Metamodeling

ME598/494 Lecture

Max Yi Ren

Department of Mechanical Engineering, Arizona State University

February 11, 2016

1. preliminaries
   - 1.1 motivation
   - 1.2 ordinary least square
   - 1.3 information criterion
   - 1.4 sampling
2. regression methods
   - 2.1 ridge regression
   - 2.2 feed-forward NN
   - 2.3 radial basis NN
   - 2.4 kriging
   - 2.5 support vector regression
   - 2.6 training and testing

Motivation

Many optimization work require function evaluations through experiments or simulations (called black-box functions in general). Applying gradient-based methods directly on black-box functions is costly.

Metamodeling (or regression) techniques are useful in (1) identifying key factors that affect the performance of a design, and (2) creating an analytical model where gradient-based methods can be applied.

When to use metamodeling:

- Expensive simulation or computation
- No physical or computational model, only data is available
- Presence of numerical noise

Metamodeling is also called “surrogate modeling”, and is closely related to statistical learning (machine learning).

Applications

Computational Material Design: Granular Material

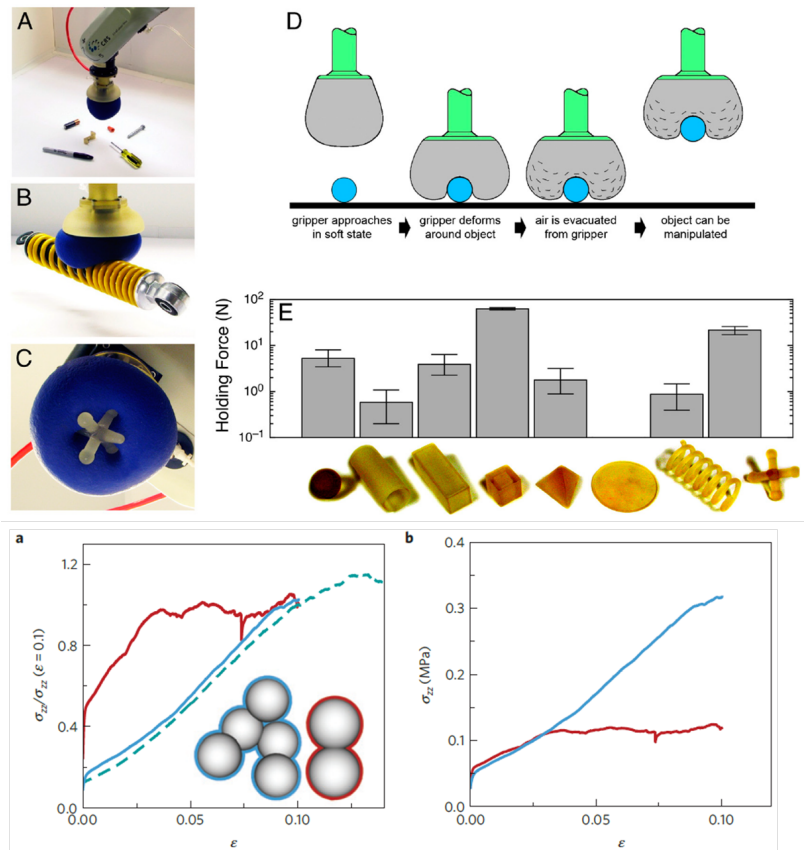
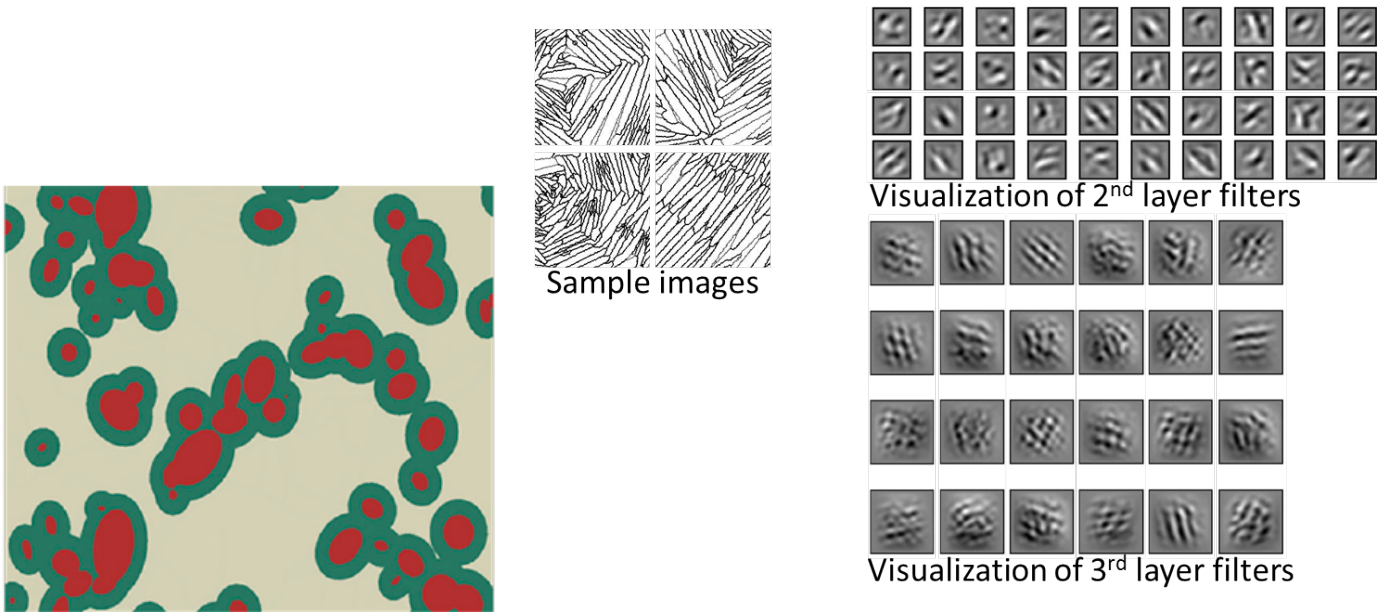


Figure: Figures from Brown et al. (2010) and Miskin et al. (2013). Granular material applications and stress-strain curves for different topologies.

Applications

Computational Material Design: Microstructure



(a) (b)

Figure: (a) Xu et al. (2014) Composite with filler (red), matrix (yellow) and interphase around filler (green). (b) Ongoing work on alloy microstructure optimal design.

