Steps C: surrogate models

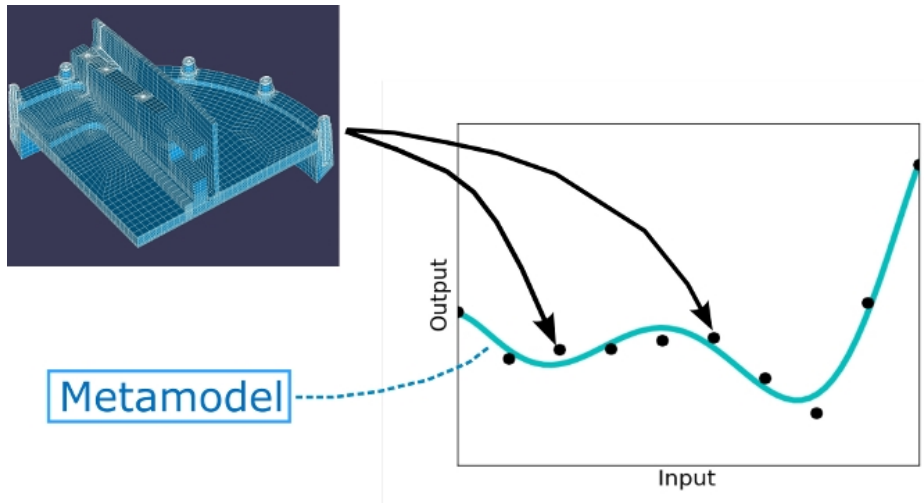
Michaël Baudin
Chu Mai

May 20, 2021

Introduction

- The goal of this course is to introduce the main surrogate methods used in steps C.
- Principles of surrogate models, cross-validation, coefficient of predictability, over-fitting.
- Linear regression, polynomial chaos, kriging.
- The emphasis is not on maths.
- Instead, we feed the intuition with graphics.

Metamodel - Definition



A metamodel is an approximation model that mimics the behavior of a computationally expensive simulator by training on *observations (data)* of the latter.

Metamodel - Definition

Model:

- Expensive simulator:

$$\boldsymbol{Y} = \boldsymbol{g}(\boldsymbol{X})$$

- $\boldsymbol{X} \in \mathbb{R}^{n_X}$: input vector
- $\boldsymbol{Y} \in \mathbb{R}^{n_Y}$: output vector

Metamodel:

$$\widetilde{\boldsymbol{Y}} = \tilde{\boldsymbol{g}}(\boldsymbol{X}, \boldsymbol{\theta}),$$

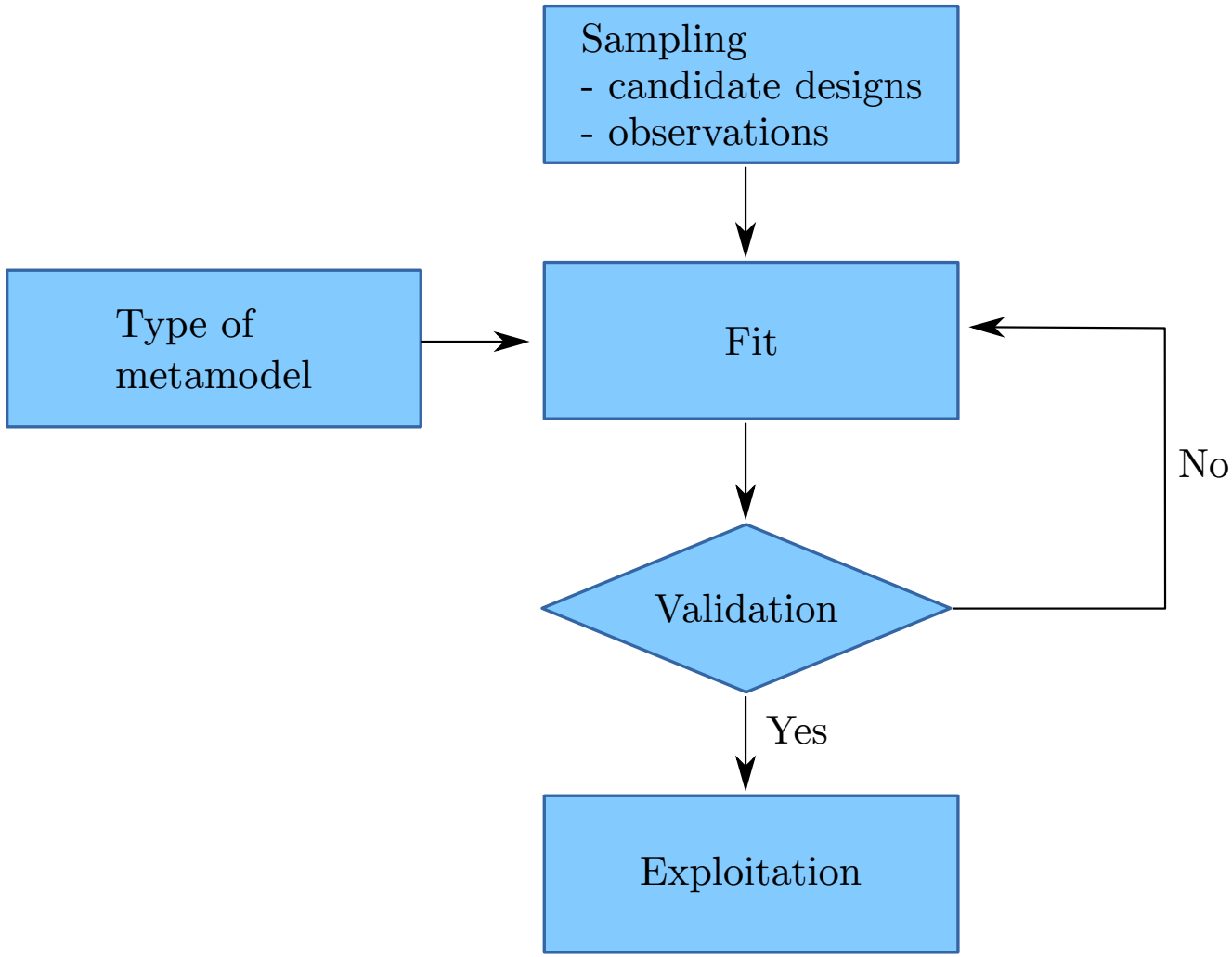
where $\boldsymbol{\theta}$ is the vector of parameters
By definition, two requirements of a metamodel $\tilde{\boldsymbol{g}}$ are:

