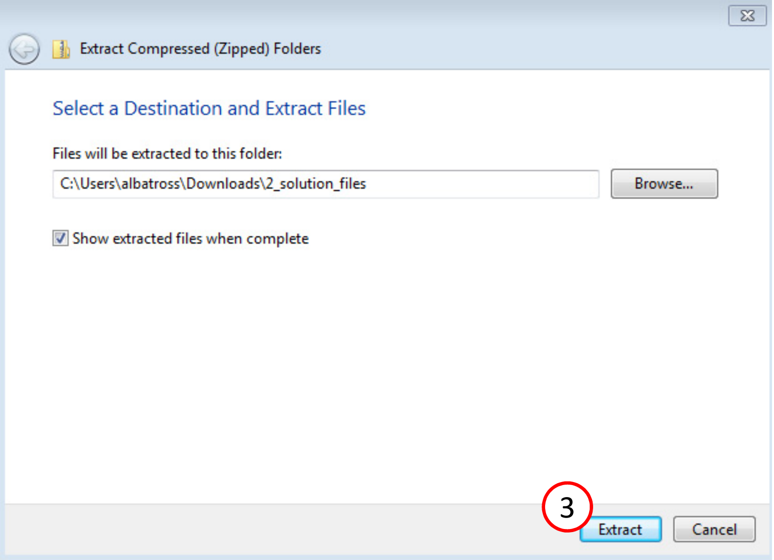


Obtain Starting Files

1. Right click on the zip file
2. Select Extract All…
3. Click Extract
4. The starting files are now available in a folder





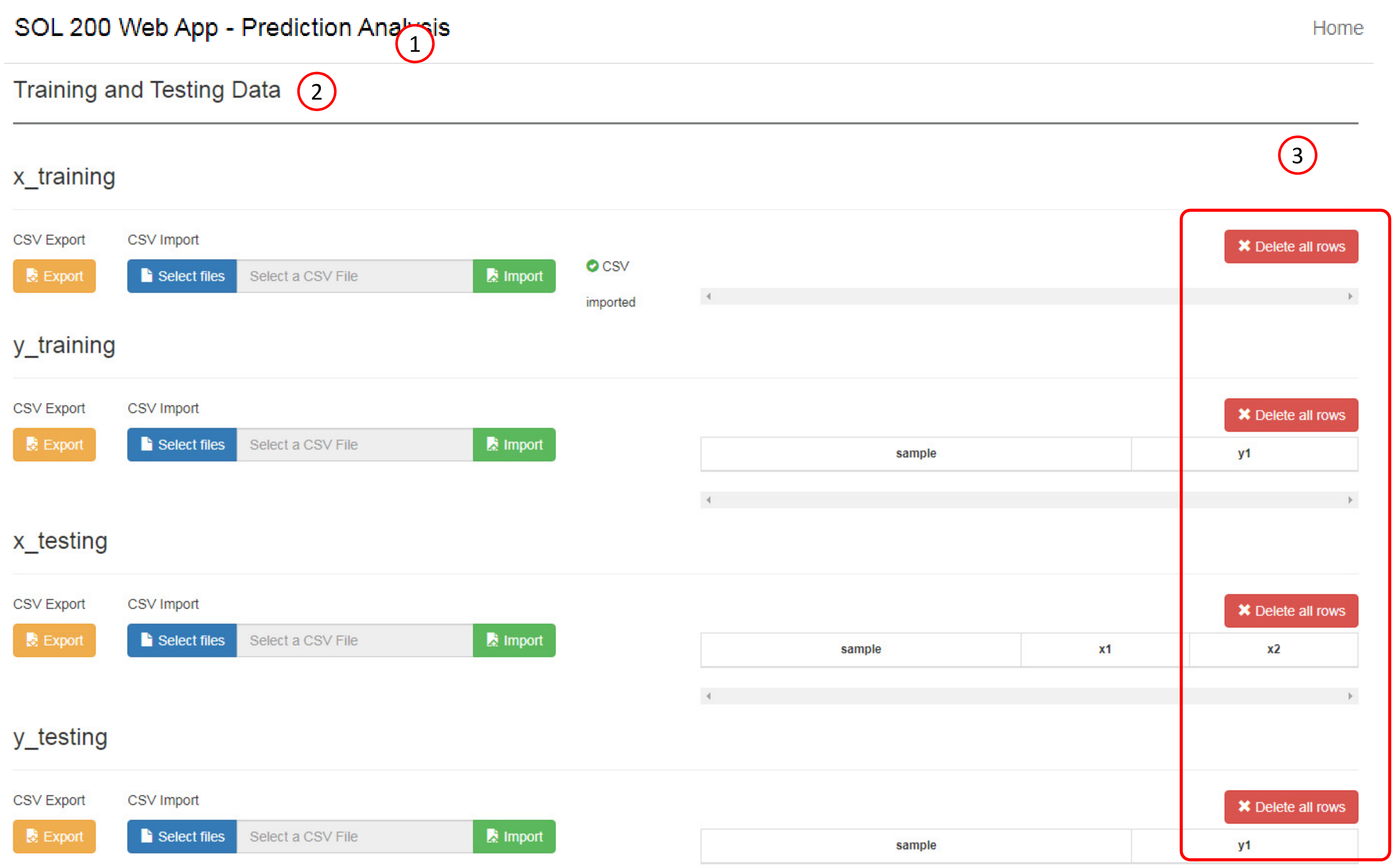


Training and Testing Data

The old training and testing data must be deleted

1. Return to the Prediction Analysis web app
2. Navigate to the Training and Testing Data section
3. Delete any previous table data by clicking the four (4) buttons named Delete all rows