Applications

Material Design: Stability of crystal structures

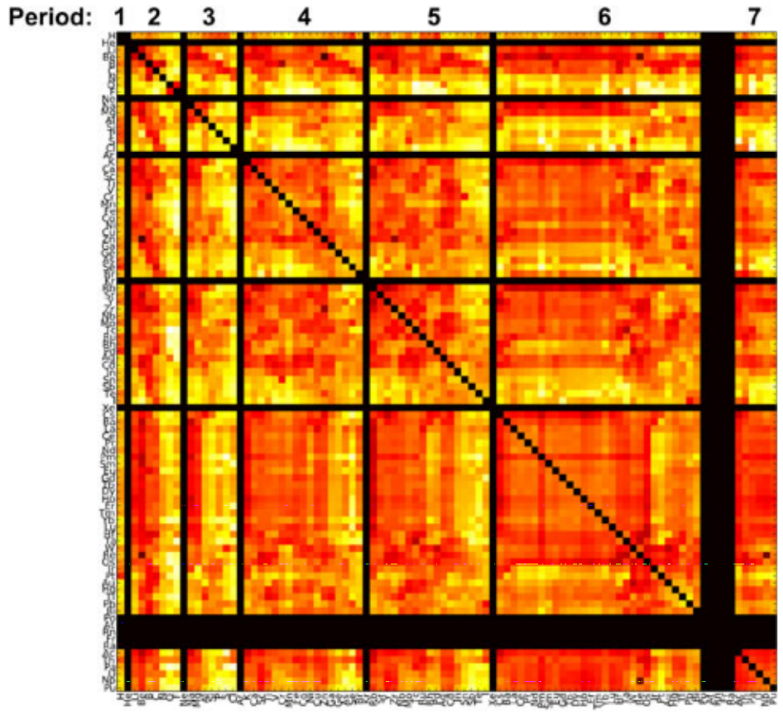
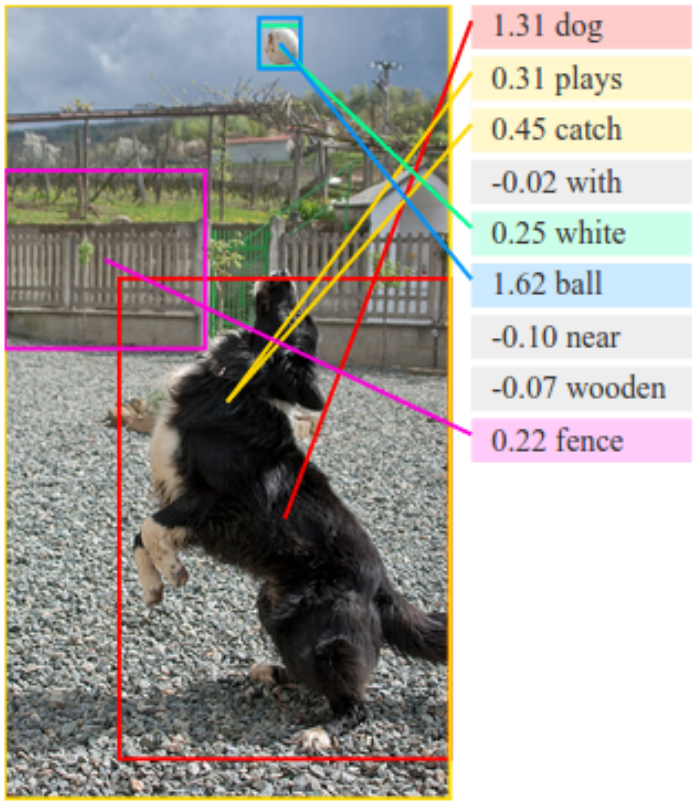


Figure: Predicted heat map of 1.6M candidate ternary compositions' stability rankings. Brighter colors imply greater stability. Figure: Saal et al. Materials Design and Discovery with High-Throughput Density Functional Theory: The Open Quantum Materials Database (OQMD).

Applications

Machine Learning: Multi-modal deep neural networks

(a) Man in black shirt is playing guitar.

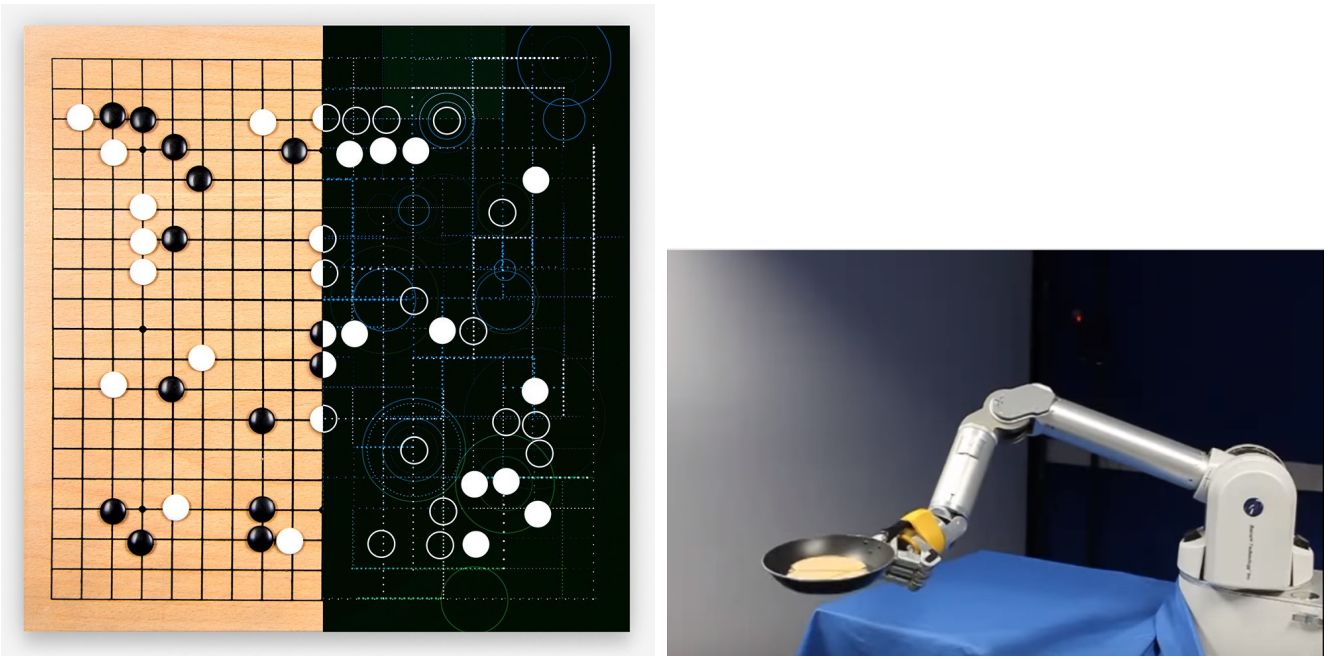


(b) Association between words and the image

Figure: A. Karpathy and F. Li, Deep Visual-Semantic Alignments for Generating Image Descriptions, http://cs.stanford.edu/people/karpathy/deepimagesent

Applications

Inverse Reinforcement Learning: Learn by demonstration



(a) Google AlphaGo (b) Flipping pancake

Figure: IRL (or immitation learning) is to learn transferable knowledge from human experts by assuming that human demonstrations are near-optimal (for some reason).

