# Regularisation and Bayesian techniques for learning from data

Pawel Herman

CB, KTH

# Outline

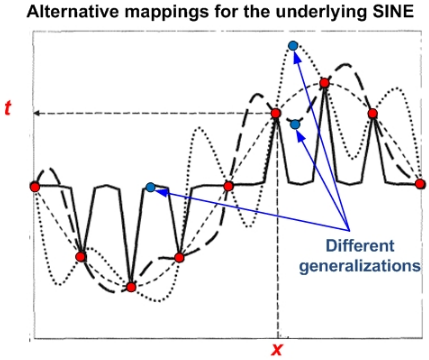
1. Generalisation in Neural Networks
2. Methods to Improve Generalisation
3. Regularisation Concept
4. Bayesian Framework

## What is generalisation?

- **Generalisation** is the capacity to apply learned knwoledge to new situations
  - the capability of a learning machine to perform well on *unseen data*
  - a tacit assumption that the test set is drawn from the same distribution as the training one

- Generalisation is affected by the training data size, complexity of the learning machine (e.g. NN) and the physical complexity of the underlying problem
- Analogy to curve-fitting problem - nonlinear mapping

# Overfitting phenomenon



- network memorises the training data (noise fitting)
- instead it should learn the underlying function
- on the other hand, the danger of underfitting/undertraining

# Statistical nature of learning (1)

- Empirical knowledge from measurements $D = \{(x_i, t_i)\}_{i=1}^{N}$
- The underlying $\mathbf{X} \to T$ can be seen as a regression problem

$$\mathbf{X} \to T: \; T = f(\mathbf{X}) + \varepsilon$$

  where $f$ is a deterministic model and $\varepsilon$ is a random error uncorrelated with the model function: $\mathbb{E}_D[\varepsilon f(\mathbf{X})] = 0$

- The expected $L_2$-norm risk of an NN estimator $F(\mathbf{x}, \mathbf{w})$ over all data $D$ can be defined as

$$R[F] = \mathbb{E}_D\left[(t - F(\mathbf{x},\mathbf{w}))^2\right]$$

# Statistical nature of learning (2)

- However, $R[F]$ is usually aproximated by empirical risk

$$R_{\mathrm{emp}}[F] = \frac{1}{N}\sum_{i=1}^{N}(t_i - F(\mathbf{x}_i, \mathbf{w}))^2$$

  - $R_{\mathrm{emp}} = 0$ does not imply generalisation and the convergence of $F \rightarrow f$,
  - $R_{\mathrm{emp}}[F(\mathbf{x}, \mathbf{w})]$ is limited to data set $D$ for which optimal **w** is found
  - $\left|R_{\mathrm{emp}} - R\right|$ decreases with the size of $D$ and increases with the number of free parameters - weights

- obvious discrepancy between $\mathbf{e_{gen}}$ and $\mathbf{e_{emp}}$

# Bias-variance dilemma (1)

- $R[F] = \mathbb{E}_D\left[(t - F(\mathbf{x}, \mathbf{w}))^2\right]$ reduces to the sum with the first term independent of NN model $F$:

$$\mathbb{E}_D\left[(t - f(\mathbf{x}))^2\right] + \boldsymbol{E_D}\left[\boldsymbol{(F(\mathbf{x}, \mathbf{w}) - f(\mathbf{x}))^2}\right]$$

- The effectiveness of NN model $F(\mathbf{x}, \mathbf{w})$ can be defined as an estimator of the regression $f = \mathbb{E}\left[t \mid \mathbf{x}\right]$ for $D$ :

$$\mathbb{E}_D\left[(F(\mathbf{x}, \mathbf{w}) - f(\mathbf{x}))^2\right]$$

# Bias-variance dilemma (2)

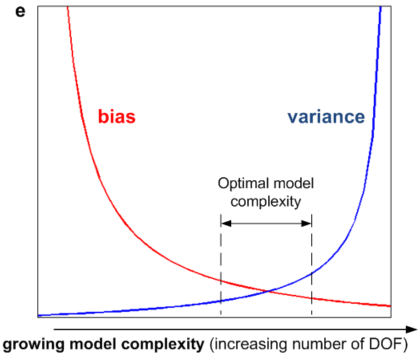
- $\mathbb{E}$ over all possible training sets $D$ helps to estimate $\mathbf{e_{gen}}$

$$\mathbb{E}_D\left[\left(F\left(\mathbf{x},\mathbf{w}\right)-f\left(\mathbf{x}\right)\right)^2\right]=$$

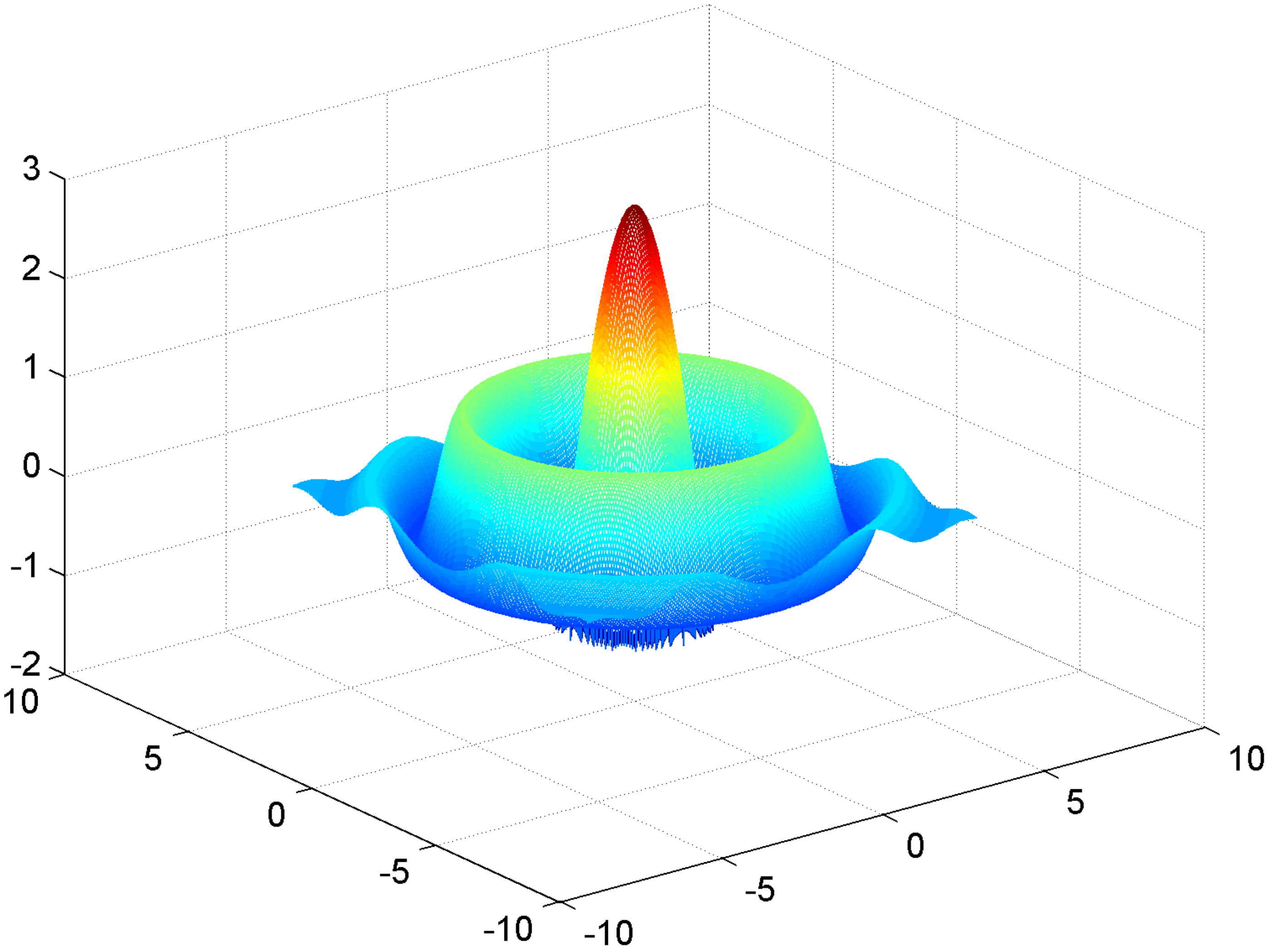
$$\left(\mathbb{E}_D\left[F\left(\mathbf{x},\mathbf{w}\right)\right]-f(\mathbf{x})\right)^2+\mathbb{E}_D\left[\left(F\left(\mathbf{x},\mathbf{w}\right)-\mathbb{E}_D\left[F\left(\mathbf{x},\mathbf{w}\right)\right]\right)^2\right]$$