- Accurate when predicting away from known observations
- Cheaper to evaluate than the primary simulator

Metamodel - Objectives

- Show functional relationships between input parameters and the output quantity of interest: impacts of variables
- Optimize the output quantity of interest: determine configurations that maximize the response or achieve specifications or customer requirements
- Replace the primary simulator in uncertainty propagation (surrogate model): e.g. in sensitivity analysis

Major steps for constructing a metamodel



Major steps for constructing a metamodel

Sampling (define an experimental design):

- A number of possible candidate designs are generated
- The designs are launched with the primary simulator

Constructing the metamodel:

- A type of metamodel is selected (among several available options)
- The metamodel is fitted to the available data
- The metamodel is validated (yes or no)
  - if yes: stop,
  - otherwise, reject the model.

Major steps for constructing a metamodel

If the model is rejected:

- change method for fitting: e.g. use advanced regression technique instead of least squares errors,
- change metamodel parameters: e.g. increase polynomial degree,
- increase the sample size if the model is not accurate enough or interesting behavior is not observed,
- change the type of metamodel: e.g. polynomial chaos instead of second-order polynomial response surface

Some types of metamodels

There are different metamodels available:

- Polynomial models
- Polynomial chaos models
- Kriging
- Neural networks,
- etc...

They all require to:

- tune the parameters $\boldsymbol{\theta}$: *training*,
- assess the quality of the prediction: *validation*.

Training and validation

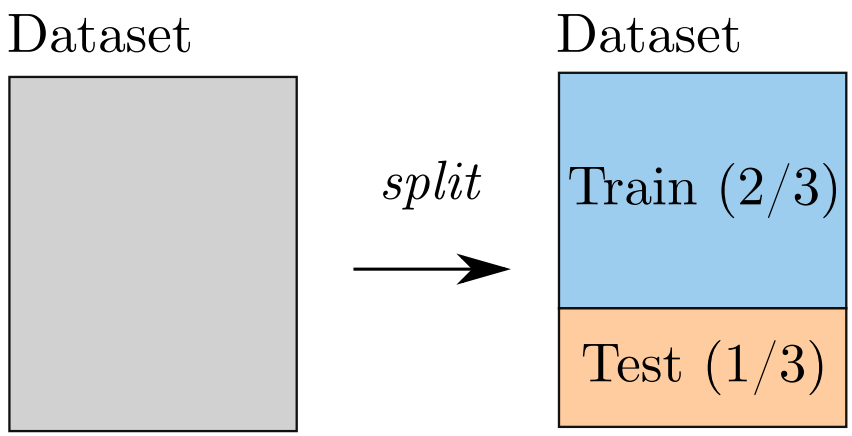
Creating a metamodel involves two steps.

1. Train: estimate the coefficients which fits to the data,
2. Test: quantify the predictability.

But, we must avoid overfitting: we must be able to predict datasets that have not been used to train the metamodel (more on this topic later).

One solution is to use cross-validation.

- Split the sample into a train sub-sample and a test sub-sample,
- Train the metamodel on the train sample,
- Test the metamodel on the test sample.



Training and validation

Assume that the output $y \in \mathbb{R}$ is a scalar, i.e. $n_Y = 1$.
Let $\left\{\mathbf{x}_t^{(j)}\right\}_{j=1,...,n}$ an i.i.d. sample of the random vector $\mathbf{X}$ that we use for training the metamodel.
We denote by g the model and $\tilde{g}$ the metamodel.
For $j = 1, ..., n$, let

$$y_t^{(j)} = g\left(\mathbf{x}_t^{(j)}\right), \qquad \tilde{y}_t^{(j)} = \tilde{g}\left(\mathbf{x}_t^{(j)}\right)$$

be the outputs of the model and metamodel on the training set.

Training and validation

The Root Mean Squared Error is:

$$RMSE = \left(\frac{1}{n} \sum_{j=1}^{n} \left(y_t^{(j)} - \tilde{y}_t^{(j)} \right)^2 \right)^{1/2}.$$

A low RMSE is better. The R^2 coefficient is:

$$R^2(g(\mathbf{x}_t), \tilde{g}(\mathbf{x}_t)) = 1 - \frac{\sum_{j=1}^{n} \left(y_t^{(j)} - \tilde{y}_t^{(j)} \right)^2}{\sum_{j=1}^{n} \left(y_t^{(j)} - \bar{y}_t \right)^2}$$

where $\bar{y}_t = \frac{1}{n} \sum_{i=1}^{n} y_t^{(j)}$. We usually have R^2 from 0 to 1, but a value < 0 or > 1 can occur. A value lower than zero indicates a poor prediction: $R^2 \approx 1$ is better.
The coefficient quantifies the part of the variance which is predicted by the model. It is the improvement of the metamodel with respect to the mean prediction.