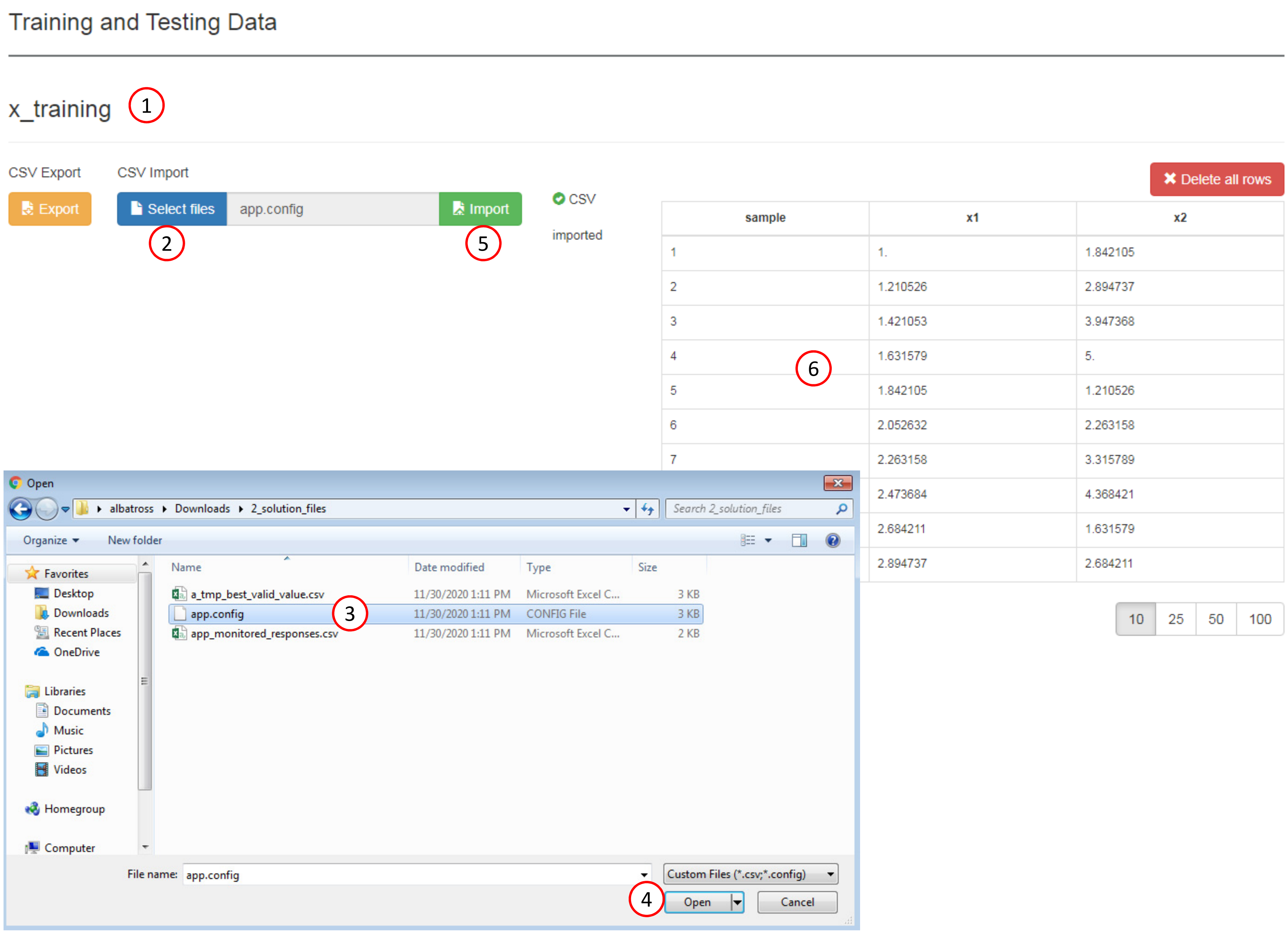
- **x_training, y_training** - This specifies the x inputs and y outputs used to train the surrogate model.
- **x_testing, y_testing** - This specifies the x inputs and y outputs used to calculate the Normalized Root Mean Square Error (NRMSE) between the predicted values and actual MSC Nastran responses. This testing data is optional.
- **x_prediction** – The x inputs at which to make predictions.