Overview

Steps for creating a metamodel for prediction purpose:

1. Sample the design space using a sample set $\mathbf{X}$ (n by p), with n observations (dependent variables) and p variables (covariates, independent variables).
2. Get objective (or constraints) values from simulations or experiments, denoted as $\mathbf{y}$, where each row is an "observation";
3. Split the data $(\mathbf{X}, \mathbf{y})$ into a training set and a test set;
4. Train a metamodel using the training data;
5. Test the model using the test data. Done if test performance is good. Otherwise try a different modeling method.

Note that training-test split is required for comparison across modeling techniques (OLS, SVR, Gaussian process, Bayesian networks, etc.). For a given modeling technique, crossvalidation is often required for parameter tuning.

Linear model

Consider a sample $\mathbf{x} = [x_1, x_2, \ldots]^T$ and its response y. A linear model is *linear in its coefficients*:

$$y = a_1 f_1(\mathbf{x}) + a_2 f_2(\mathbf{x}) + \ldots + a_p f_p(\mathbf{x}). \tag{1}$$

Here p is the degree of freedom (complexity) of a linear model.

The following models are all linear:

$$\begin{aligned} y &= a_1 x_1 + a_2 x_2 + \ldots + a_p x_p, \\ y &= a_1 x_1^2 + a_2 \sin(x_2), \\ y &= \exp(a_1 x_1 + a_2 x_2). \end{aligned} \tag{2}$$

Give an example of a nonlinear model.

Ordinary least square regression for linear models

Consider training data $(\mathbf{X}, \mathbf{y})$. A linear model assumes:

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon}, \tag{3}$$

where $\boldsymbol{\varepsilon}$ are random errors following a certain distribution. The goal of OLS is to estimate the model parameters $\boldsymbol{\beta}$ so that the estimations $\hat{\mathbf{y}} = \mathbf{X}\boldsymbol{\beta}$ are close to the observed $\mathbf{y}$. This can be formulated as follows:

$$\boldsymbol{\beta}^* = arg\min_{\boldsymbol{\beta}} \ ||\mathbf{y} - \mathbf{X}\boldsymbol{\beta}||^2, \tag{4}$$

which has an analytical solution:

$$\boldsymbol{\beta}^* = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}. \tag{5}$$

What can go wrong with this solution $\boldsymbol{\beta}^*$?

Information matrix

The matrix $\mathbf{X}^T\mathbf{X}$ is called the information matrix. Possible reasons for a singular information matrix: Cause 1 - "Small n large p":

1. Understand RNA string (10^5 in size) functionalities from a few cases (e.g. tumor) and controls;
2. Associate brain functionalities (e.g. speech) with brain cells (size depend on resolution) using a few fMRI scanning;
3. Identify your face with a few photos uploaded on Facebook;
4. Detect moving objects (time sensitive);
5. Estimate covariance of stocks (time sensitive);

Cause 2 - Linear dependency: e.g., x_1: Number of games won, x_2: Number of games lost, y: Final rank

How will redundant observation affect your model?

OLS for nonlinear models

For a nonlinear model $y = f(\mathbf{x}, \boldsymbol{\beta})$, the least square problem is:

$$\min_{\boldsymbol{\beta}} \sum_{i=1}^{n} (y_i - f(\mathbf{x}_i, \boldsymbol{\beta}))^2 \tag{6}$$

How do we solve this problem? What problems could we encounter? And how do we address those problems?

R^2 measure

$$R^2 = 1 - \frac{\sum_i (\hat{y}_i - y_i)^2}{\sum_i (y_i - \bar{y})^2},$$

where $\mathbf{y}$ are from the test data, $\hat{\mathbf{y}} = \mathbf{X}(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$ are estimates of $\mathbf{y}$. $\bar{y}$ is the average of $\mathbf{y}$. We can use R^2 to evaluate the test performance of the model. Higher R^2 indicates better performance.

If your regression model has low R^2 value, first try normalizing $\mathbf{X}$:

$$x_{ij} = \frac{x_{ij} - \bar{x}_j}{\sigma_j},$$

where $\bar{x}_j$ and σ_j are the mean and standard deviation on dimension j.

Crossvalidation for model selection

For a given modeling technique, e.g., OLS, one would still like to choose among models, e.g., the number of covariates or degree of polynomials. One way is to perform crossvalidation on each model and pick the one with the lowest average validation error.

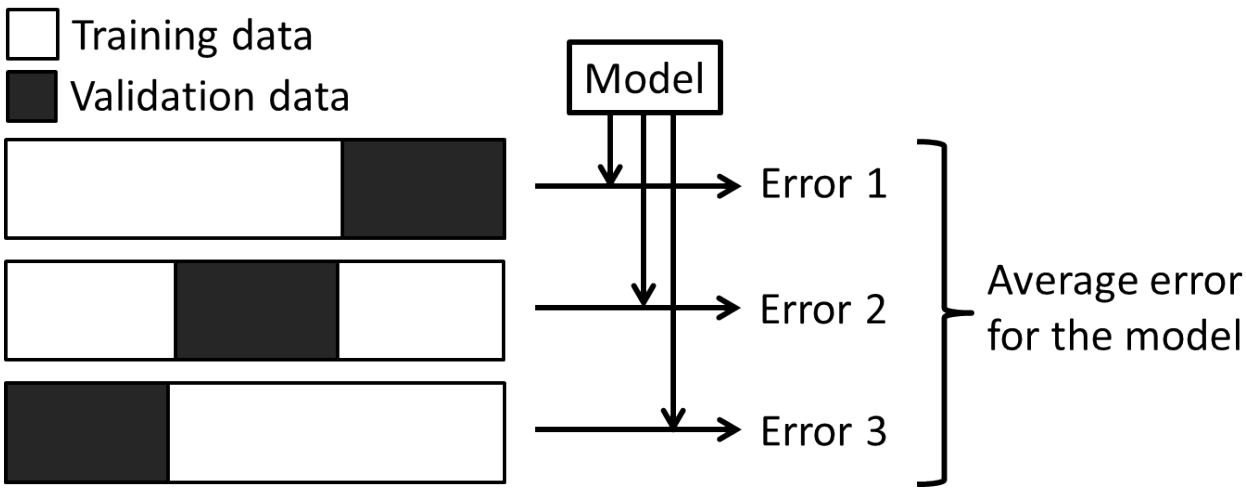


Figure: Three-fold crossvalidation

When validation size is one, it's called leave-one-out crossvalidation.

Akaike information criterion

Occam's razor: Among competing hypotheses, the hypothesis with the fewest assumptions should be selected.

AIC: A measure of goodness of fit and model complexity, for a given set of data. Provides a means for model selection.

$$AIC = 2p - 2\ln(L), \tag{7}$$

where p is the number of parameters and L is the maximum likelihood.