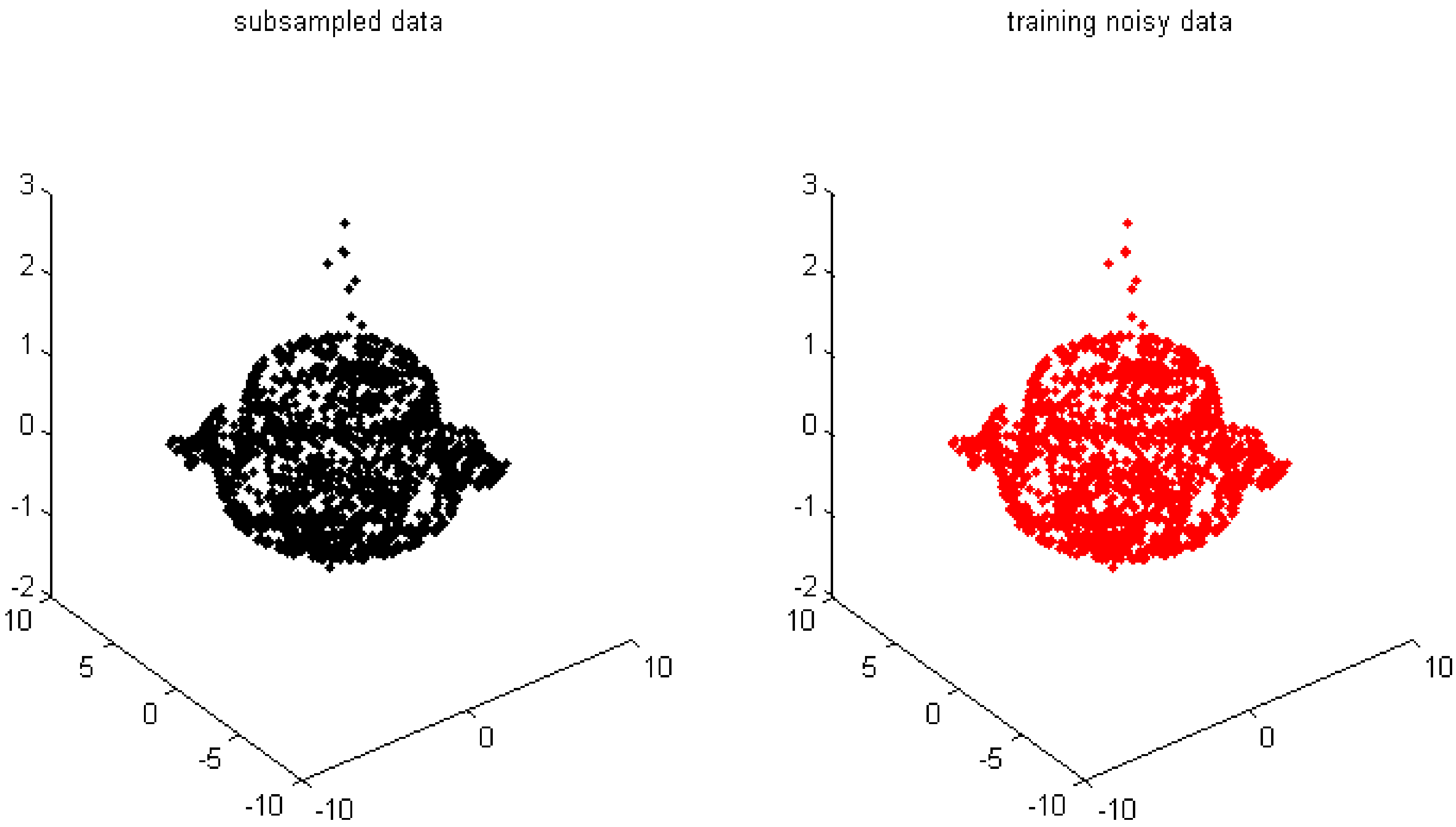
- The need to balance bias (approximation error) and variance (estimation error) on a limited data sample