Training and validation

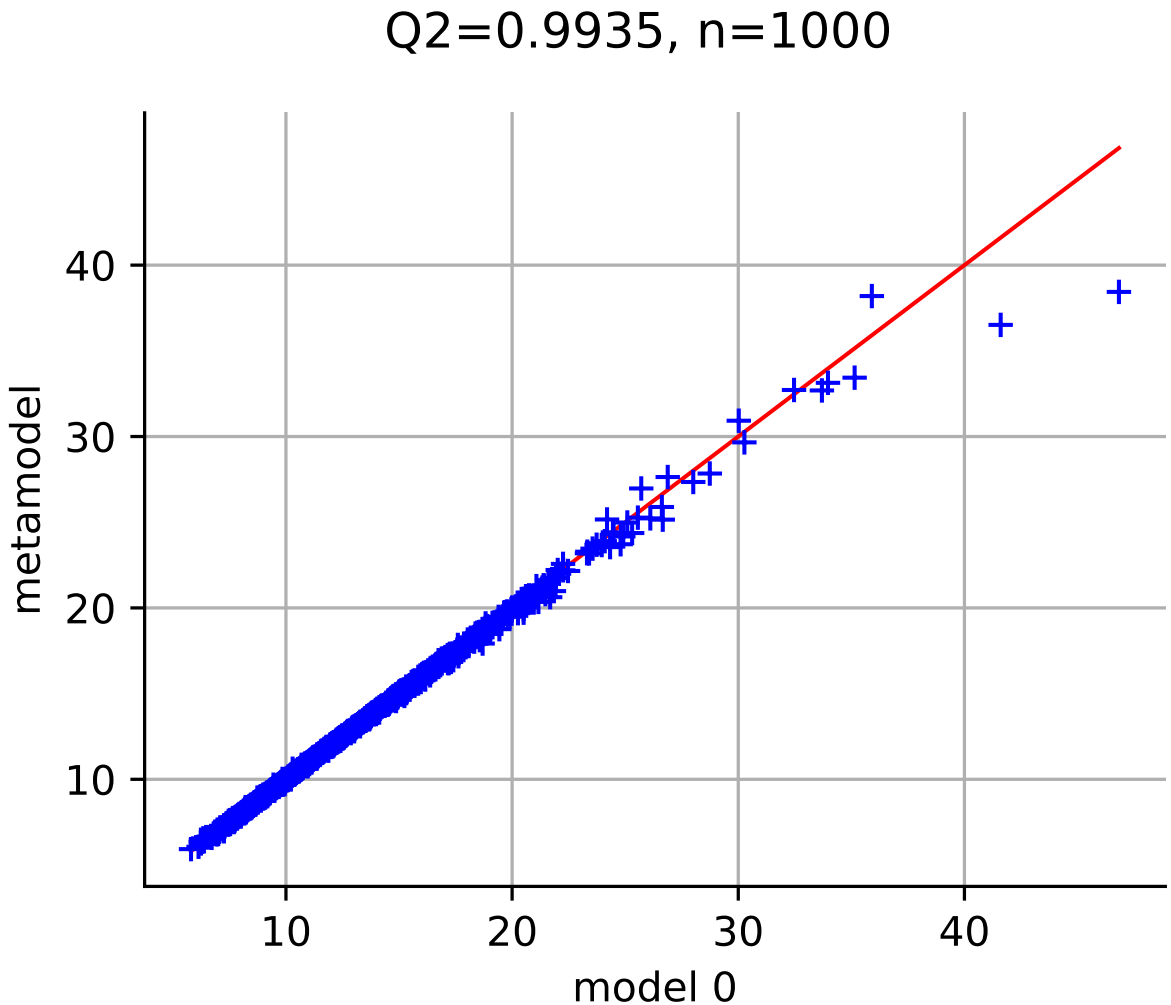
Denote $\left\{\mathbf{x}_v^{(j)}\right\}_{j=1,\dots,n}$ the validation sample.
The goal of this sample is to test the metamodel on a dataset that was not used for training.
Let $g(\mathbf{x}_v)$ and $\tilde{g}(\mathbf{x}_v)$ the outputs of the model and metamodel on the validation sample.
The Q^2 predictability coefficient is defined as the R^2 applied to the validation sample:

$$Q^2 = R^2(g(\mathbf{x}_v), \tilde{g}(\mathbf{x}_v)).$$

A suggestion of decision rule:

- $Q^2 > 0.95$: accept the metamodel,
- otherwise: try to improve the metamodel or reject it.

Graphical validation



Overfitting

Overfitting is the fact that the surrogate model fits to the training sample, but predicts poorly on new points.

- Any surrogate has coefficients tuned on the training sample.
- More coefficients generally reduce the error on the training sample
- *but* may increase the error on the validation sample.

There is a bias-variance trade-off to solve:

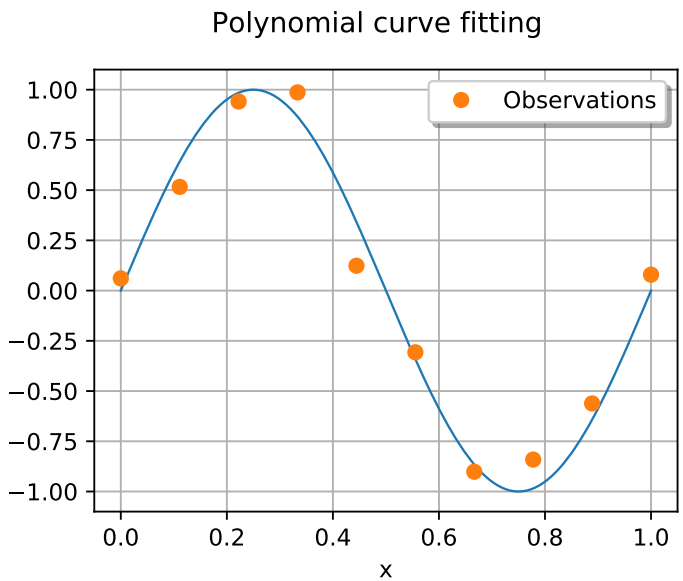
- More coefficients may reduce the variance (seen on the training sample),
- but may increase the bias (revealed on the validation sample).