In OLS

$$L = \exp\left(-\frac{1}{2}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2\right), \tag{8}$$

where n is the sample size.

AIC and leave-one-out crossvalidation are asymptotically ($n \to \infty$) equivalent.

Akaike information criterion

AICc: AIC with a correction:

$$AICc = AIC + \frac{2p(p+1)}{n-p-1}. \tag{9}$$

Use AICc instead of AIC when $n/p < 40$.

Bayesian information criterion

BIC:

$$BIC = -2\ln(L) + p\ln(n), \tag{10}$$

Compared with AIC, BIC penalizes the number of covariates more strongly.

CV, AIC, BIC on model selection

Information-criterion based model selection (AIC, BIC) is computationally less expensive than CV, but it relies on a proper estimation of model complexity (degree of freedom).

For OLS with large data size (n) and relatively smaller dimensions (p), information criterion is more recommended than CV.

On the other hand, CV is applicable across various modeling techniques.

Exercise

Find a polynomial model for the following vapor-liquid equilibria data for a water-1,4 dioxane system:

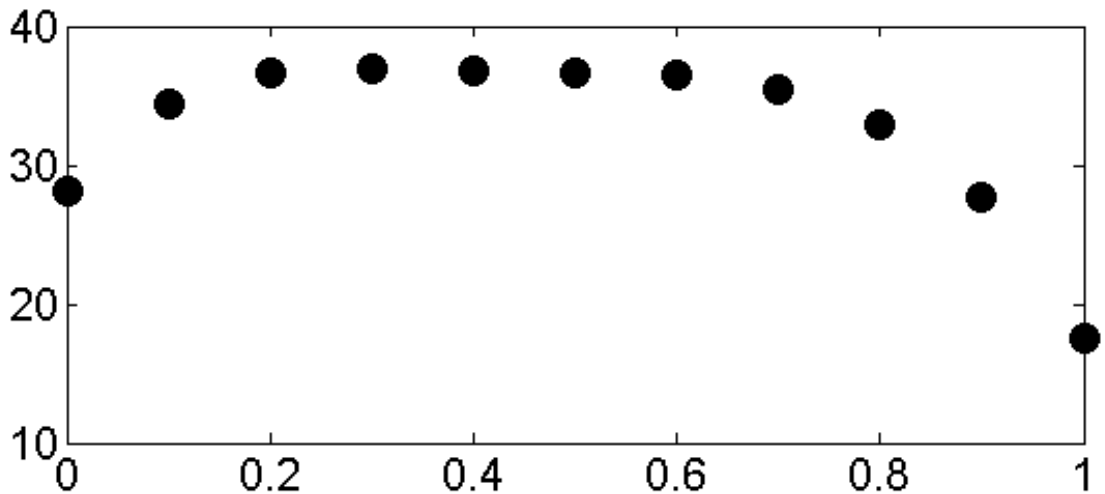
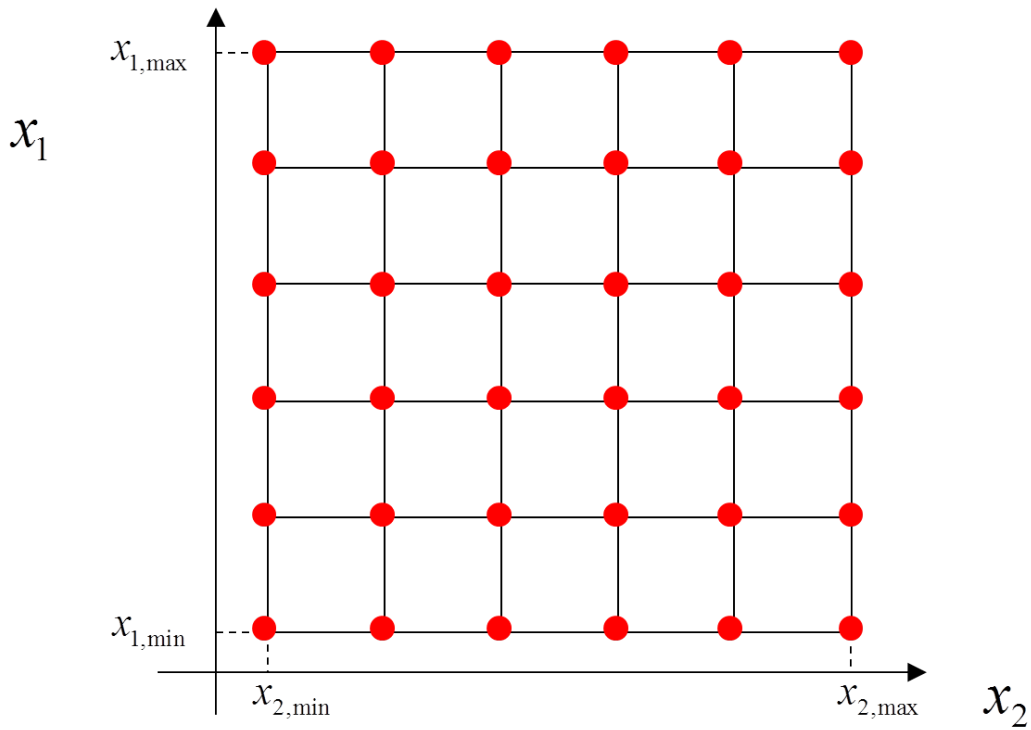


Figure: Exercise from M. Kokkolaras, McGrill University

Sampling

Design of experiments (optimal experiment design): Effectively sample the design space to create a statistical model with high prediction performance.

A naive way is to use a *full factorial experiment*:



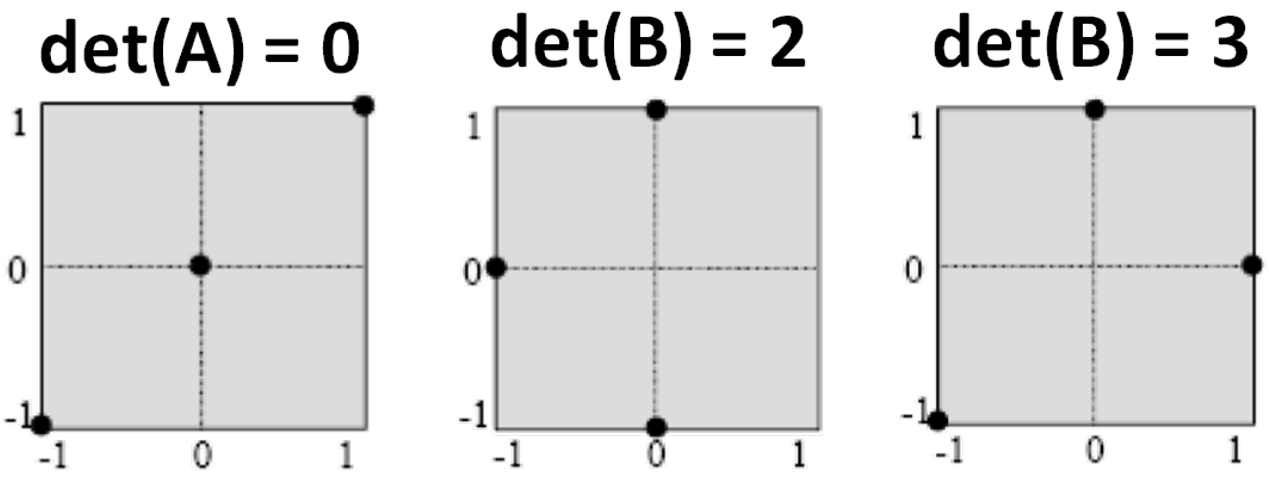
Full factorial sampling costs l^p samples.