# Bias-variance illustration

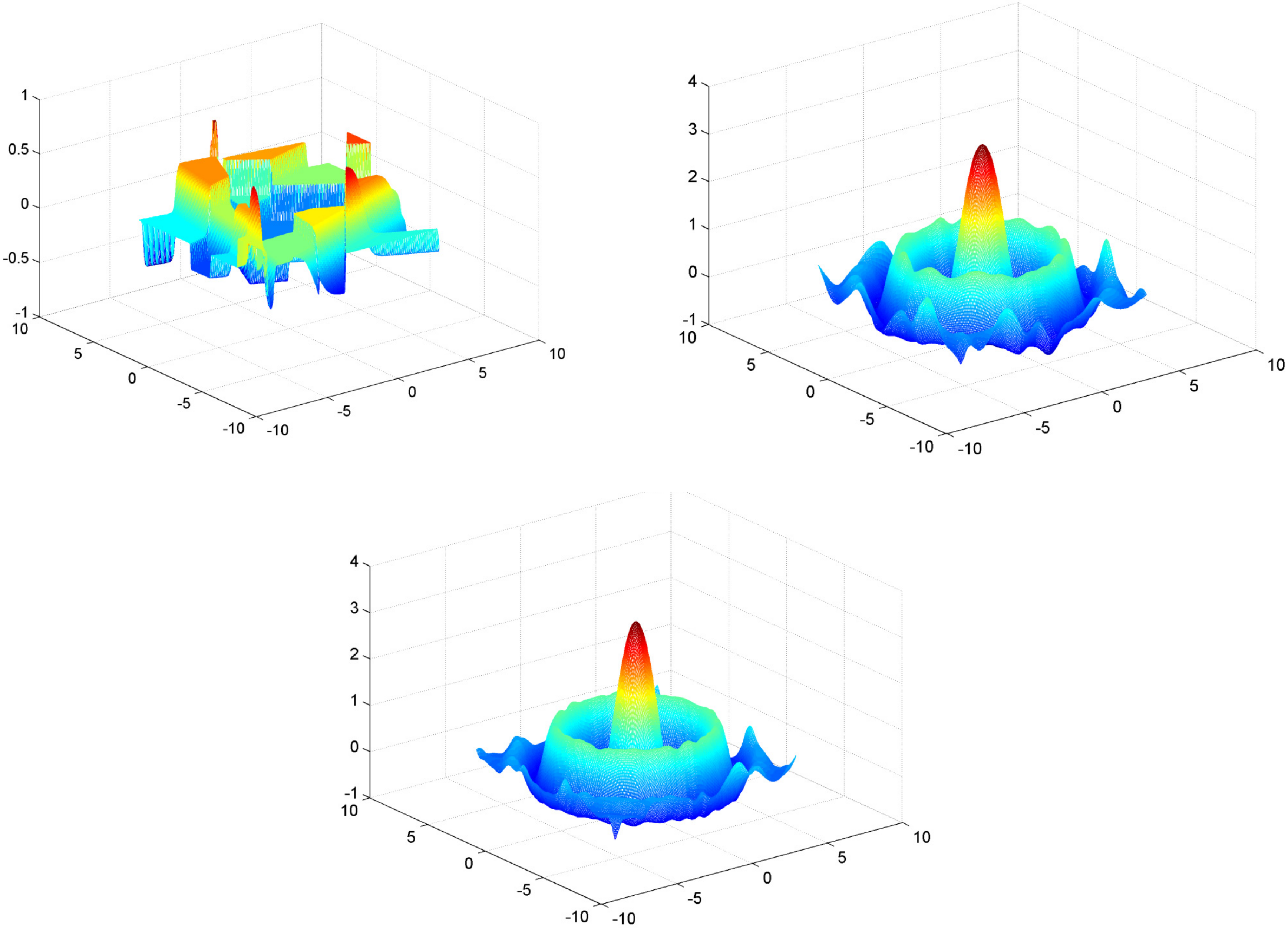


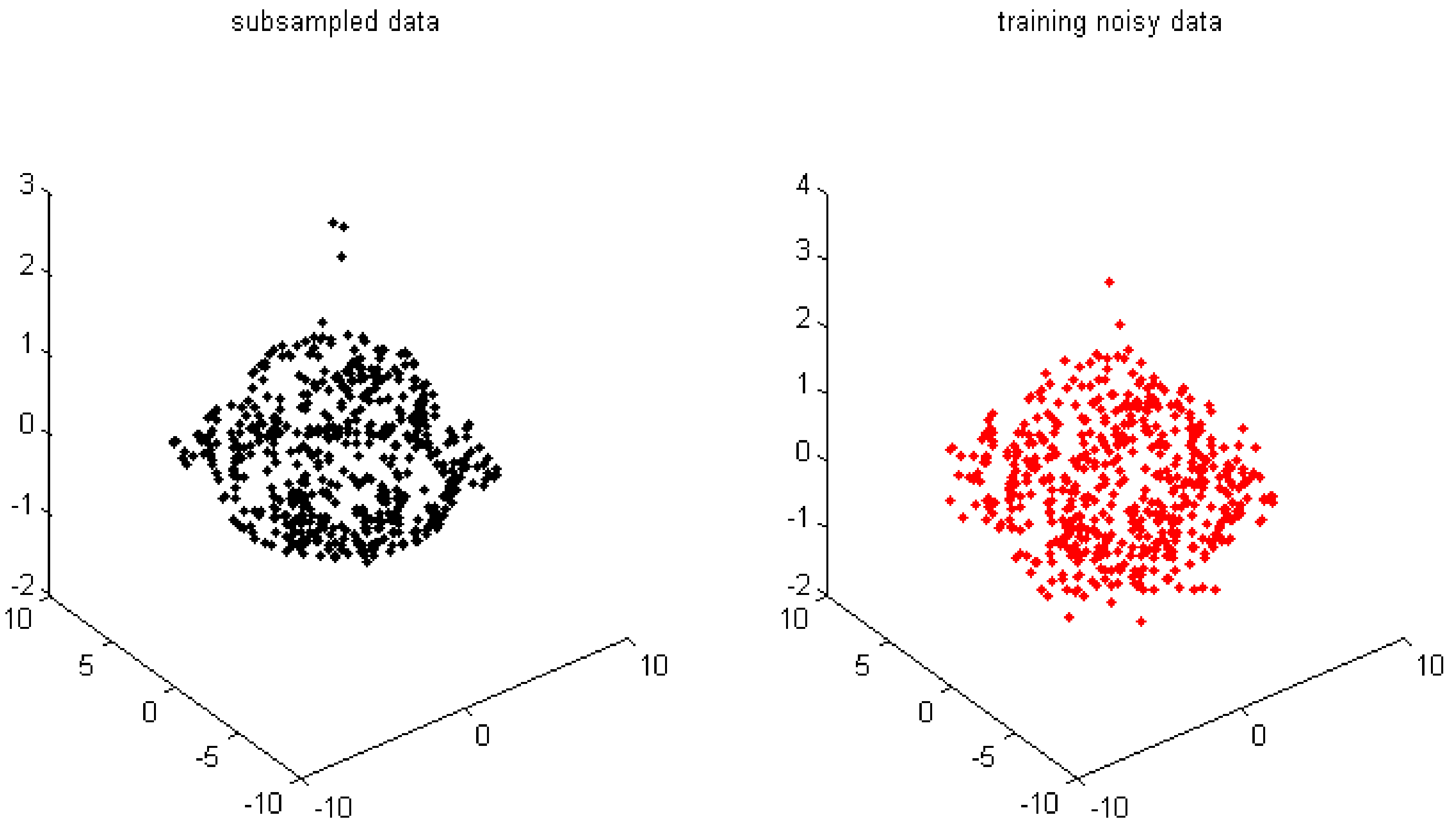
- the problem can be alleviated by increasing training data size
- otherwise, the complexity of the model has to be restricted
- for NNs, identification of the optimal network architecture