Training and Testing Data

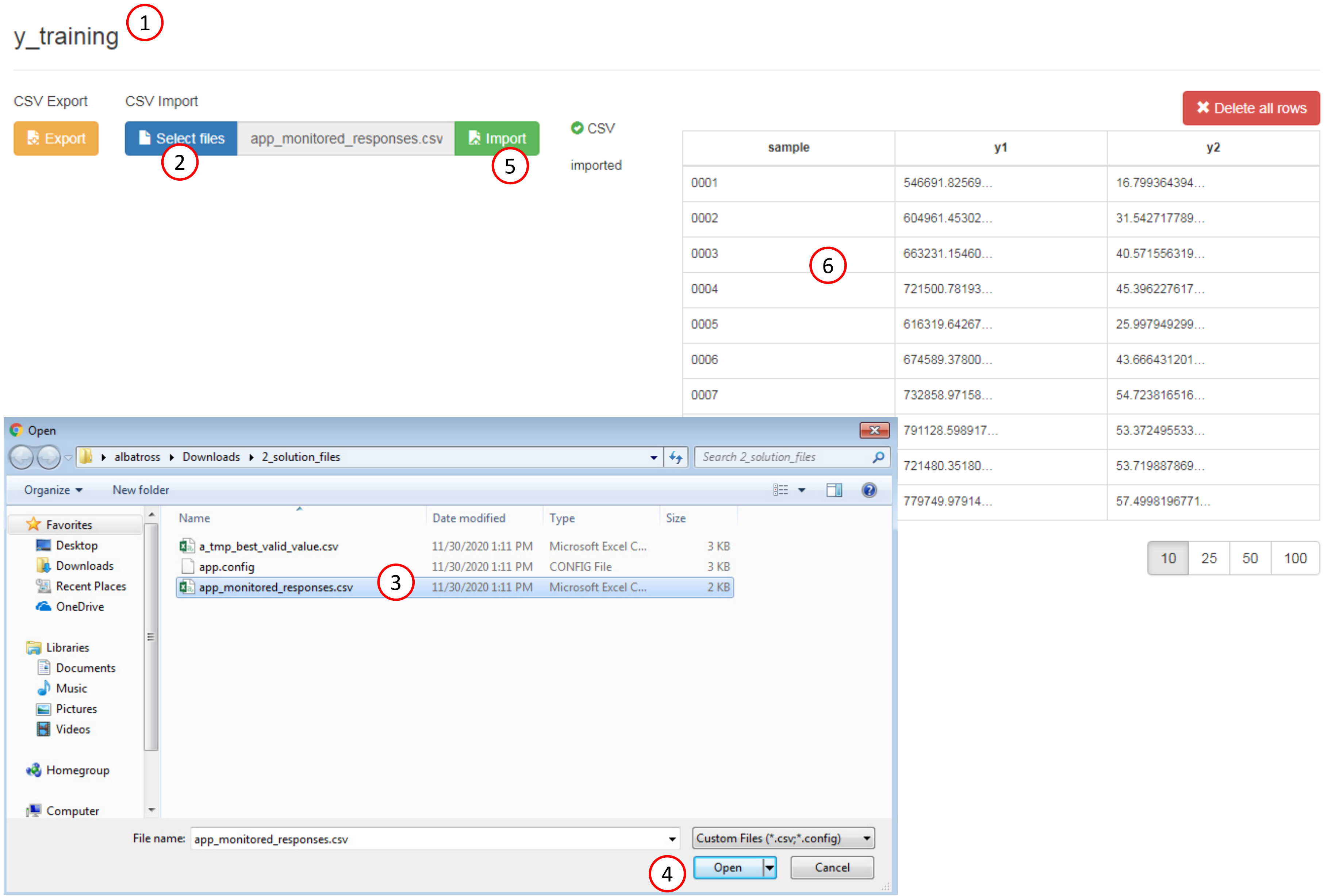
New training data can be imported as follows

1. Navigate to the section titled x_training
2. Click Select Files
3. Select the file titled app.config
4. Click Open
5. Click Import
6. The x inputs of the training data are now imported



Training and Testing Data

1. Ensure you are at the y_training section
2. Click Select Files
3. Select the file titled app_monitored_responses.csv
4. Click Open
5. Click Import
6. The observed values, or monitored responses, of the training data are now imported

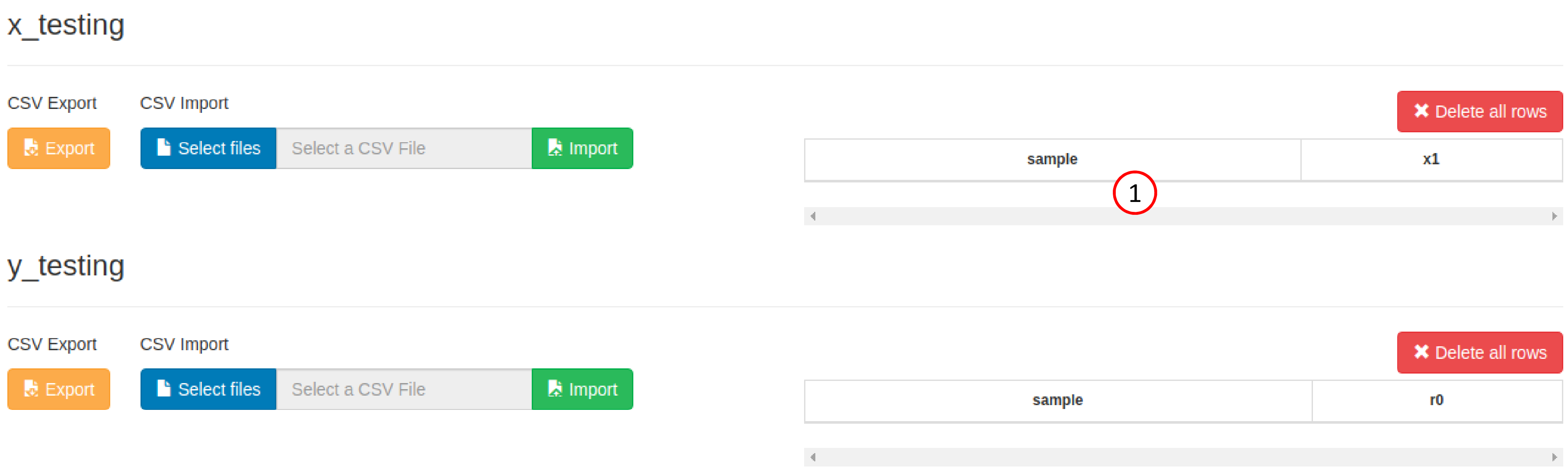


Training and Testing Data

1. No data is imported for the testing data sections.

<u>The testing data is optional.</u> The testing data is used to test the predictions, i.e. the observed responses of the testing data are compared to the predictions. The procedure is generally as follows:

- A second parameter study is performed at different x inputs.
- The results of the second parameter study (app.config and app_monitored_responses.csv) is imported to the testing section.
- The web app computes the normalized root mean square (NRMSE) value.



