A two-stage surrogate modelling approach for the approximation of models with non-smooth outputs

Author(s):
Moustapha, Maliki; Sudret, Bruno

Publication Date:
2019-06-26

Permanent Link:
https://doi.org/10.3929/ethz-b-000352847

Rights / License:
In Copyright - Non-Commercial Use Permitted
A two-stage surrogate modelling approach for the approximation of models with non-smooth outputs

M. Moustapha and B. Sudret

Chair of Risk, Safety and Uncertainty Quantification – ETH Zurich

Uncertainty Quantification in Computational Sciences and Engineering (UNCECOMP 2019)
Crete, Greece, June 26, 2019
Surrogate modelling in uncertainty quantification

- Surrogate models are unavoidable in uncertainty quantification (UQ)
- They are cheap approximations of expensive-to-evaluate computational models
- Built by learning from an experimental design

\[ D = \{(x^{(i)}, y^{(i)}) : x^{(i)} \in \mathbb{R}^M, y^{(i)} = M(x^{(i)}) , i = 1 \ldots N \} \]

- In the general setting, the computational model $M$ is assumed to have accommodating properties such as regularity
Non-smooth outputs in engineering

- The regularity assumption does not always hold:
  - Mathematical discontinuity
  - Instability patterns, e.g., snap-through
  - Multiple behaviors of a system, e.g., crash simulation

- Such models cannot be approximated by traditional metamodeling approaches

A two-stage surrogate modelling is considered to deal with such cases
Main idea

- **Identify** distinct behaviors of the system in the output space
- **Surrogate** each of them **separately**
- **Predict** using the **ensemble** of surrogates

**LEARNING**
1. Identification
2. Classification
3. Prediction

**PREDICTING**
1. Identification
2. Evaluation
3. Recombination
Learning

1. Identification
   - Partition the data according to the observed outputs

   \[ D_k = \left\{ (x^{(i)}, y^{(i)}) \in D : y^{(i)} \in C_k \right\}, \quad k = \{1, \ldots, K\} \]

   ▶ Expert knowledge or unsupervised learning

2. Classification
   - Partition the input space

   \[ \{R_k, k = 1, \ldots, K\} \text{ with } \bigcup_{k=1}^{K} R_k = R \text{ and } R_i \cap R_j = \emptyset \forall i \neq j \]

   ▶ Support vector machines, Classification tree, random forest, etc.

3. Prediction
   - For each dataset, build a surrogate model \( \{\hat{M}_k, k = 1 \ldots K\} \)

   ▶ Kriging, polynomial chaos expansions, support vector machines for regression, etc.
Predicting

1. Identification
   - Predict to which class belongs the new sample

2. Evaluation
   - Evaluate the sample using all surrogate models \( \{ \hat{M}_k, k = 1 \ldots K \} \)

3. Recombination
   - Recombine evaluations to get the final prediction:
     \[
     \hat{M}(x) = \sum_{k=1}^{K} w_k(x) \hat{M}_k(x) \quad \text{with} \quad \sum_{k} w_k(x) = 1
     \]
   - Two recombination strategies
     - Binary weighting
       \[
       w_k(x) = \mathbb{1}_{R_k}(x) = \begin{cases} 
       1 & \text{if } x \in R_k, \\
       0 & \text{otherwise}
       \end{cases}
       \]
     - Probabilistic weighting (classifier dependent)
Applications

Implementation

- Independent methods can be used in each step

1. Identification
   - Expert knowledge
   - $K$-means clustering
   - Gaussian mixtures
   - Hierarchical clustering
   - ...

2. Classification
   - SVC
   - CART
   - Random forest
   - Bayesian classification
   - ...

3. Prediction
   - Kriging
   - PCE
   - SVR
   - CART
   - ...

Application examples

- Two-dimensional illustration example
- Snap-through buckling
- Manhattan function
Applications

Implementation

- Independent methods can be used in each step

1. Identification
   - Expert knowledge
   - *K*-means clustering
   - Gaussian mixtures
   - Hierarchical clustering
   - ...

2. Classification
   - SVC
   - CART
   - Random forest
   - Bayesian classification
   - ...

3. Prediction
   - Kriging
   - PCE
   - SVR
   - CART
   - ...

Application examples

- Two-dimensional illustration example
- Snap-through buckling
- Manhattan function
Applications

Implementation

- Independent methods can be used in each step

1. Identification
   - Expert knowledge
   - \(K\)-means clustering
   - Gaussian mixtures
   - Hierarchical clustering
   - ...