3
2
1
0
-1
-2
10
5
0
-5
-10
-10
-5
0
5
10

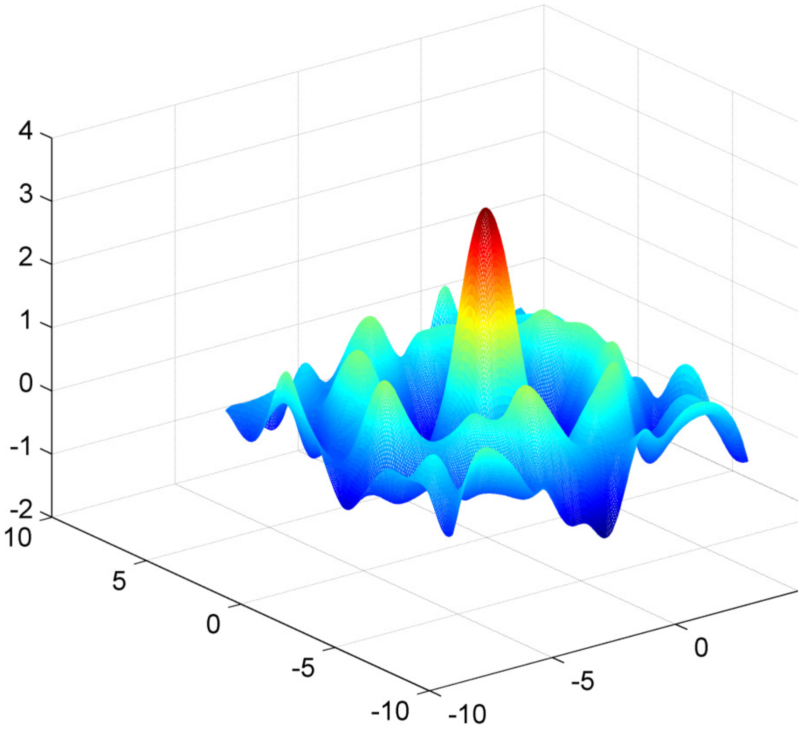


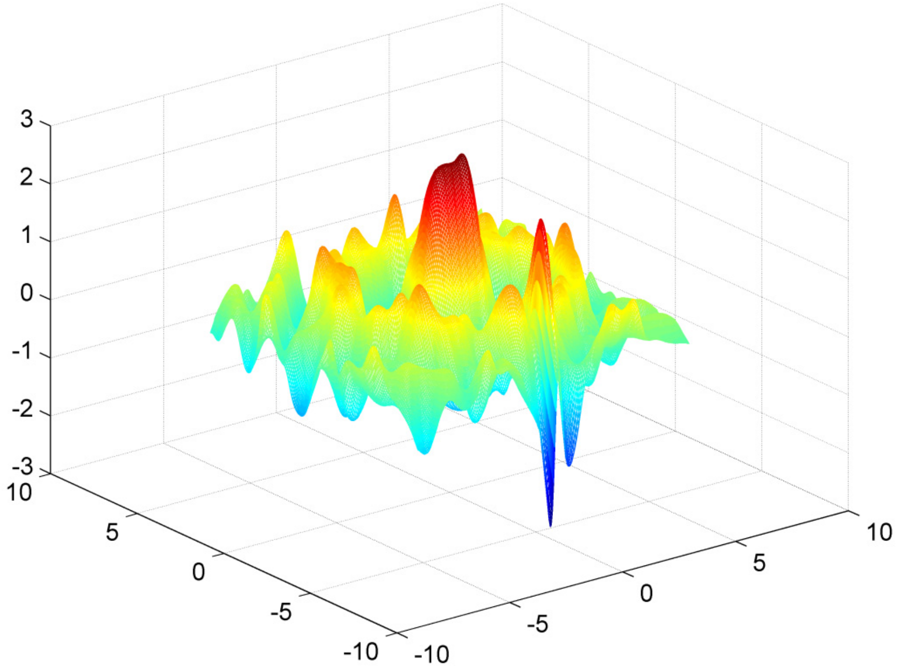
High number of training points, no noise

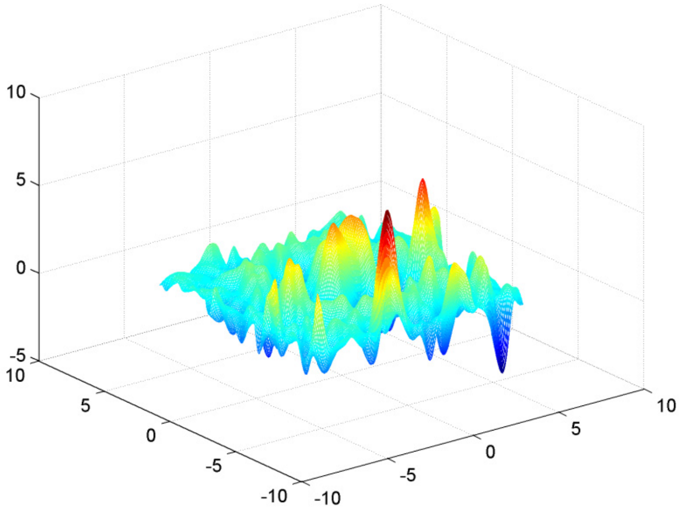

1
0.5
0
-0.5
-1
10
5
0
-5
-10
-10
-5
0
5
10
4
3
2
1
0
-1
10
5
0
-5
-10
-10
-5
0
5
10
4
3
2
1
0
-1
10
5
0
-5
-10
-10
-5
0
5
10