Regression

1. After the training data has been imported, a regression and prediction can be performed.

SOL 200 Web App - Prediction Analysis

Home

Regression

Data	Link to Table	Status	Status Description
x_training	Link		Ready
y_training	Link		Ready
x_testing (Optional)	Link		Ready
y_testing (Optional)	Link		Ready

Perform Regression 1

Process complete

Click here to view the Regression Results section

Output

```
GP App Update - The web browser has requested a regression
GP App Update - Normalizing Design - Scaling and shifting the input space to
GP App Update - Starting regression
                - Constructing model for response y1
                  - Performing regression using the Matern52 kernel
                    - Optimizing model hyperparameters
                    - Optimizing complete

                     Log Marginal Likelihood: -22.641525
```

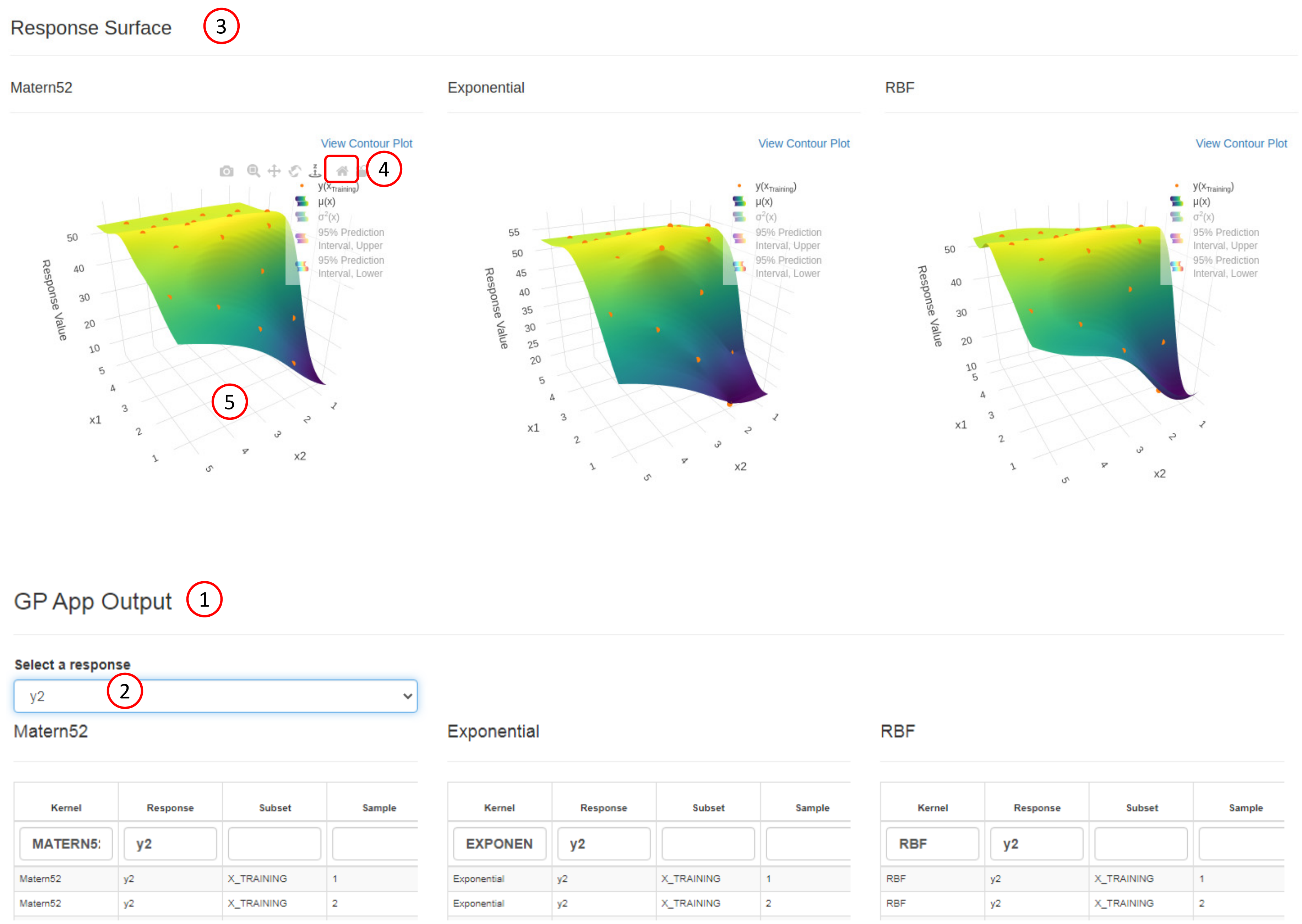
Warnings and Errors

Warnings can be ignored

Response Surface

1. Navigate to the section titled GP App Output
2. Select response y2
3. Navigate to the section title Response Surface
4. Click the Home icon to fit the plot in the view
5. Rotate the model by holding down the left mouse button and moving the mouse

- If the model consists of 1 or 2 parameters, a response surface will be displayed. For models with 3 or more parameters, the response surface is not displayed.



End of Tutorial

Appendix

Appendix Contents

- What is Gaussian Process Regression?

What is Gaussian Process Regression?