D-optimal design

Consider the OLS solution:

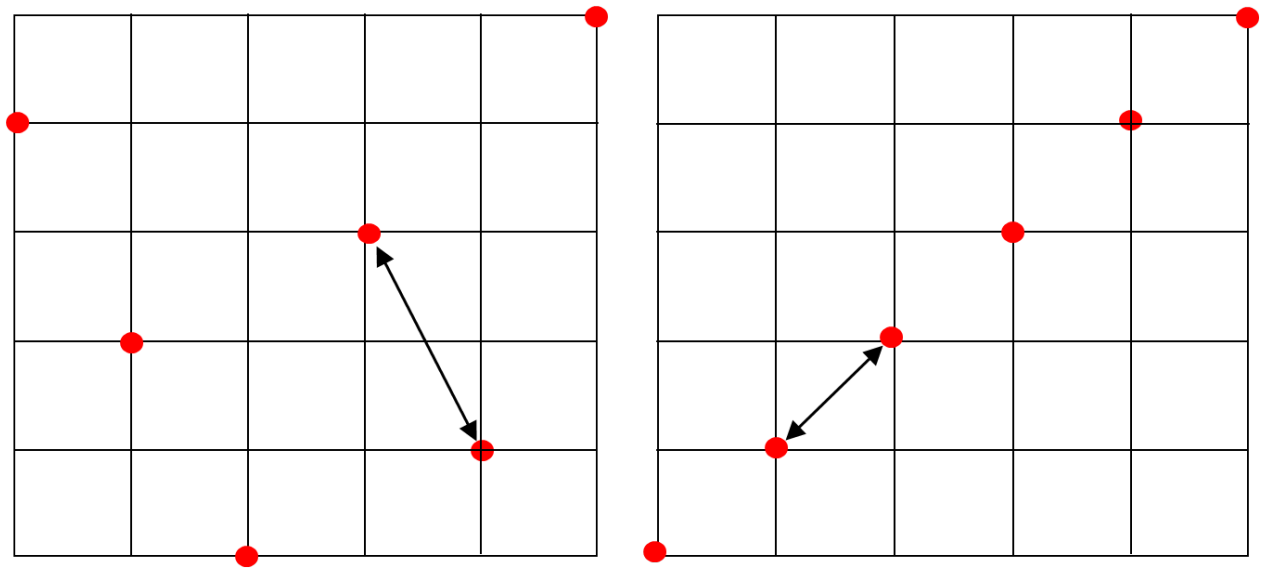
$$\boldsymbol{\beta}^* = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}. \tag{11}$$

The variance between $\boldsymbol{\beta}^*$ and the unknown true model can be reduced with larger $|\mathbf{X}^T\mathbf{X}|$. Therefore, it is preferred to maximize $|\mathbf{X}^T\mathbf{X}|$ for a fixed amount of samples.



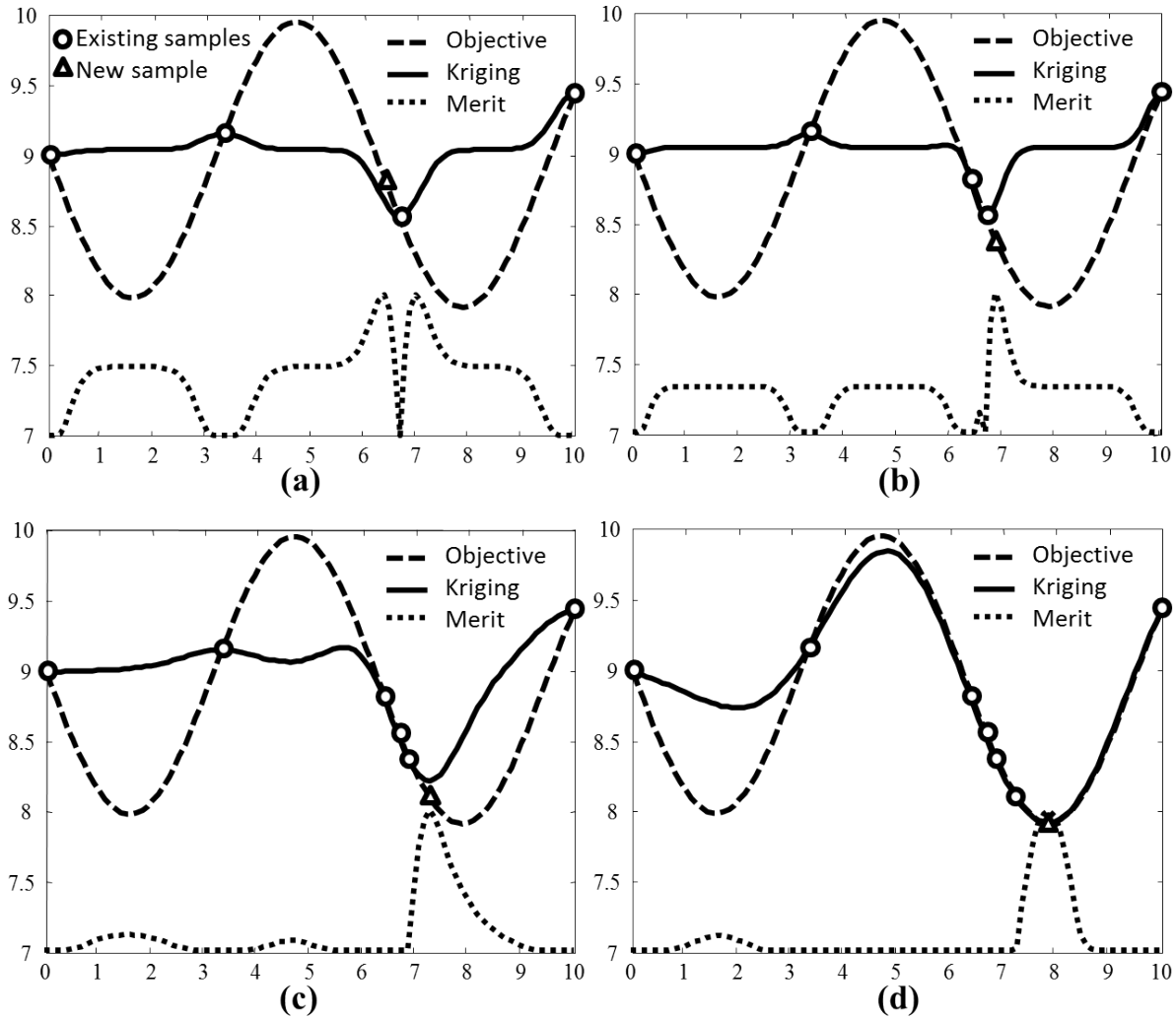
Latin hypercube

Latin hypercube sampling (LHS) uses l samples regardless of the number of variables (p), and is therefore widely adopted.