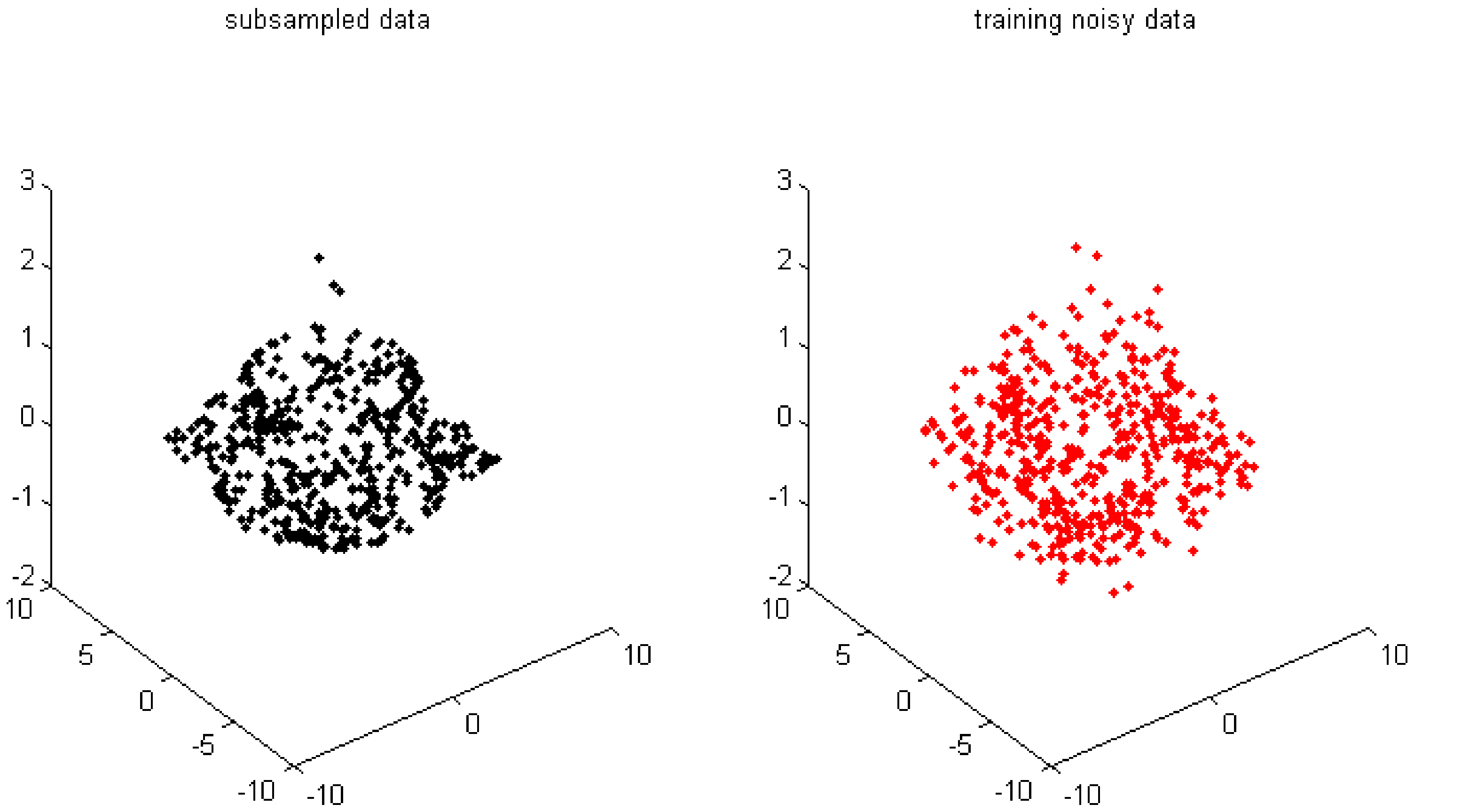


Low number of training points, high noise

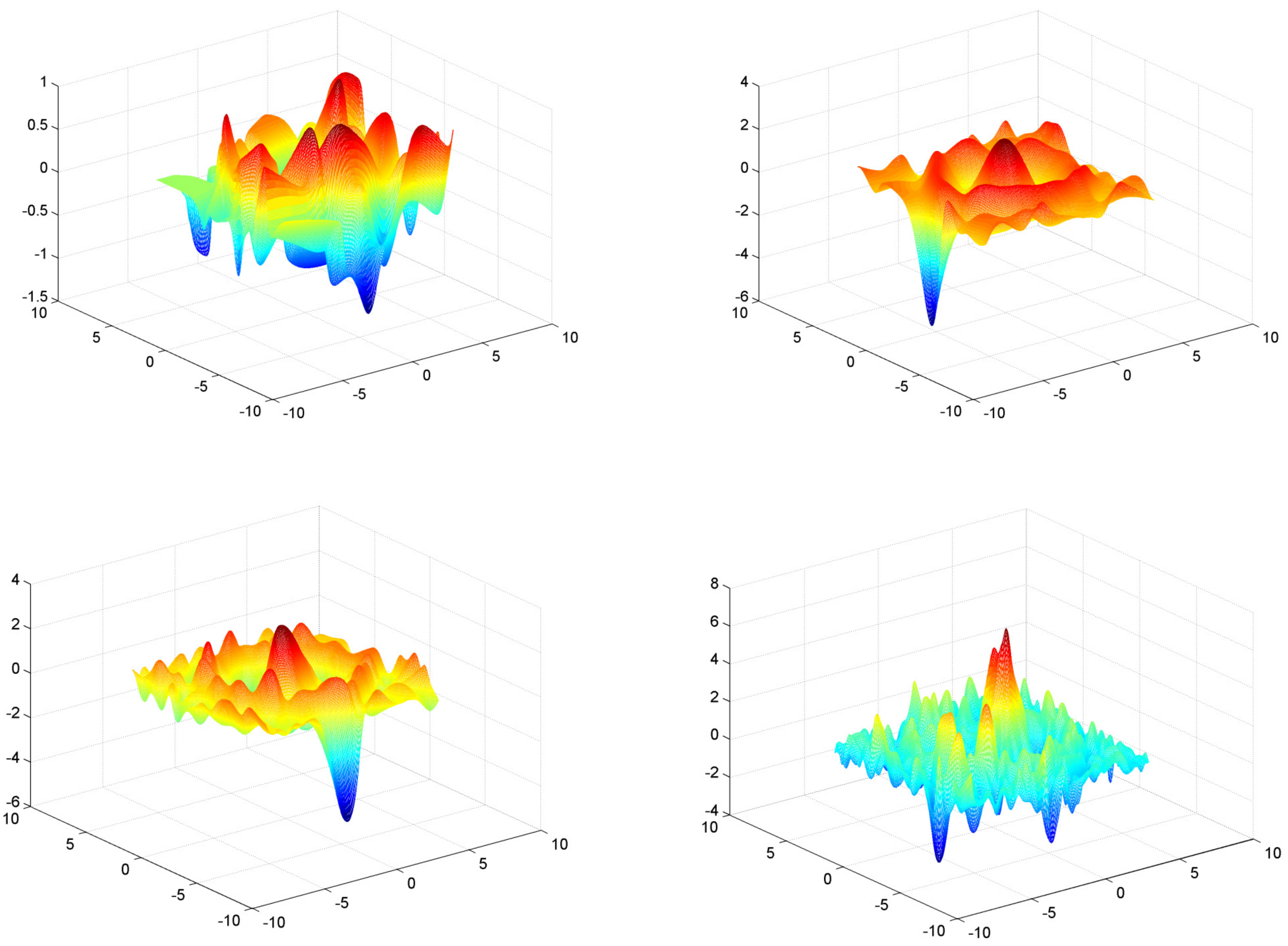

4
3
2
1
0
-1
-2
10
5
0
-5
-10
-10
-5
0


3
2
1
0
-1
-2
-3
10
5
0
-5
-10
-10
-5
0
5
10

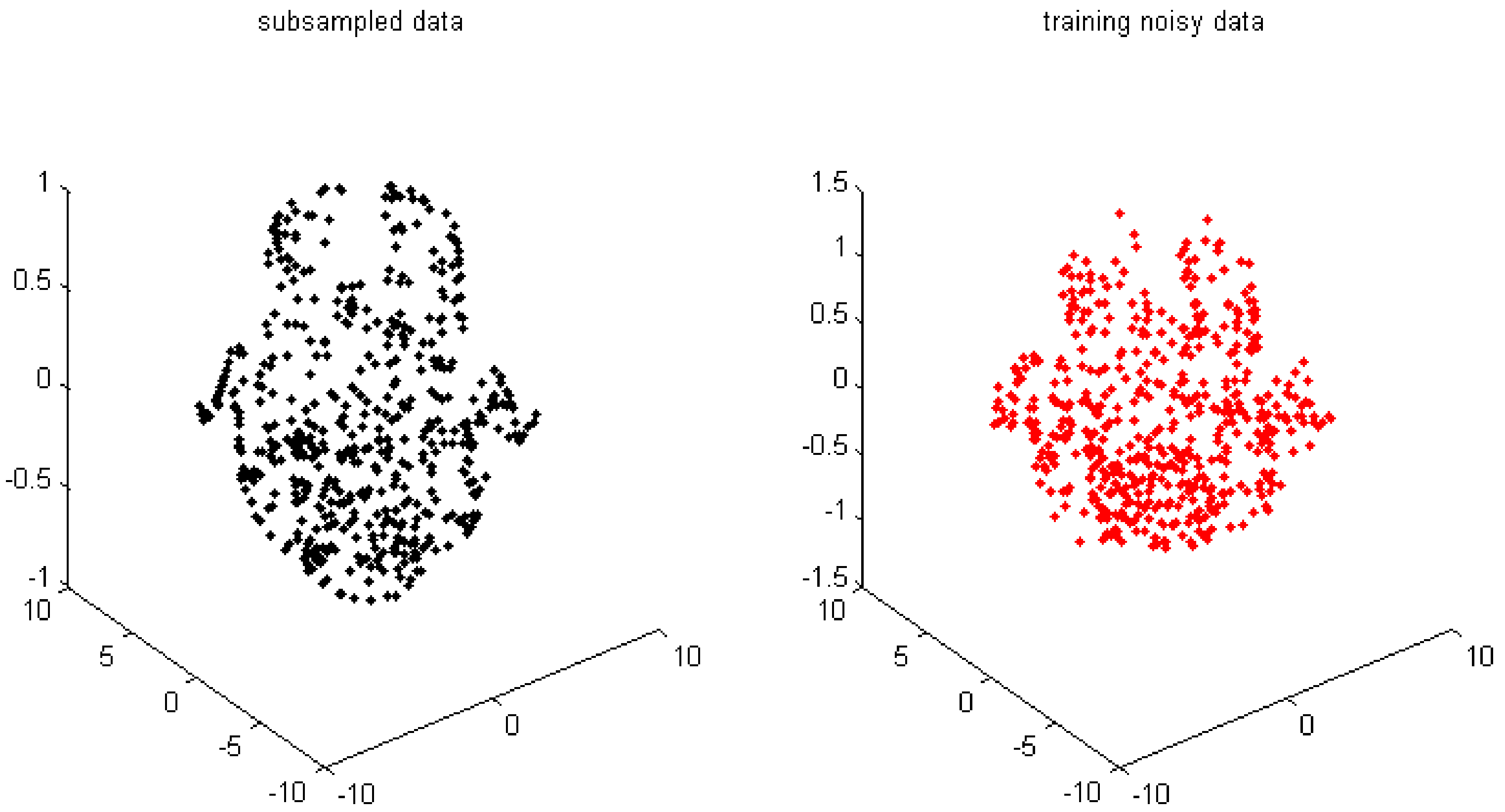

10
5
0
-5
10
5
0
-5
-10
-10
-5