Gaussian Process Regression Overview

Training Data

D_n: Training data X_n (inputs) and Y_n (outputs)

Kernel (Covariance Function)*

$k(x, x') = \Sigma(x, x')$

MVN Conditioning Equations

$$\mu(\mathcal{X}) = \Sigma(\mathcal{X}, X_n)\Sigma_n^{-1}Y_n \qquad \text{(Prediction Model)}$$

$$\Sigma(\mathcal{X}) = \Sigma(\mathcal{X}, \mathcal{X}) - \Sigma(\mathcal{X}, X_n)\Sigma_n^{-1}\Sigma(\mathcal{X}, X_n)^\top$$

Predicted Values

Mean: $\mu(x)$

Prediction Uncertainty

Variance: $\sigma^2(x)$

Sometimes expressed as $\Sigma(x)$

* Hyperparameter optimization is part of the procedure but not covered in this presentation

** $\mu(x)$: This function corresponds to the mean function or kriging model. This function is the prediction model, also known as the surrogate model, meta model or emulator.

Multivariate Normal (MVN) Conditioning Equations

The following must be calculated: Covariance Matrix, Mean and Variance

Covariance Matrix

$$\Sigma = \begin{pmatrix} \Sigma(\chi, \chi) & \Sigma(\chi, X_n) \\ \Sigma(X_n, \chi) & \Sigma_n = \Sigma(X_n, X_n) \end{pmatrix}$$

X_n: Training locations
χ: Testing (predictive) locations

Apply the covariance function $\Sigma(x, x')$ (kernel $k(x, x')$)

- $\Sigma(\chi, \chi)$: Covariance between testing (predictive) locations and themselves
- $\Sigma(\chi, X_n)$: Covariance between testing (predictive) and training locations
- $\Sigma(\chi, X_n)$: Covariance between training and testing (predictive) locations, which is the transpose of $\Sigma(\chi, X_n)$
- $\Sigma_n = \Sigma(X_n, X_n)$: Covariance between training locations and themselves

MVN Conditioning Equations (Mean and Variance)

Also referred to as “Gaussian process regression,” “kriging” or “kriging equations”

mean $\quad \mu(\mathcal{X}) = \Sigma(\mathcal{X}, X_n)\Sigma_n^{-1}Y_n$ Prediction Model (Vary χ to make predictions)

and variance $\quad \Sigma(\mathcal{X}) = \Sigma(\mathcal{X}, \mathcal{X}) - \Sigma(\mathcal{X}, X_n)\Sigma_n^{-1}\Sigma(\mathcal{X}, X_n)^\top$ Prediction Uncertainty

Example 1

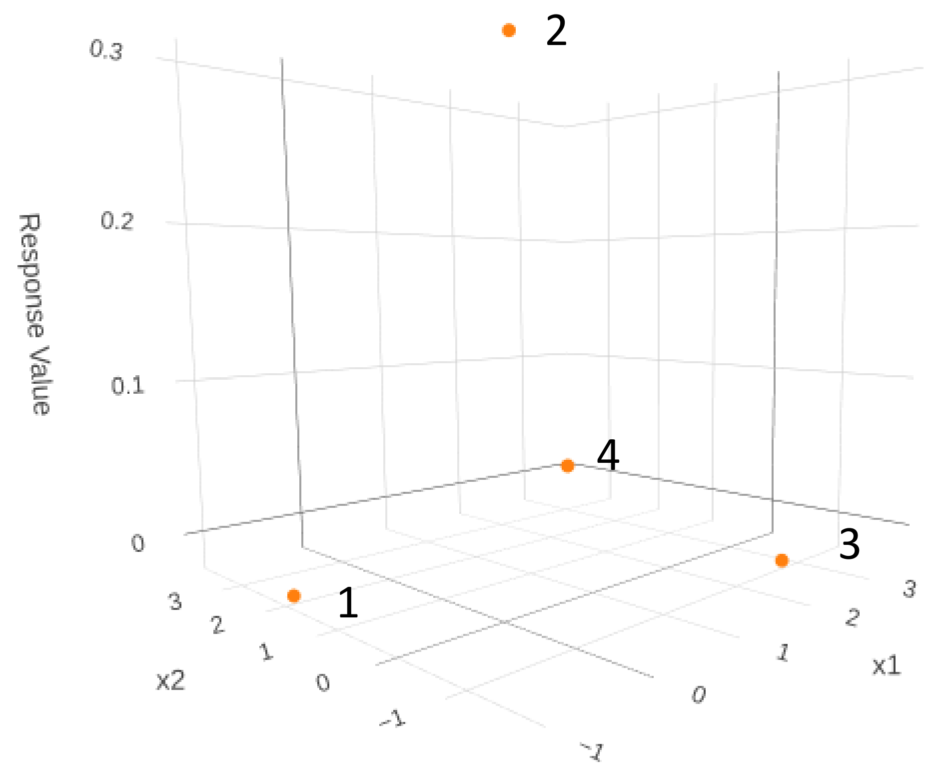
Example 1

Suppose a black box function was executed at 4 different samples (x1, x2 combinations)

With limited data (x and y), what does the response surface look like?

Training Data

Sample	x1	x2	y
1	-1.03	1.76	-1.56E-02
2	.49	.49	3.04E-01
3	1.77	-1.77	3.38E-03
4	3.62	3.76	5.43E-12



Training Data and Testing (Predictive) Locations

Suppose you have the following training data (X_n and Y_n) and testing locations (χ)

- X_n : The training design consists of 4 points
- χ : The test design (locations to make predictions) consists of 2 points

$$X = \begin{bmatrix} \chi \\ X_n \end{bmatrix} = \begin{bmatrix} .35 & .69 \\ .65 & .46 \\ -1.03 & 1.76 \\ .49 & .49 \\ 1.77 & -1.77 \\ 3.62 & 3.76 \end{bmatrix}$$

$$\begin{bmatrix} y* \\ Y_n \end{bmatrix} = \begin{bmatrix} ? \\ ? \\ -1.56e-02 \\ 3.04e-01 \\ 3.38e-03 \\ 5.43e-12 \end{bmatrix}$$