Overfitting: example

Example 1

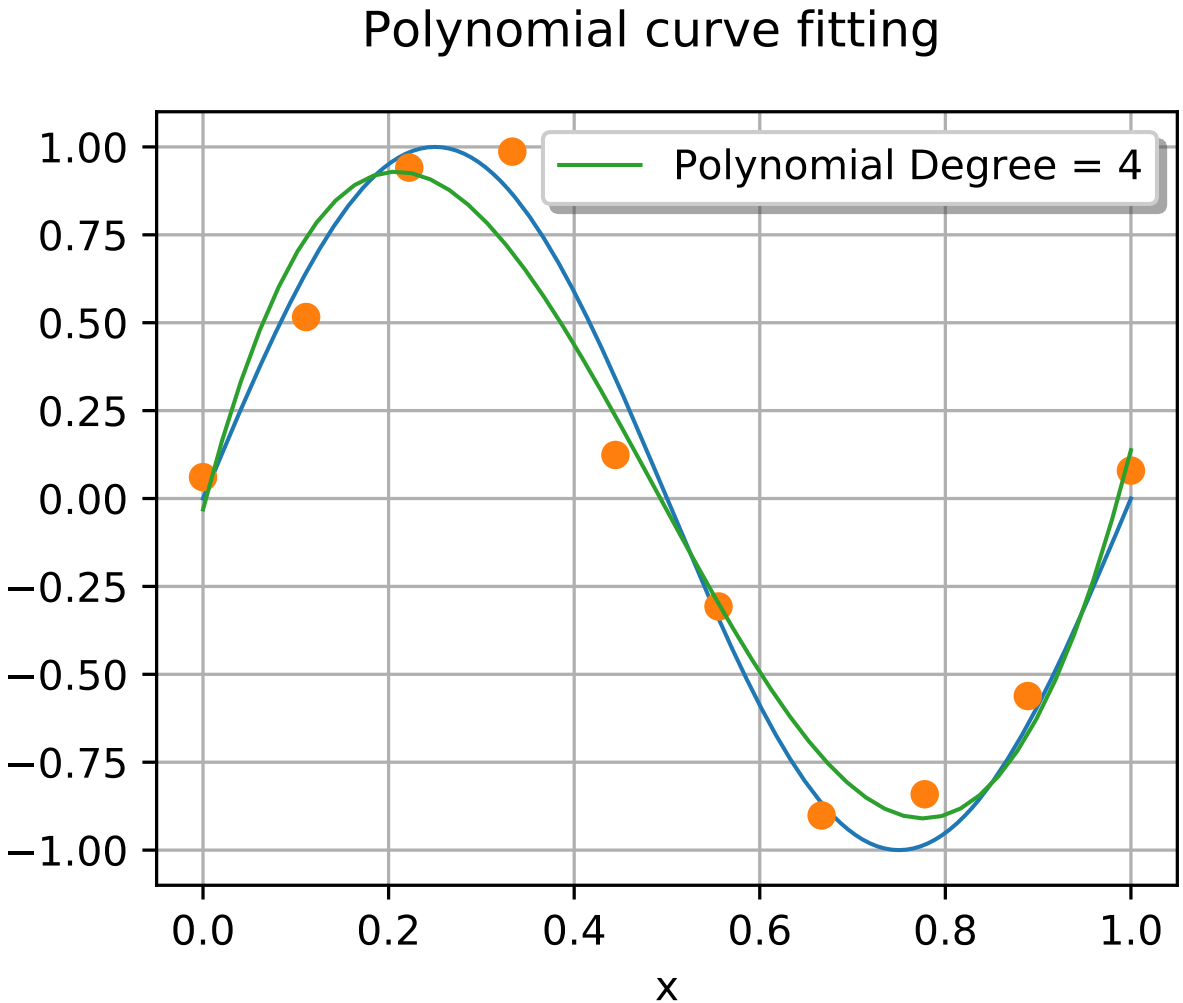
We least linear least squares with a polynomial ([Bis95], p.11).

- We have 10 points on the [0,1] interval, produced by adding a small noise to the sine function.
- We approximate it with linear regression based on polynomial canonical basis.



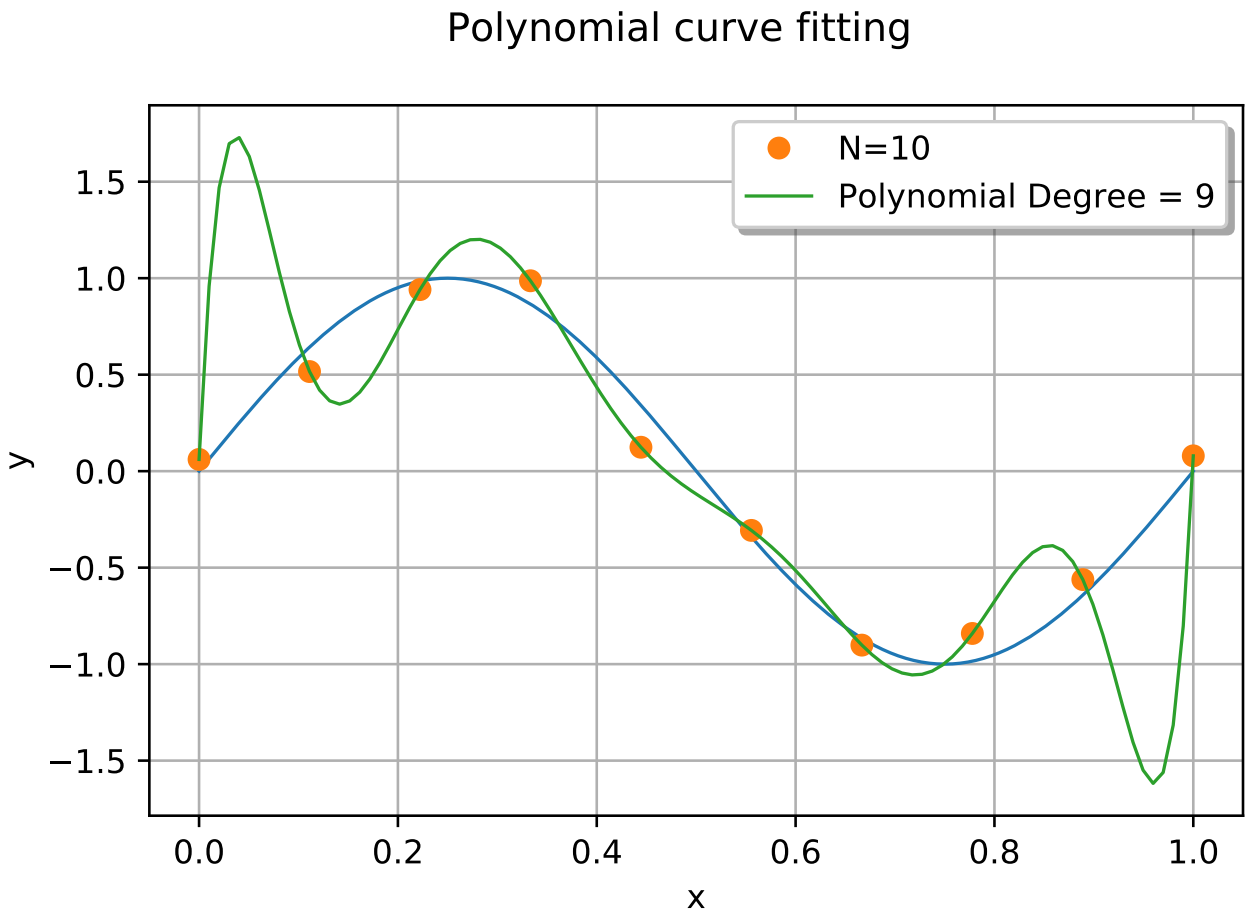
Overfitting: example

We use linear least squares to fit a degree 4 polynomial (5 coefficients).