2. Classification
   - SVC
   - CART
   - Random forest
   - Bayesian classification
   - ...

3. Prediction
   - Kriging
   - PCE
   - SVR
   - CART
   - ...

Application examples

- Two-dimensional illustration example
- Snap-through buckling
- Manhattan function
Support vector machines for classification

- Consider a dataset \( C = \{(x^{(i)}, \ell^{(i)}) : i = 1, \ldots, N\} \) with labels \( \ell^{(i)} = \{-1, 1\} \)
- SVC provides a binary classifier given by the sign of:

\[
M^{SVC}(x) = \sum_{i=1}^{N} \alpha_i \ell^{(i)} k(x^{(i)}, x) + b
\]

Recombination weighting
- Parametric sigmoid function

\[
P(\ell(x) = 1|M^{SVC}(x)) = \frac{1}{1 + \exp(A M^{SVC}(x) + B)}
\]
- \( A \) and \( B \) are computed by maximum likelihood
- The final weights are given by

\[
w_1(x) = 1 - P(\ell(x) = 1|M^{SVC}(x)) \quad \text{and} \quad w_2(x) = P(\ell(x) = 1|M^{SVC}(x))
\]
Random forest

- Intuitive and well suited to multi-class problems
- Enhancement of classification trees
- Use of bootstrap aggregation a.k.a bagging

Recombination weighting

- Consider $K$ classes $\{C_k, k = 1, \ldots, K\}$
- The probabilities are derived from the bootstrap replicates

$$w_k(x) = \frac{1}{B} \sum_{b=1}^{B} I[h_b(x) = k]$$

where $\{h_b, b = 1, \ldots, B\}$ is the $b$-th classifier obtained from the bootstrap set
Benchmarks

Tools

- **UQLab**: SVC and Kriging
- **MATLAB’s TreeBagger**: Random forest

Error metrics

- Normalized mean square error:
  \[
  NMSE = \frac{\sum_{i=1}^{N_{val}} (y_i - \hat{y}_i)^2}{\sum_{i=1}^{N_{val}} (y_i - \bar{y})^2}
  \]

- Maximum absolute error:
  \[
  MAE = \frac{\sum_{i=1}^{N_{val}} |y_i - \hat{y}_i|}{N}
  \]

where \(N_{val} = 10,000\) is the validation set
Two-dimensional mathematical function

- Function defined over $\mathbb{X} = [-\pi, \pi]^2$

\[ M(x) = \begin{cases} 
\sin(x_1) + 7\sin(x_2)^2 & \text{if } (x_1 - \pi)^2 + (x_2 - \pi)^2 - 2\pi^2 \geq 0, \\
 x_1 - 2x_2 - 10; & \text{otherwise} 
\end{cases} \]

- Linear and non-linear areas separated by a discontinuity
Two-dimensional mathematical function

- Training set of size 100

<table>
<thead>
<tr>
<th>Case</th>
<th>Case #1</th>
<th>Case #2</th>
<th>Case #3</th>
</tr>
</thead>
<tbody>
<tr>
<td>NMSE</td>
<td>0.0911</td>
<td>0.0530</td>
<td>0.0346</td>
</tr>
<tr>
<td>MAE</td>
<td>1.0124</td>
<td>0.2048</td>
<td>0.2436</td>
</tr>
</tbody>
</table>

- Partitioning of the input space

- Global model

- Binary weighting

- Probabilistic weighting
Snap-through buckling

- Consider a two-bar truss submitted to a vertical loading
- The load at a deformed position reads:
  \[ P = -2EA \tan(\alpha) (\cos(\alpha_0) - \cos(\alpha)) \]
- The corresponding displacement reads:
  \[ w = l_0 \cos(\alpha_0) (\tan(\alpha_0) - \tan(\alpha)) \]
- Snap-through may occur for some configurations of the system

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Distribution</th>
<th>Mean</th>
<th>C.o.V.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Load ((P) in N)</td>
<td>Gumbel</td>
<td>430</td>
<td>0.20</td>
</tr>
<tr>
<td>Young’s modulus ((E) in GPa)</td>
<td>Lognormal</td>
<td>210</td>
<td>0.10</td>
</tr>
<tr>
<td>Cross sectional area ((A) in cm(^2))</td>
<td>Gaussian</td>
<td>10</td>
<td>0.05</td>
</tr>
</tbody>
</table>
Snap-through buckling

(a) Experimental design

(b) Global model

(c) Local models

- Training set of size 100

<table>
<thead>
<tr>
<th></th>
<th>Case #1</th>
<th>Case #2</th>
<th>Case #3</th>
</tr>
</thead>
<tbody>
<tr>
<td>(NMSE)</td>
<td>0.2478</td>
<td>0.0803</td>
<td>0.0670</td>
</tr>
<tr>
<td>(MAE)</td>
<td>0.2714</td>
<td>0.0390</td>
<td>0.0399</td>
</tr>
</tbody>
</table>
Manhattan function