In LHS, there does not exist samples that share the same value on any variable. To implement LHS, one should also include dispersion criteria, e.g., maximizing the minimum distance between sample points, or minimizing the correlation.

Active learning (Adaptive sampling)

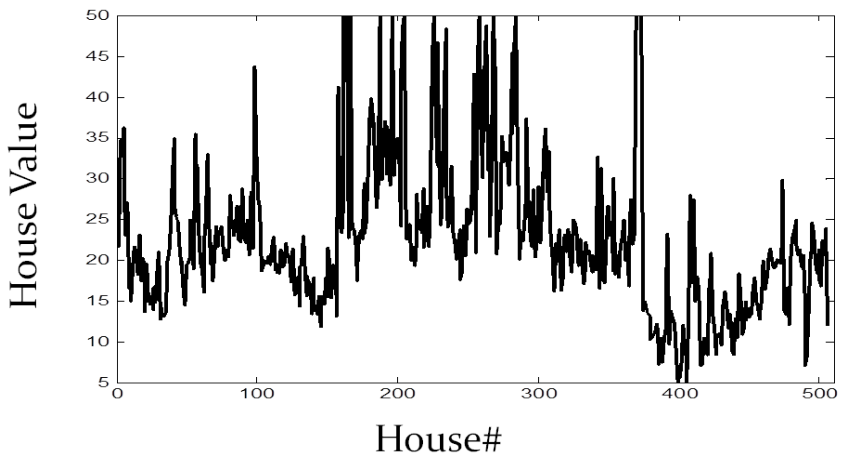


The Efficient Global Optimization (EGO) algorithm finds new samples based on the current model and model uncertainty.

Regression methods

When the function to model cannot be linearly approximated by design variables, or we don't know what features (e.g., polynomial terms) to use for modeling the observation, OLS may not work well.

An example from Matlab House.data: $\mathbf{X}$ (506×13): 506 houses with 13 parameters, $\mathbf{y}$ (506×1): house values.



	OLS	feed-forward NN	SVR RBF
Test R^2	0.66	0.77	0.83

Table: Cross-validated test R^2 on House.data

Ridge regression (RR)

Recall the solution of OLS

$$\boldsymbol{\beta}^* = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}.$$

When $\mathbf{X}^T\mathbf{X}$ is ill-conditioned, we can try

$$\boldsymbol{\beta}^* = (\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^T\mathbf{y},$$

where λ is an unknown parameter.

This solution corresponds to minimizing

$$||\mathbf{X}\boldsymbol{\beta} - \mathbf{y}||^2 + \lambda||\boldsymbol{\beta}||^2.$$

This objective tries to minimize MSE within a sphere of possible $\boldsymbol{\beta}$.

λ represents your believe of the observations, i.e., the larger λ is, the less believe you have. One can use cross-validation on the training data to find the optimal value of λ.

Feed-forward neural networks (NNFF)