The goal is make predictions ($y*$) for points in χ

Note

- X_n: inputs of the training data
- Y_n: outputs of the training data
- χ or x: inputs of the testing data (predictive locations, i.e. points to make predictions)
- $y*$: predicted outputs
- D_n: Training data X_n and Y_n

X: upper case of Greek letter chi (pronounced kai in English)
χ: lower case of Greek letter chi

Calculation of the Covariance Matrix

1. Select a covariance (kernel) function
   - Many covariance functions (kernels) exist: Radial Basis Function (RBF), Matern 5/2, 3/2, Exponential, …
   - For this example, a form of the RBF covariance function is used. This covariance function is described as the "inverse exponentiated squared Euclidean distance"

$$k(x, x') = \Sigma(x, x') = \exp\{-||x - x'||^2\} = e^{-\|x-x'\|^2}$$

2. Calculate D (Distance Matrix)

$$D = \|X - X\|^2$$

"Norm between X and X, squared"

3. Calculate Σ (Covariance Matrix)

$$\Sigma = e^{-D}$$

Calculation of D

$$D = \begin{bmatrix}
\sqrt{(.35-.35)^2+(.69-.69)^2}^2 = 0 & \sqrt{(.35-.65)^2+(.69-.46)^2}^2 = .1429 & \sqrt{(.35--1.03)^2+(.69-1.76)^2}^2 = 3.0493 & \sqrt{(.35-.49)^2+(.69-.49)^2}^2 = .0596 & \sqrt{(.35-1.77)^2+(.69--1.77)^2}^2 = 8.068 & \sqrt{(.35-3.62)^2+(.69-3.76)^2}^2 = 20.1178 \\
.1429 & \sqrt{(.65-.65)^2+(.46-.46)^2}^2 = 0 & \sqrt{(.65--1.03)^2+(.46-1.76)^2}^2 = 4.5124 & \sqrt{(.65-.49)^2+(.46-.49)^2}^2 = .0265 & \sqrt{(.65-1.77)^2+(.46--1.77)^2}^2 = 6.2273 & \sqrt{(.65-3.62)^2+(.46-3.76)^2}^2 = 19.7109 \\
3.0493 & 4.5124 & \sqrt{(-1.03--1.03)^2+(1.76-1.76)^2}^2 = 0 & \sqrt{(-1.0\,3-.49)^2+(1.76-.49)^2}^2 = 3.9233 & \sqrt{(-1.03-1.77)^2+(1.76--1.77)^2}^2 = 20.3009 & \sqrt{(-1.03-3.62)^2+(1.76-3.76)^2}^2 = 25.6225 \\
.0596 & .0265 & 3.9233 & \sqrt{(.49-.49)^2+(.49-.49)^2}^2 = 0 & \sqrt{(.49-1.77)^2+(.49--1.77)^2}^2 = 6.746 & \sqrt{(.49-3.62)^2+(.49-3.76)^2}^2 = 20.4898 \\
8.068 & 6.2273 & 20.3009 & 6.746 & \sqrt{(1.77-1.77)^2+(-1.7\,7--1.77)^2}^2 = 0 & \sqrt{(1.77-3.62)^2+(-1.77-3.76)^2}^2 = 34.0034 \\
20.1178 & 19.7109 & 25.6225 & 20.4898 & 34.0034 & \sqrt{(3.62-3.62)^2+(3.76-3.76)^2}^2 = 0
\end{bmatrix}$$

Calculation of Σ

$$\Sigma = \begin{bmatrix} e^{0} = 1 & e^{-.1429} = .8668 & e^{-3.0493} = .0474 & e^{-.0596} = .9421 & e^{-8.068} = .0003 & e^{-20.1178} = 1.832\text{e-}9 \\ .8668 & e^{0} = 1 & e^{-4.5124} = .0110 & e^{-.0265} = .9738 & e^{-6.2273} = .0020 & e^{-19.7109} = 2.8\text{e-}9 \\ .0474 & .0110 & e^{0} = 1 & e^{-3.9233} = .0198 & e^{-20.3009} = 1.5\text{e-}9 & e^{-25.6225} = 7.5\text{e} - 12 \\ .9421 & .9738 & .0198 & e^{0} = 1 & e^{-6.746} = .0012 & e^{-20.4898} = 1.263\text{e-}9 \\ .0003 & .0020 & 1.5\text{e-}9 & .0012 & e^{0} = 1 & e^{-34.0034} = 1.7\text{e} - 15 \\ 1.832\text{e-}9 & 2.8\text{e-}9 & 7.5\text{e} - 12 & 1.263\text{e-}9 & 1.7\text{e} - 15 & e^{0} = 1 \end{bmatrix}$$

Calculation of Σ

$$\Sigma = \left[\begin{array}{cc|cccc} e^{0}=1 & e^{-.1429}=.8668 & e^{-3.0493}=.0474 & e^{-.0596}=.9421 & e^{-8.068}=.0003 & e^{-20.1178}=1.832\text{e-}9 \\ .8668 & e^{0}=1 & e^{-4.5124}=.0110 & e^{-.0265}=.9738 & e^{-6.2273}=.0020 & e^{-19.7109}=2.8\text{e-}9 \\ \hline .0474 & .0110 & e^{0}=1 & e^{-3.9233}=.0198 & e^{-20.3009}=1.5\text{e-}9 & e^{-25.6225}=7.5e-12 \\ .9421 & .9738 & .0198 & e^{0}=1 & e^{-6.746}=.0012 & e^{-20.4898}=1.263\text{e-}9 \\ .0003 & .0020 & 1.5\text{e-}9 & .0012 & e^{0}=1 & e^{-34.0034}=1.7e-15 \\ 1.832\text{e-}9 & 2.8\text{e-}9 & 7.5e-12 & 1.263\text{e-}9 & 1.7e-15 & e^{0}=1 \end{array}\right]$$

$$\begin{array}{cc} \Sigma(\chi,\chi) & \Sigma(\chi,X_n) \\ \Sigma(X_n,\chi) & \Sigma_n = \Sigma(X_n,X_n) \end{array}$$