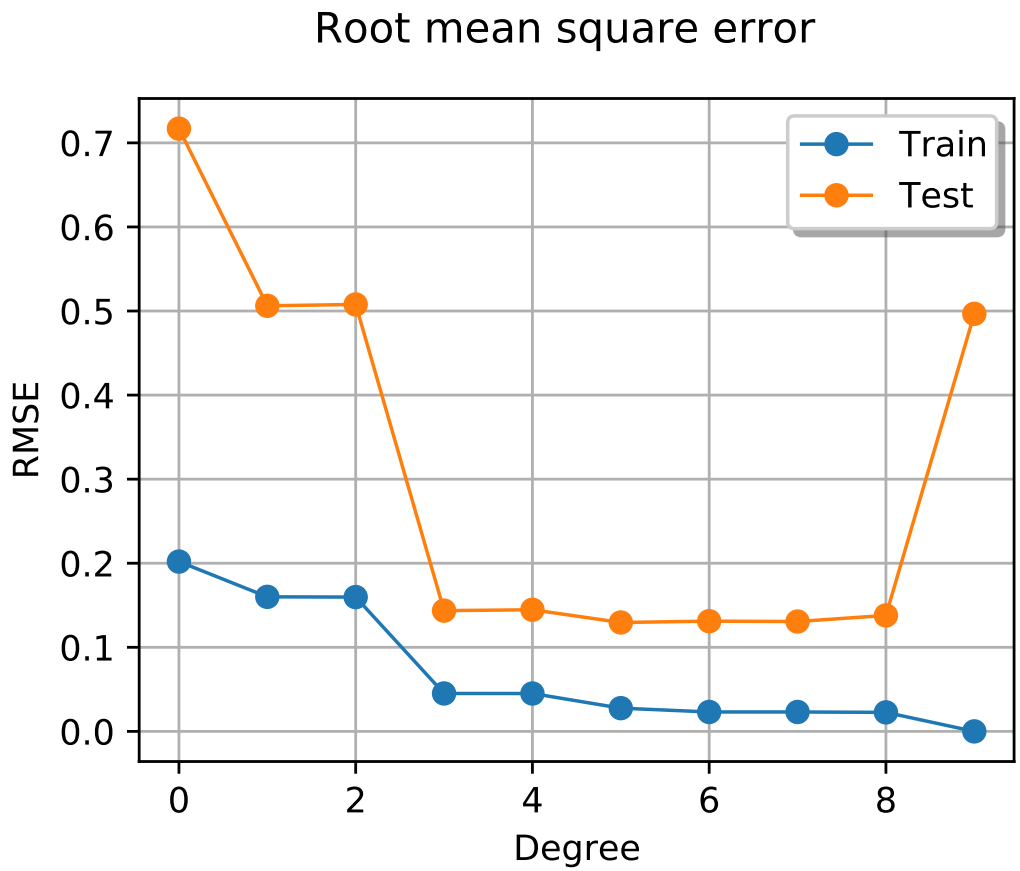
Overfitting: example

Increasing the degree does not always reduce the generalization error.



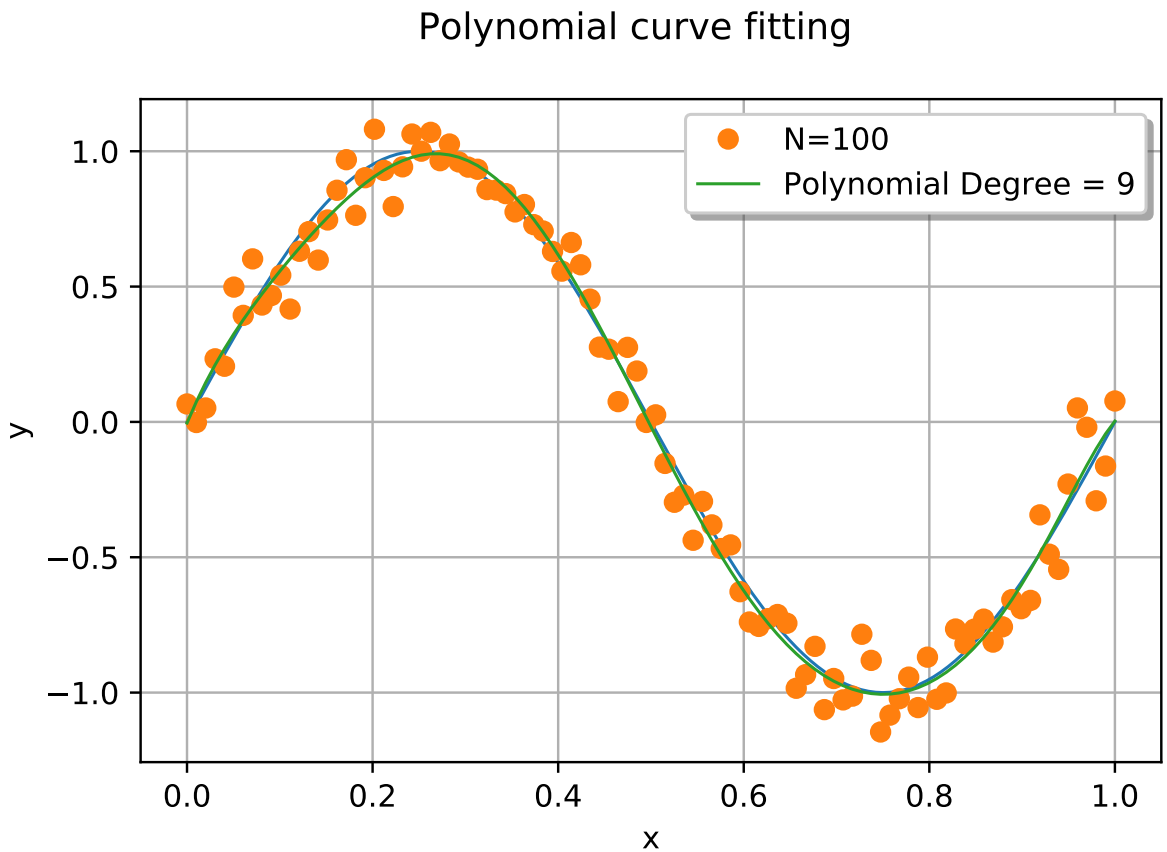
Overfitting: example

When we increase the degree, the root mean square error (RMSE) first decreases, then increases.