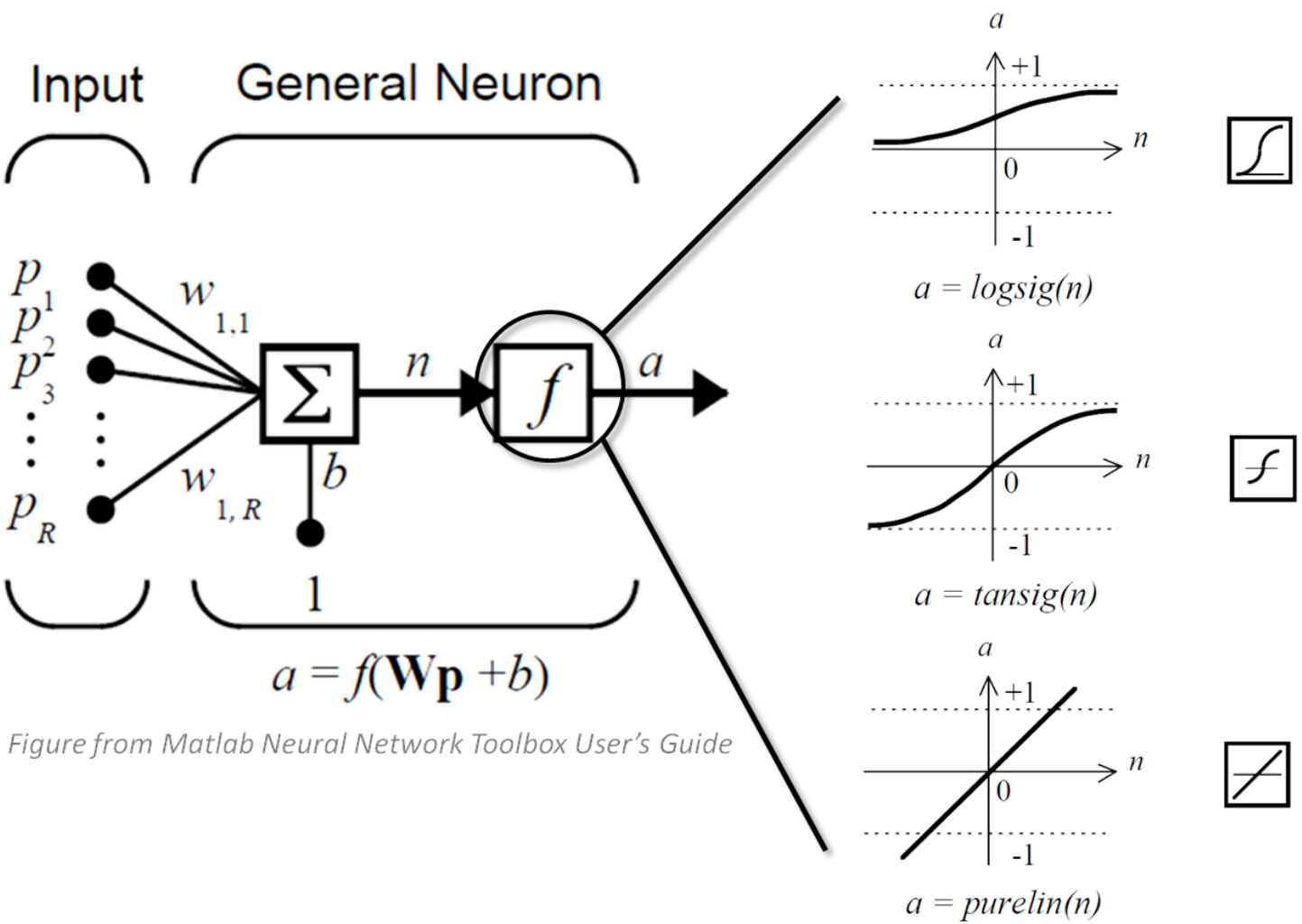
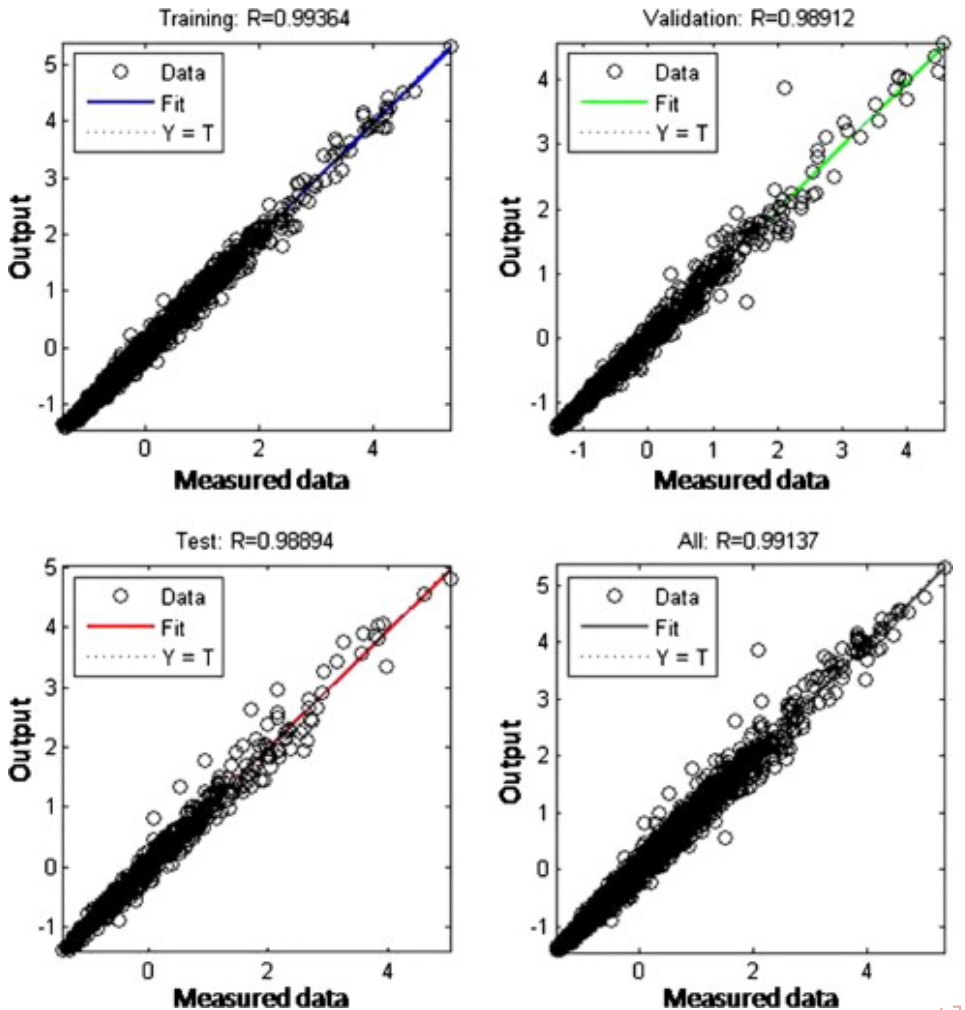


Figure from Matlab Neural Network Toolbox User's Guide

A simplest feed-forward neural net. One may add arbitrary number of layers and neurons to the model.

Feed-forward neural networks (NNFF)

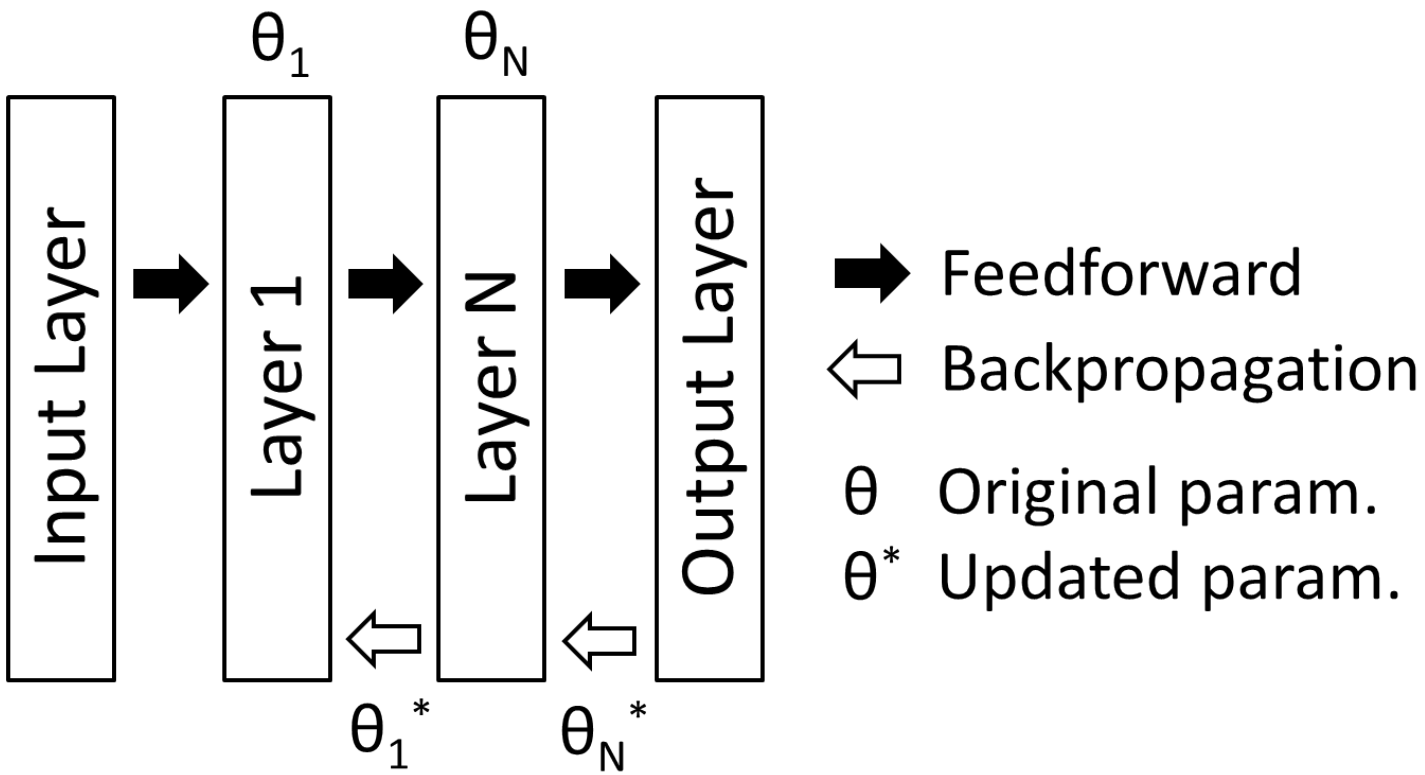
Matlab uses a portion of the training data for validation. The training (optimization) will terminate when gradient is close to zero or MSE of the validation set does not decrease for a few iterations.