0
5
10



Very low number of training points, moderate noise



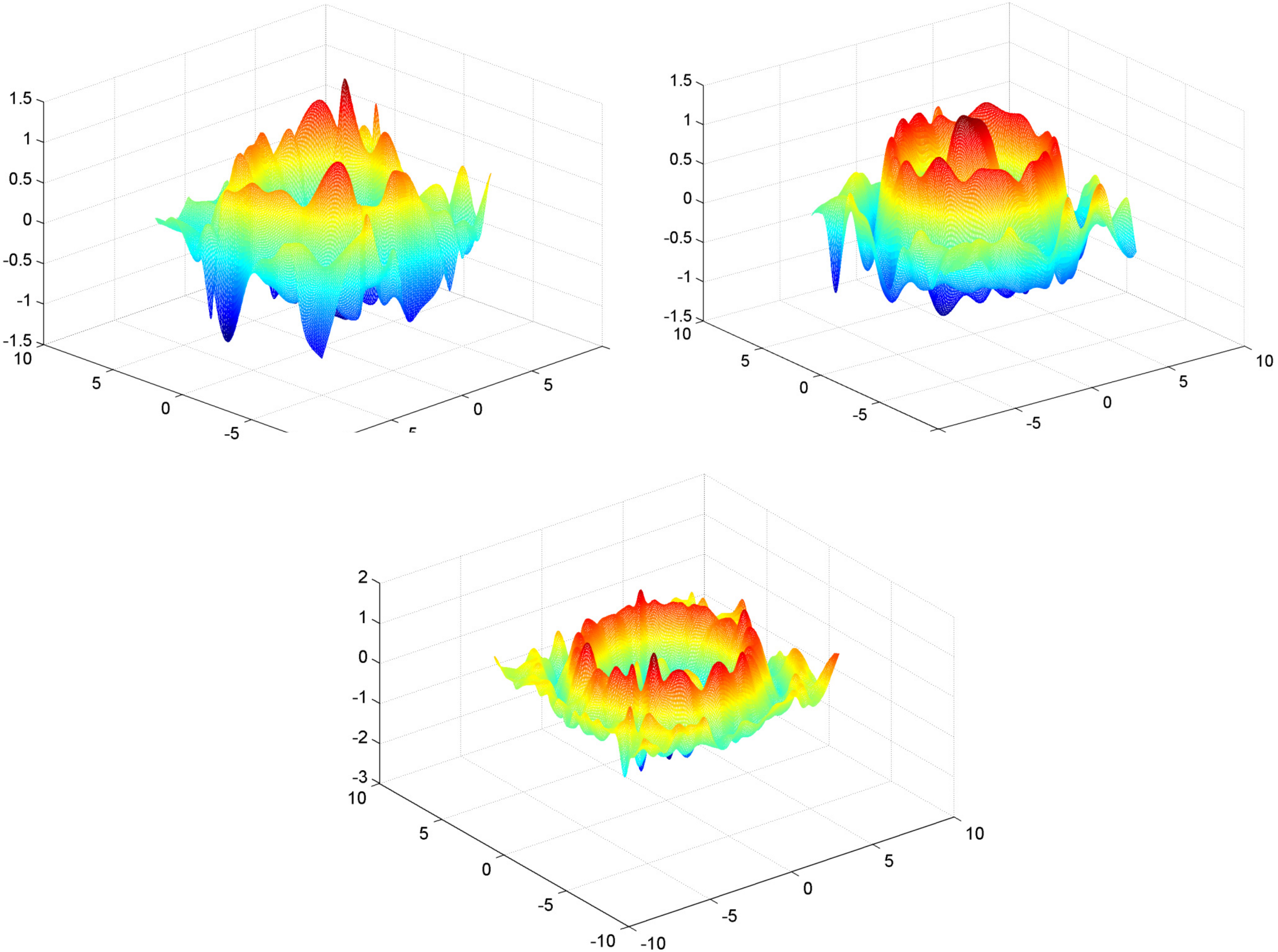
1
0.5
0
-0.5
-1
-1.5
10
5
0
-5
-10
-10
-5
0
5
10
4
2
0
-2
-4
-6
10
5
0
-5
-10
-10
-5
0
5
10
4
2
0
-2
-4
-6
10
5
0
-5
-10
-10
-5
0
5
10
8
6
4
2
0
-2
-4
10
5
0
-5
-10
-10
-5
0
5
10