Overfitting: example

Adding points to the training sample reduces the validation error, ...

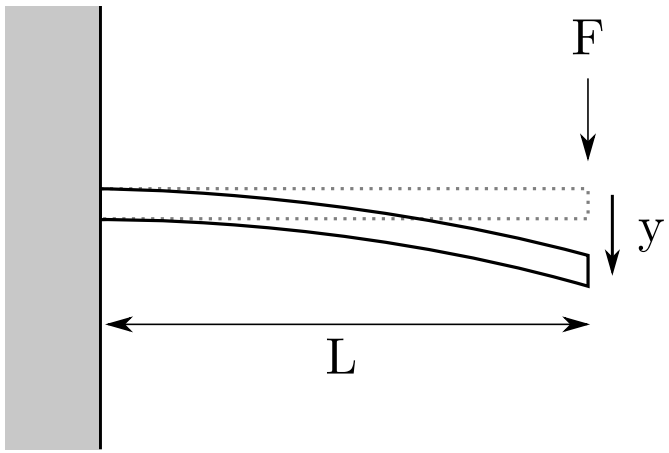


... but this is not always possible, e.g. the computer code may require costly simulations.

Polynomial chaos example

Example 2

- Consider the cantilever beam example $Y = \dfrac{F\,L^3}{3\,E\,I}$ with independent marginals: E: Beta, F: Lognormal, L: Uniform, I: Beta.
- We approximate it with *sparse* polynomial chaos: the model selection limits the number of coefficients in the model.



Polynomial chaos example

Questions:

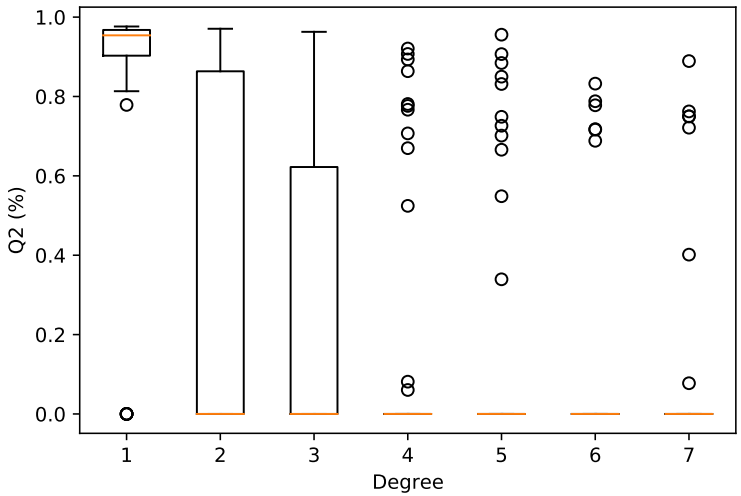
- How to set the sample size n ?
- How to set the polynomial degree d ?

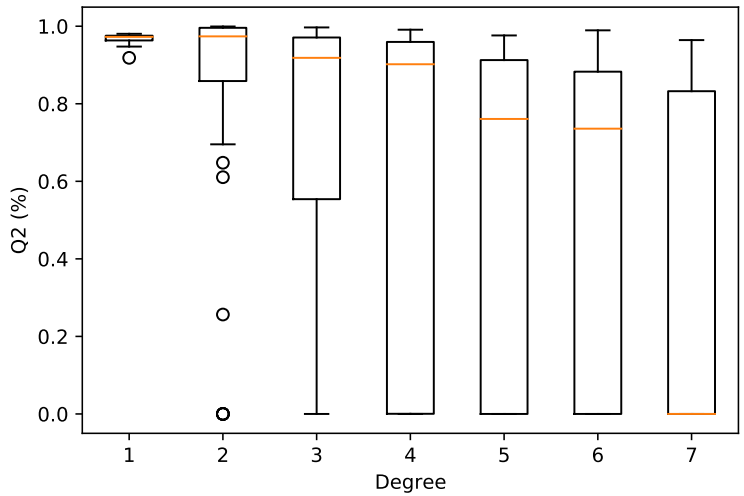
Method [GMS16]:

- Generate a training sample sample with size n (e.g. $n = 10$)
- Generate a validation sample with size 1000.
- Compute the Q^2.
- Repeat this experiment 50 times: get 50 Q^2 coefficients representing the variability of the Q^2 coefficient depending on the sample.
- We plot this sample of Q^2 coefficients in a boxplot: median, first quartile (25%) and third quartile (75%).

Polynomial chaos example

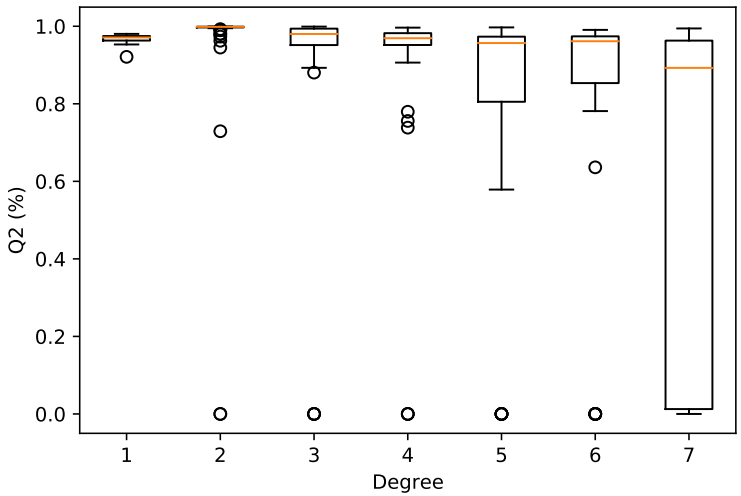
Left: n=10, Right: n=20.