Feed-forward neural networks (NNFF)

We find optimal network parameters to minimize the mean-squared error (MSE) of the testing data.

Training of an NNFF requires back-propagation: We start by minimizing MSE by w.r.t. parameters of the last hidden layer and fixing all other parameters; then move on to optimize the parameters at previous layers.



Radial-basis neural networks (NNRB)

When sample $\mathbf{y}$ are deterministic, the following NNRB model can be used for interpolation purpose:

$$y(\mathbf{x}) = \sum w_i \exp(-\gamma ||\mathbf{x} - \mathbf{x}_i||^2)$$

, where $\mathbf{w}$ are network weights.

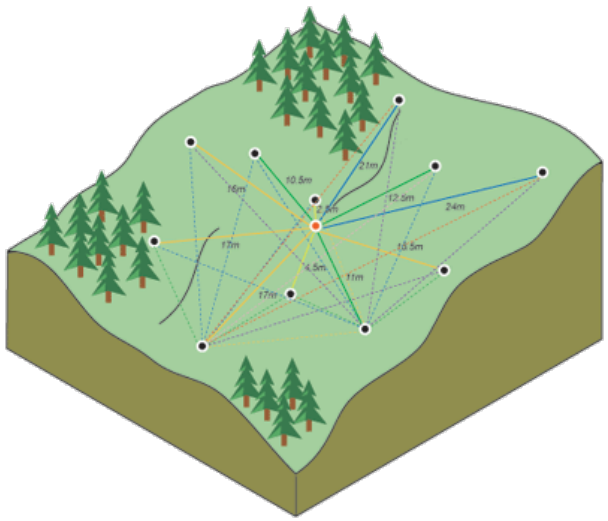
Let $r_j(\mathbf{x}) = \exp(-\gamma ||\mathbf{x} - \mathbf{x}_j||^2)$, with n samples we have $\mathbf{Rw} = \mathbf{y}$, where the matrix $\mathbf{R}$ is

$$\begin{bmatrix} r_1(\mathbf{x}_1) & r_2(\mathbf{x}_1) & \cdots & r_n(\mathbf{x}_1) \\ r_1(\mathbf{x}_2) & r_2(\mathbf{x}_2) & \cdots & r_n(\mathbf{x}_2) \\ \vdots & \vdots & \ddots & \vdots \\ r_1(\mathbf{x}_n) & r_2(\mathbf{x}_n) & \cdots & r_n(\mathbf{x}_n) \end{bmatrix}$$

It can be proved that $\mathbf{R}$ is non-singular if samples $\mathbf{X}$ are distinct, and thus the weights can be solved as $\mathbf{w} = \mathbf{R}^{-1}\mathbf{y}$.

Kriging

Kriging: A geostatistical techniques to interpolate the elevation of the landscape as a function of the geographic location at an unobserved location from observations of its value at nearby locations.



The Kriging model has

$$\hat{Y}(\mathbf{x}) = \sum_{i=1}^{n} w_i(\mathbf{x}) Y(\mathbf{x}_i),$$

where $Y(\mathbf{x})$ is a random field on $\mathbf{x}$, $w_i(\mathbf{x})$ is the weight measuring the similarity between $\mathbf{x}$ and $\mathbf{x}_i$.

Kriging

The simple Kriging model assumes $E[Y(\mathbf{x})] = 0$, which results in the model

$$\hat{y}(\mathbf{x}) = \mathbf{r}(\mathbf{x})^T \mathbf{R}^{-1} \mathbf{y},$$

where the vector $\mathbf{r}(\mathbf{x})$ measures the similarities between $\mathbf{x}$ and all samples $\mathbf{x}_i$, and the matrix $\mathbf{R}$ measures the similarities among all samples. When we use the radial-basis (Gaussian) function for measuring similarity, simple Kriging results in the same model as from NNRB.

When assuming $E[Y(\mathbf{x})] = \text{const}$, we will have the Kriging model:

$$\hat{y}(\mathbf{x}) = \hat{b} + \mathbf{r}(\mathbf{x})^T \mathbf{R}^{-1} (\mathbf{y} - \hat{b}\mathbf{1}),$$

where

$$\hat{b} = \frac{\mathbf{1}^T \mathbf{R}^{-1} \mathbf{y}}{\mathbf{1}^T \mathbf{R}^{-1} \mathbf{1}}.$$

The prediction $\hat{y}$ at any sampled $\mathbf{x}$ matches the sampled value y. Therefore Kriging is widely used for metamodeling from computer simulations (with deterministic outputs).

Support vector regression (SVR)

SVR is the regression version of the original support vector machine (SVM) for classification. The idea is to balance the training error (MSE) and model complexity to prevent over-fitting:

$$\begin{aligned} \min_{\boldsymbol{\beta},b,\boldsymbol{\xi},\boldsymbol{\xi}^*} \quad & ||\beta||^2 + C_1 \sum \xi_i + C_2 \sum \xi_i^* \\ \text{subject to} \quad & \boldsymbol{\beta}^T \mathbf{x}_i + b - y_i \leq \varepsilon + \xi_i, \\ & y_i - \boldsymbol{\beta}^T \mathbf{x}_i - b \leq \varepsilon + \xi_i^*, \\ & \xi_i,\ \xi_i^* \geq 0, \forall i. \end{aligned}$$

Similar to Kriging, with a definition of similarity, SVR can train nonlinear models. It will have the same analytical solution to Kriging when training error is forced to zero.

Summary

- Sample using Latin hypercube, $\mathbf{X}$ needs to have similar scale on each dimension;
- Always try OLS first;
- Use AIC (or BIC) for (linear) model selection;
- Use crossvalidation (within the training data) for model parameter tuning;
- OLS, Kriging, RR, SVR are easier to tune than NNFF but NNFF can be more powerful if well-tuned;
- OLS, RR, SVR can be used when data is noisy;
- Kriging (simple and ordinary) can be used when data is deterministically generated (through computer simulations);
- Choose a model (OLS, Kriging, RR, SVR, etc.) with the best test (or crossvalidation) performance. Retrain the model with all data.