subsampled data

training noisy data

Low number of training points (none in the middle), moderate noise

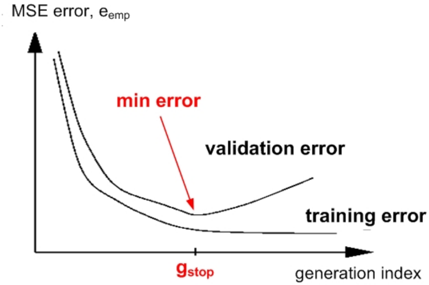

1.5
1
0.5
0
-0.5
-1
-1.5
10
5
0
-5
5
0
1.5
1
0.5
0
-0.5
-1
-1.5
10
5
0
-5
-5
0
5
10
2
1
0
-1
-2
-3
10
5
0
-5
-10
-10
-5
0
5
10

# Adding noise to the training data

- Adding a noise term (zero mean) to the input training data has been reported to enhance generalisation capabilities of MLPs
- It is expected that noise smears out data points and thus prevents the network from precise fitting original data
- In practice, each time the data point is presented for network learning new noise vector should be added
- Alternatively, sample-by-sample training with data re-ordering for every generation

# Early stopping

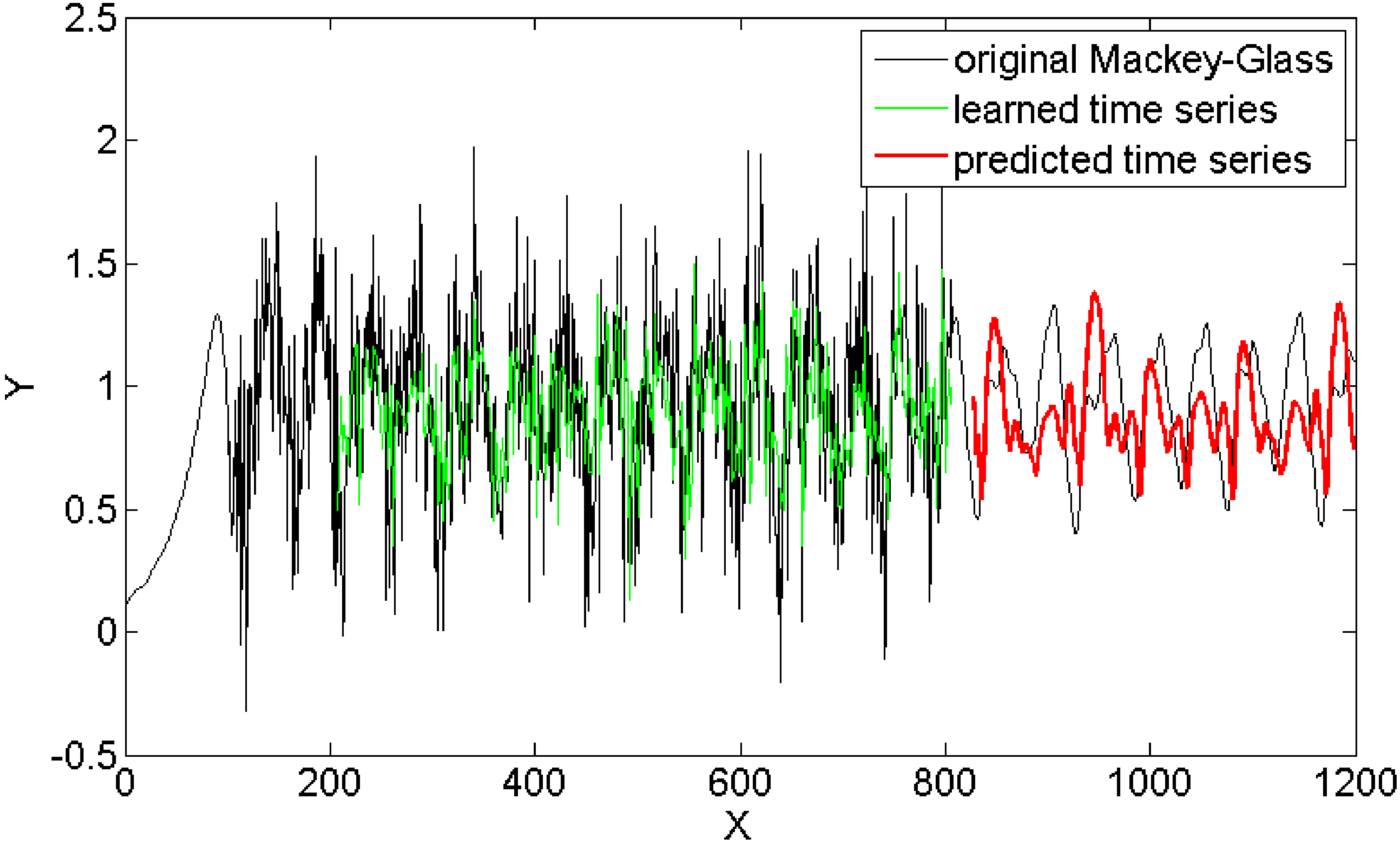
- An additional data set is required - **validation set** (split of original data)
- The network is trained with BP until the error monitored on the validation set reaches minimum (further on only noise fitting)