Since Σ is symmetric, note that $\Sigma(X_n,\chi) = \Sigma(\chi,X_n)^{\mathrm{T}}$

Calculation of Predictive Quantities

The MVN conditioning equations are used to determine the predictive quantities mean and variance

$$\text{mean} \quad \mu(\mathcal{X}) = \Sigma(\mathcal{X}, X_n)\Sigma_n^{-1}Y_n$$

$$\mu(\chi) = y* = \begin{pmatrix} 0.2849657 \\ 0.2954011 \end{pmatrix}$$

Predicted values for locations in χ

$$\text{and variance} \quad \Sigma(\mathcal{X}) = \Sigma(\mathcal{X}, \mathcal{X}) - \Sigma(\mathcal{X}, X_n)\Sigma_n^{-1}\Sigma(\mathcal{X}, X_n)^\top$$

$$\Sigma(\chi) = \begin{pmatrix} 0.11154162 & -0.05042265 \\ -0.05042265 & 0.05155061 \end{pmatrix}$$

Prediction Uncertainty

The diagonal terms are the variances at prediction points 1 and 2

$$\sigma^2(\chi) = \begin{pmatrix} 0.11154162 \\ 0.05155061 \end{pmatrix}$$

R
Code to replicate this example in R

```r
library(plgp)

eps = sqrt(.Machine$double.eps)

# Training points
X = rbind(c(-1.03,1.76), c(.49,.49), c(1.77,-1.77), c(3.62,3.76))

# The goal is to fit this function: y(x) = x1 * exp(-x1^2 - x2^2)
y = X[,1] * exp(-X[,1]^2 - X[,2]^2)

# Test points
XX = rbind(c(.35, .69),c(.65, .46))
XX

# Sigma 22 (Sigma) and its inverse (Si)
# ################################################
# Distance among the Training Data
D = distance(X)
Sigma = exp(-D)
Si = solve(Sigma)

# Sigma 11
# ################################################
# Distance among the Testing Data
DXX = distance(XX)
SXX = exp(-DXX)

# Sigma 12 and Sigma 21 (Transpose of Sigma 12)
# ################################################
# Distance between training and testing data
DX = distance(XX, X)
SX = exp(-DX)

# Calculate the predictive mean and predictive variance
# ################################################
# Predictive mean
mup = SX %*% Si %*% y
mup

# Predictive variance
Sigmap = SXX - SX %*% Si %*% t(SX)
Sigmap
```

Output

```
Console   Terminal   Jobs
~/
> # ################################################
> # Predictive mean
> mup = SX %*% Si %*% y
> mup
          [,1]
[1,] 0.2849657
[2,] 0.2954011
>
> # Predictive variance
> Sigmap = SXX - SX %*% Si %*% t(SX)
> Sigmap
            [,1]        [,2]
[1,]  0.11154162 -0.05042265
[2,] -0.05042265  0.05155061
>
```