- Two-dimensional function with 3 areas and 10 classes

\[
\mathcal{M}(x) = \begin{cases} 
\text{Checker board} & \text{if } x_1 \geq 0, \\
\sin(7X_1) \cdot \sin(4X_2) & \text{if } x_1 \leq 0 \text{ and } x_2 \leq 0, \\
1 - \frac{2}{7}(2X_1 + 1)^2 - (2X_2 + 1)^2 & \text{if } x_1 \leq 0 \text{ and } x_2 \geq 0
\end{cases}
\]

Rai (2015)
Applications

Manhattan function

- Training set of size 500

<table>
<thead>
<tr>
<th></th>
<th>Case #1</th>
<th>Case #2</th>
<th>Case #3</th>
</tr>
</thead>
<tbody>
<tr>
<td>NMSE</td>
<td>0.1392</td>
<td>0.0146</td>
<td>0.0125</td>
</tr>
<tr>
<td>MAE</td>
<td>0.0868</td>
<td>0.0042</td>
<td>0.0097</td>
</tr>
</tbody>
</table>

(a) Partitioning of the input space

(b) Global model

(c) Binary weighting

(d) Probabilistic weighting
Summary

- Standard surrogate models do not allow for an effective approximation of non-smooth functions
- A two-stage surrogate modelling framework is proposed

Outlook

- Accuracy
  - Adaptive construction of the classifier
  - Adaptive sampling for unbalanced data
- Output identification
  - Advanced unsupervised learning techniques
  - Combine classification and clustering
Questions ?

Chair of Risk, Safety & Uncertainty Quantification

www.rsuq.ethz.ch

Thank you very much for your attention !
Support vector machines for classification

- Consider a dataset $\mathcal{C} = \{(\mathbf{x}^{(i)}, \ell^{(i)}) \; | \; i = 1, \ldots, N\}$ with labels $\ell^{(i)} = \{-1, 1\}$
- The prediction is given by the sign of the following classifier:
  \[
  \mathcal{M}^{SVC}(\mathbf{x}) = \sum_{i=1}^{N} \alpha_i \ell^{(i)} k(\mathbf{x}^{(i)}, \mathbf{x}) + b
  \]
  where $k(\bullet, \bullet)$ is a kernel function, e.g. Linear, Gaussian, Matérn 5/2
  - The parameters $\alpha_i$ and $b$ are obtained by solving a quadratic optimization problem
  \[
  \min_{\alpha} \frac{1}{2} \alpha^T \left( \tilde{K} YY^T \right) \alpha + c^T \alpha
  \]
  subject to: $\alpha^T Y = 0$, $\alpha_i \geq 0$, $i = \{1, \ldots, N\}$
  where $c = \{-1, \ldots, -1\}$ and $\tilde{K} = K + 1/C I_N$
- An outer optimization loop is used to fully calibrate the model
  - Parameters: The kernel parameters $\theta$ and a penalty term $C$
  - Metric: (Span) estimate of the leave-one-out error
Appendix

Bagged classification tree

Basic idea

- A decision tree recursively partition the space into disjoint sets
- Best split chosen according to some rules, e.g. impurity measure
- The algorithm ends when:
  - Further splitting does not add value (predefined stopping criterion)
  - No more splitting is possible

Bootstrap aggregation (Bagging)

- $B$ classifiers are trained by sampling with replacement the training set
- The final prediction is given by the most voted class
- Further enhancements can be considered e.g. random forest
Kriging

- Kriging assumes that $\mathcal{M}(\mathbf{x})$ is a trajectory of an underlying Gaussian process
  
  $$
  \mathcal{M}(\mathbf{x}) = \beta^T f(\mathbf{x}) + \sigma^2 Z(\mathbf{x}),
  $$

  - $\beta^T f(\mathbf{x})$: trend
  - $Z(\mathbf{x})$: zero-mean, unit variance Gaussian process
  - $\sigma^2$ process variance

- Given an experimental design, the prediction for a new point reads
  
  $$
  \mu_{\mathcal{M}}(\mathbf{x}) = f^T(\mathbf{x}) \hat{\beta} + r^T(\mathbf{x}) R^{-1} \left( \mathbf{y} - F^T \hat{\beta} \right)
  $$

  where $R_{ij} = R(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}; \hat{\gamma})$, $r(\mathbf{x}) = R(\mathbf{x}, \mathbf{x}^{(i)}; \hat{\gamma})$ and $F = F_{ij} = f_j(\mathbf{x}^{(i)})$

- Calibration by a two-step process
  
  - $\left\{ \hat{\beta}, \hat{\sigma}^2 \right\}$ are estimated by least-square
  - $\hat{\gamma}$ is estimated by maximum likelihood or cross validation