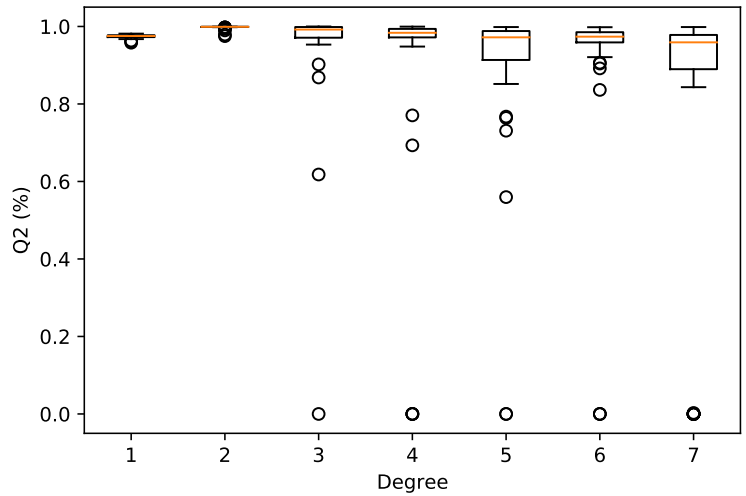




Polynomial chaos example

Left: n=30, Right: n=40.





Answer:

- How to set the sample size n ? $n \geq 40$
- How to set the polynomial degree d ? $d = 2$ is the smallest degree with high Q_2

Polynomial chaos example

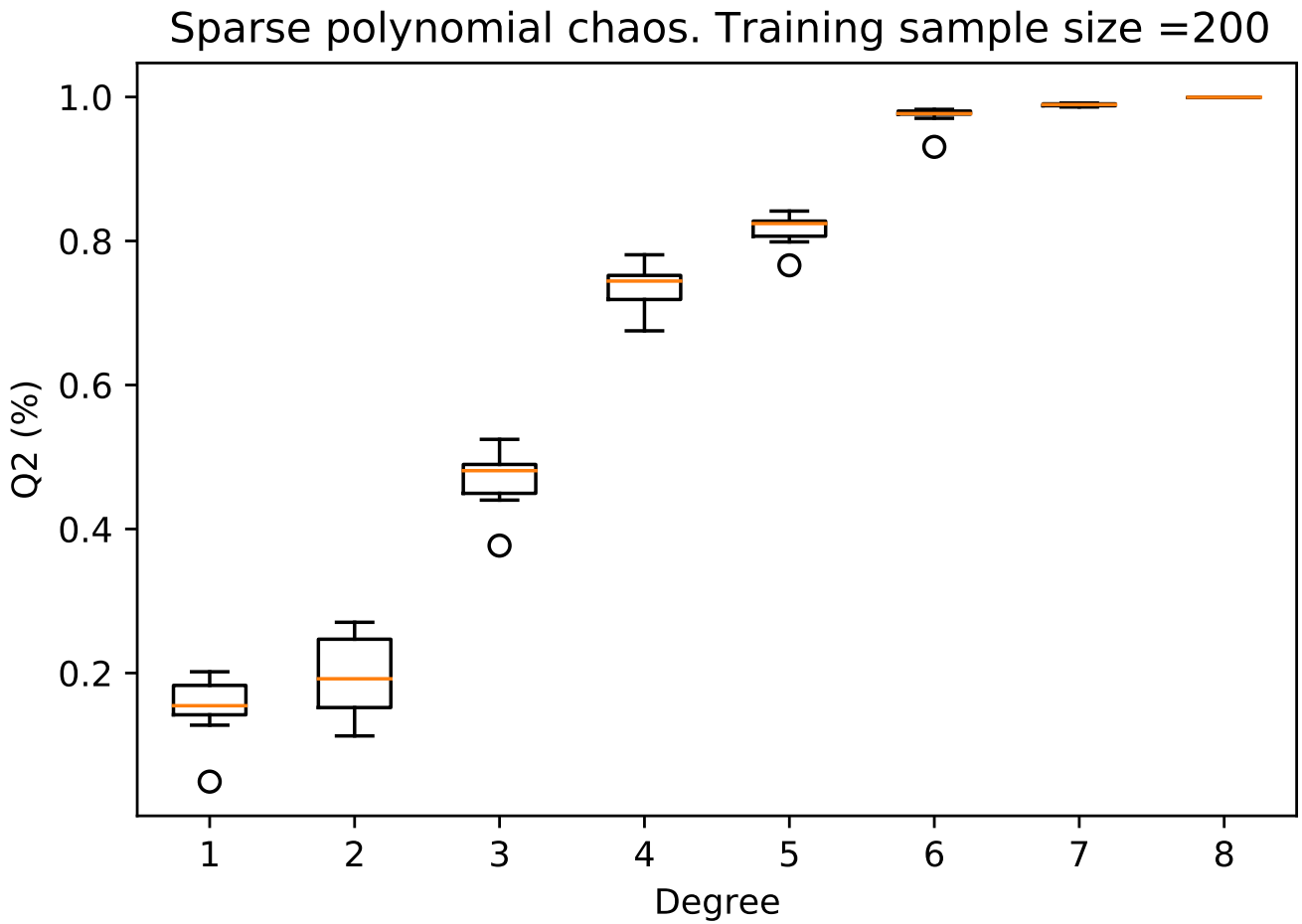
Example 3

- Consider the Ishigami model [IH90]:

$$Y = \sin(X_1) + a \sin(X_2)^2 + b X_3^4 \sin(X_1)$$

  where $a = 7$ and $b = 0.1$, with independent marginals: $X_1, X_2, X_3 \sim \mathcal{U}(-\pi, \pi)$.
- Training sample size: 200, test sample size: 200.
- We approximate it with *sparse* polynomial chaos: this limits the number of coefficients in the model.

Polynomial chaos example



Références I


Christopher M. Bishop, *Neural networks for pattern recognition*, Oxford university press, 1995.


Loïc Le Gratiet, Stefano Marelli, and Bruno Sudret, *Metamodel-based sensitivity analysis: Polynomial chaos expansions and gaussian processes*, Handbook of Uncertainty Quantification (Roger Ghanem, David Higdon, and Houman Owhadi, eds.), Springer International Publishing, Cham, 2016, pp. 1–37.


Tsutomu Ishigami and Toshimitsu Homma, *An importance quantification technique in uncertainty analysis for computer models*, [1990] Proceedings. First International Symposium on Uncertainty Modeling and Analysis, IEEE, 1990, pp. 398–403.