Mean

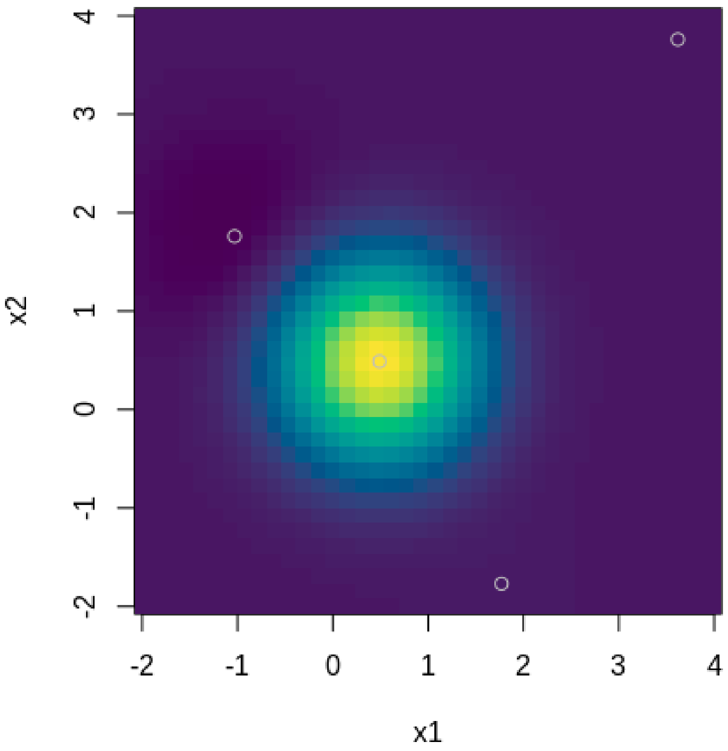
Variance

R
Code to replicate this example in R with Plots

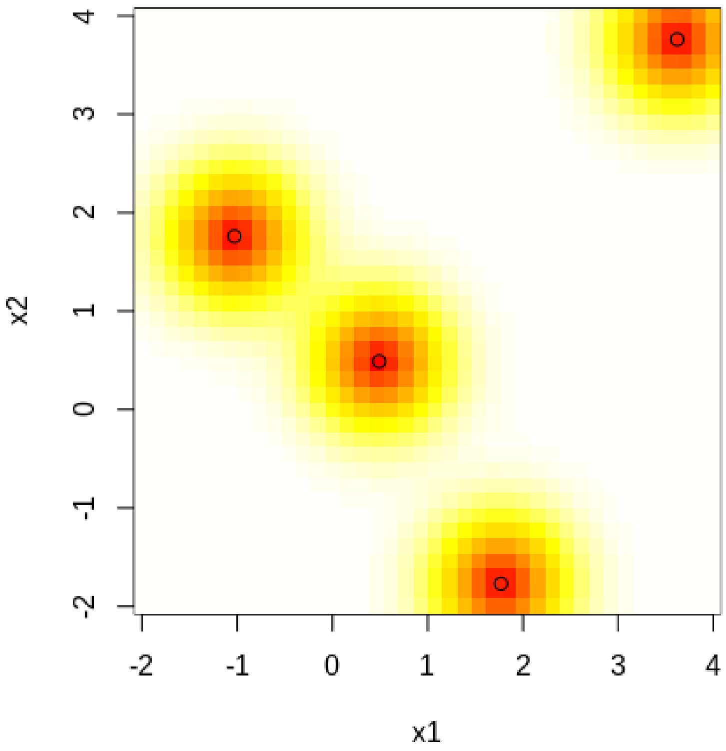
```
library(plgp)
library(lhs)

eps = sqrt(.Machine$double.eps)

# Training Data
# ################################################
# Training points
number_of_sample_points = 4
X = rbind(c(-1.03,1.76), c(.49,.49), c(1.77,-1.77), c(3.62,3.76))

# Observed values
# The goal is to fit this function: y(x) = x1 * exp(-x1^2 - x2^2)
y = X[,1] * exp(-X[,1]^2 - X[,2]^2)

# Testing Data
# ################################################
# Test points
number_of_test_points_per_axis = 40
xx = seq(-2, 4, length=number_of_test_points_per_axis)
XX = expand.grid(xx, xx)

# Sigma 22 (Sigma) and its inverse (Si)
# ################################################
# Distance among the Training Data
D = distance(X)
Sigma = exp(-D) + diag(eps, nrow(X))
Si = solve(Sigma)

# Sigma 11
# ################################################
# Distance among the Testing Data
DXX = distance(XX)
SXX = exp(-DXX)

# Sigma 12 and Sigma 21 (Transpose of Sigma 12)
# ################################################
# Distance between training and testing data
DX = distance(XX, X)
SX = exp(-DX)

# Calculate the predictive mean and predictive variance
# ################################################
mup = SX %*% Si %*% y
Sigmap = SXX - SX %*% Si %*% t(SX)

# Predictive standard deviation
diag(Sigmap)
sdp = sqrt(diag(Sigmap))

# Figure 5.5
par(mfrow=c(1, 2))
cols_a = hcl.colors(128, palette = "viridis")
cols_b = heat.colors(128)
image(xx, xx, matrix(mup, ncol=length(xx)), xlab='x1', ylab='x2', col=cols_a)
points(X[,1], X[,2])
image(xx, xx, matrix(sdp, ncol=length(xx)), xlab='x1', ylab='x2', col=cols_b)
points(X[,1], X[,2])

# Figure 5.6
persp(xx, xx, matrix(mup, ncol=number_of_test_points_per_axis), theta=-30, phi=30,
xlab='x1', ylab='x2', zlab='y', zlim = c(-.5,.5))
```

Predictive Quantities Mean and Standard Deviation

Mean: $\mu(\chi)$



Standard Deviation: $\sigma(\chi)$
(Square root of variance)



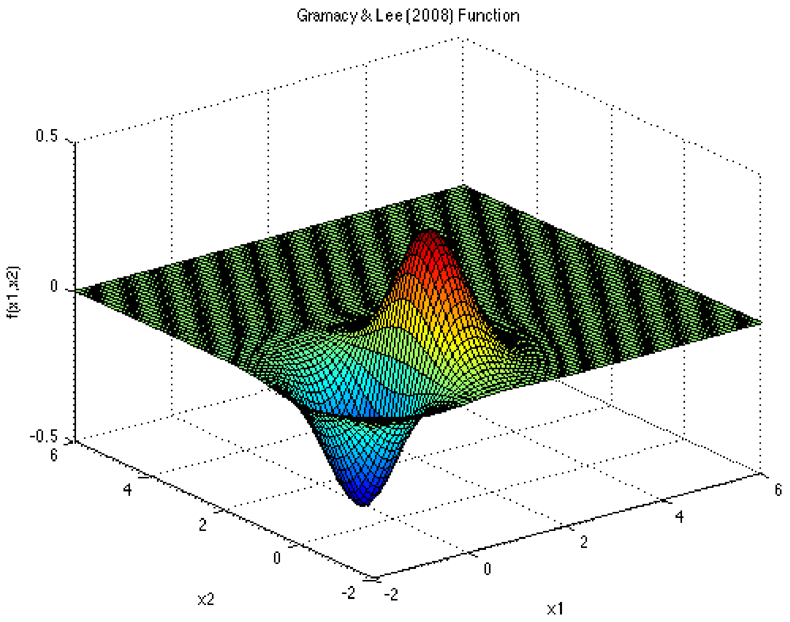
Not Good – High Standard Deviation – High Variance – High Prediction uncertainty (Prediction is not reliable)

Good – Low Standard Deviation – Low Variance – Low Prediction uncertainty (Prediction is reliable)

Prediction uncertainty is lowest near training data locations

Comparison of True Function and Prediction Model

True Function



$$f(\mathbf{x}) = x_1 \exp(-x_1^2 - x_2^2)$$

Source: https://www.sfu.ca/~ssurjano/grlee08.html

Prediction Model ($\mu(\chi)$)