Appendix

Some types of metamodels

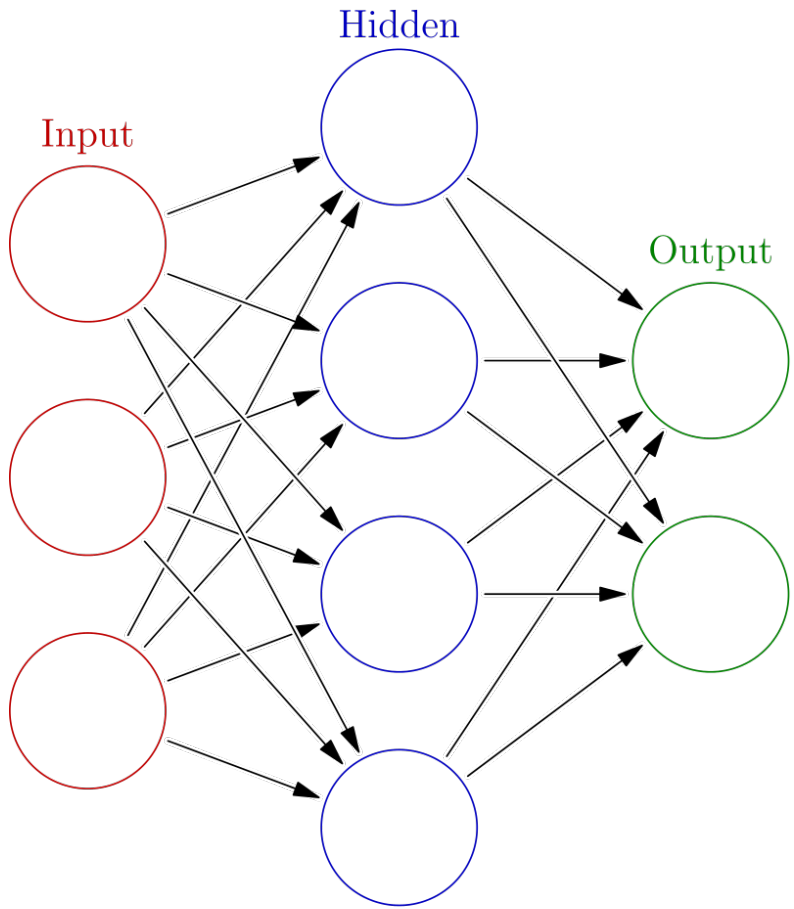
Radial basis function:

$$\tilde{\boldsymbol{Y}} = \sum_{k=1}^{N} \theta_k \psi(\|\boldsymbol{X} - \boldsymbol{X}_k\|)$$

where $\psi()$ being a radial basis function with its centers taken at $\boldsymbol{X}_k$, $k = 1, \ldots N$ in the experimental design

Some types of metamodels

Artificial neural network: Radial basis function is a single layer neural network with radial coordinate neurons



Regularized regression methods: minimize sum of squared errors under a constraint

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \sum_{i=1}^{N} (Y_i - \hat{f}(\boldsymbol{X}_i, \boldsymbol{\theta}))^2 + \lambda\, R(\boldsymbol{\theta})$$

- $R(\boldsymbol{\theta}) = ||\boldsymbol{\theta}||_2$: Ridge regression,
- $R(\boldsymbol{\theta}) = ||\boldsymbol{\theta}||_1$: LASSO regression,
- λ: non-negative regularization coefficient

Methods for fitting metamodels

Least square regression: minimize sum of squared errors of a linear regression model

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \sum_{i=1}^{N} \epsilon_i^2 = \arg\min_{\boldsymbol{\theta}} \sum_{i=1}^{N} (Y_i - \hat{f}(\boldsymbol{X}_i, \boldsymbol{\theta}))^2$$

Methods for fitting metamodels

Maximum likelihood estimation: e.g. assume that the errors ϵ are independently randomly distributed according to a normal distribution with standard deviation σ

$$\mathcal{L}(\boldsymbol{\theta}) = \frac{1}{(2\pi\sigma^2)^{N/2}} \prod_{i=1}^{N} \exp\left(-\frac{1}{2}\left(\frac{Y_i - \hat{f}(\boldsymbol{X}_i, \boldsymbol{\theta})}{\sigma}\right)^2\right)$$

$$\boldsymbol{\theta}^* = \arg\max_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta})$$

Methods for fitting metamodels

K-fold cross-validation method:

$\mathcal{K} : \{1, \ldots, N\} \mapsto \{1, \ldots, K\}$ partition of N observations to K roughly equal-sized parts, $K = N$: leave-one-out

$\hat{f}^{-k}()$: fitted metamodel with k-th part of data set aside

Cross-validation estimate of prediction error:

$$CV(\hat{f}, \boldsymbol{\theta}) = \frac{1}{N} \sum_{i=1}^{N} L(Y^i, \hat{f}^{-\mathcal{K}(i)}(X_i))$$

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} CV(\hat{f}, \boldsymbol{\theta})$$

Some types of metamodels

Polynomial models: (response surface)
Second-order model

$$\tilde{Y} = \theta_0 + \sum_{i=0}^{M} \theta_i \, X_i + \sum_{i=0}^{M} \theta_{ii} \, X_i^2 + \sum_{i<j} \sum_{j=2}^{M} \theta_{ij} \, X_i \, X_j$$

Polynomial chaos expansion:

$$\tilde{\boldsymbol{Y}} = \sum_{0 \leq |\boldsymbol{k}| \leq p} \theta_{\boldsymbol{k}} \psi_{\boldsymbol{k}}(\boldsymbol{X})$$

where $\psi_{\boldsymbol{k}}()$ being polynomial chaos functions

Some types of metamodels

Kriging: (Gaussian process regression)
Deterministic trend: linear regression on a fixed basis

$$m(\boldsymbol{X}) = \boldsymbol{r}(\boldsymbol{X})^T \boldsymbol{\theta}$$

Random fluctuation: zero-mean stationary Gaussian process of covariance function

$$k(\boldsymbol{X}, \boldsymbol{X'}) = \sigma^2 \, \rho(\left\| \boldsymbol{X} - \boldsymbol{X'} \right\|)$$