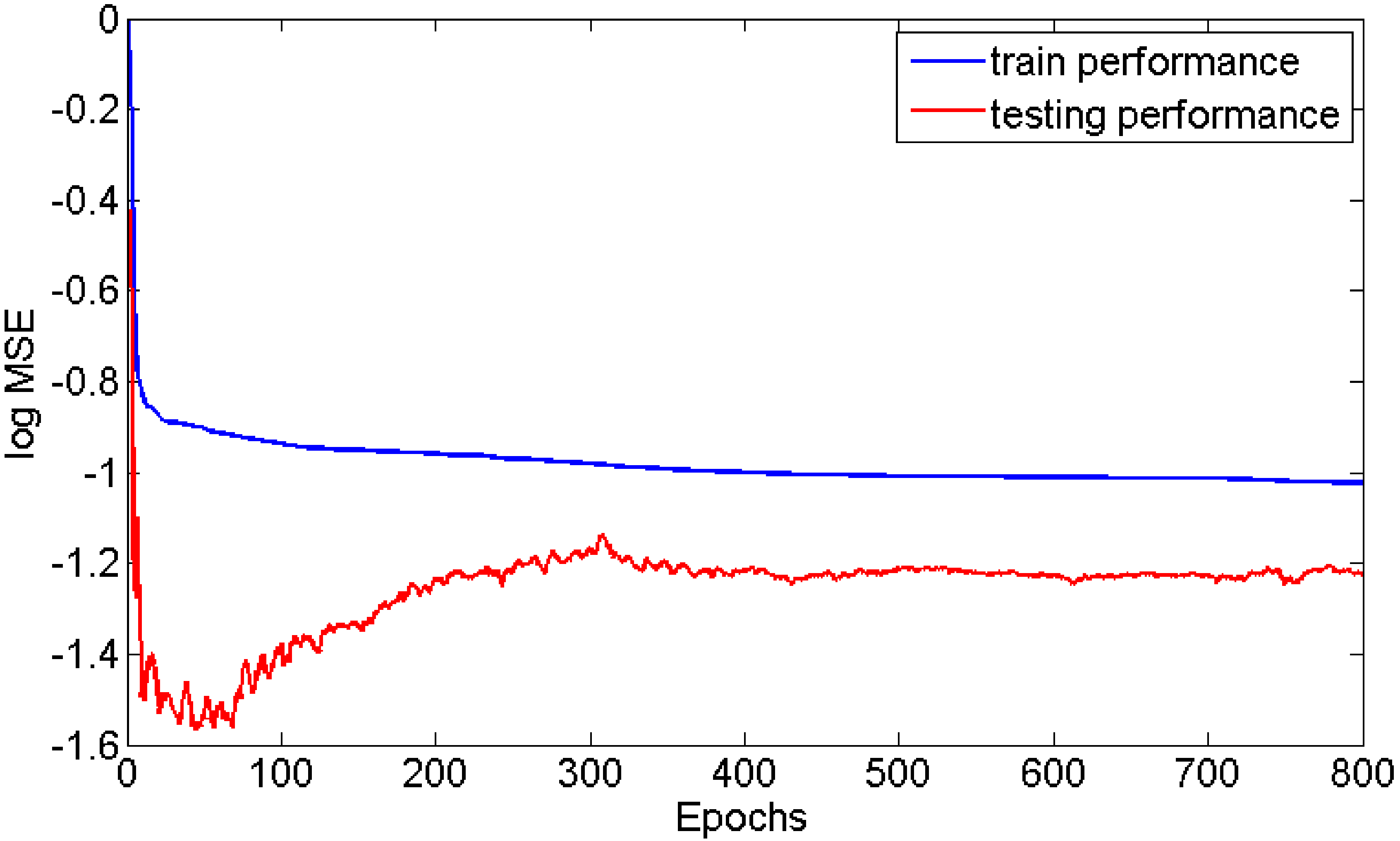


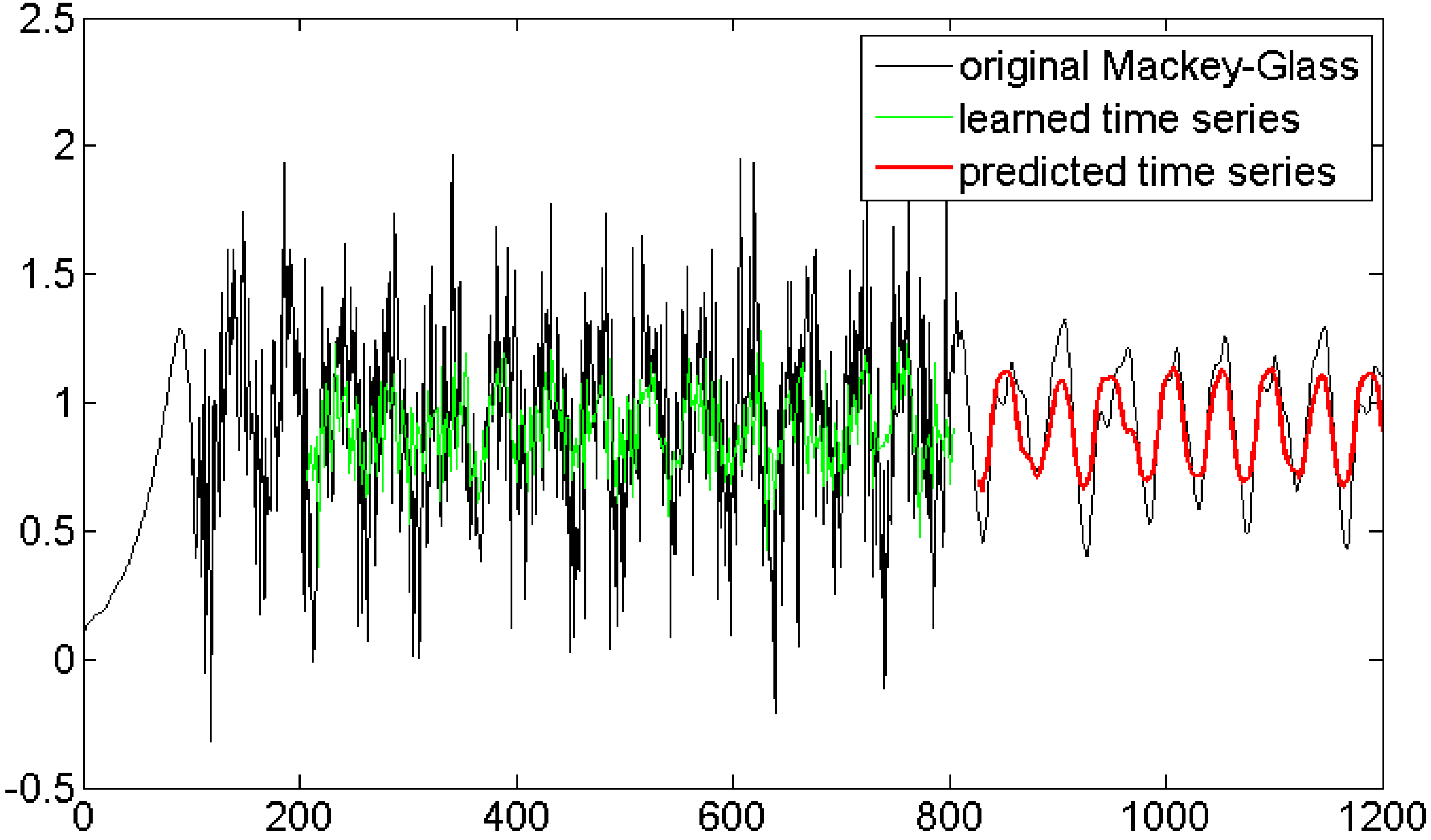
- For quadratic error, it corresponds to learning with weight decay

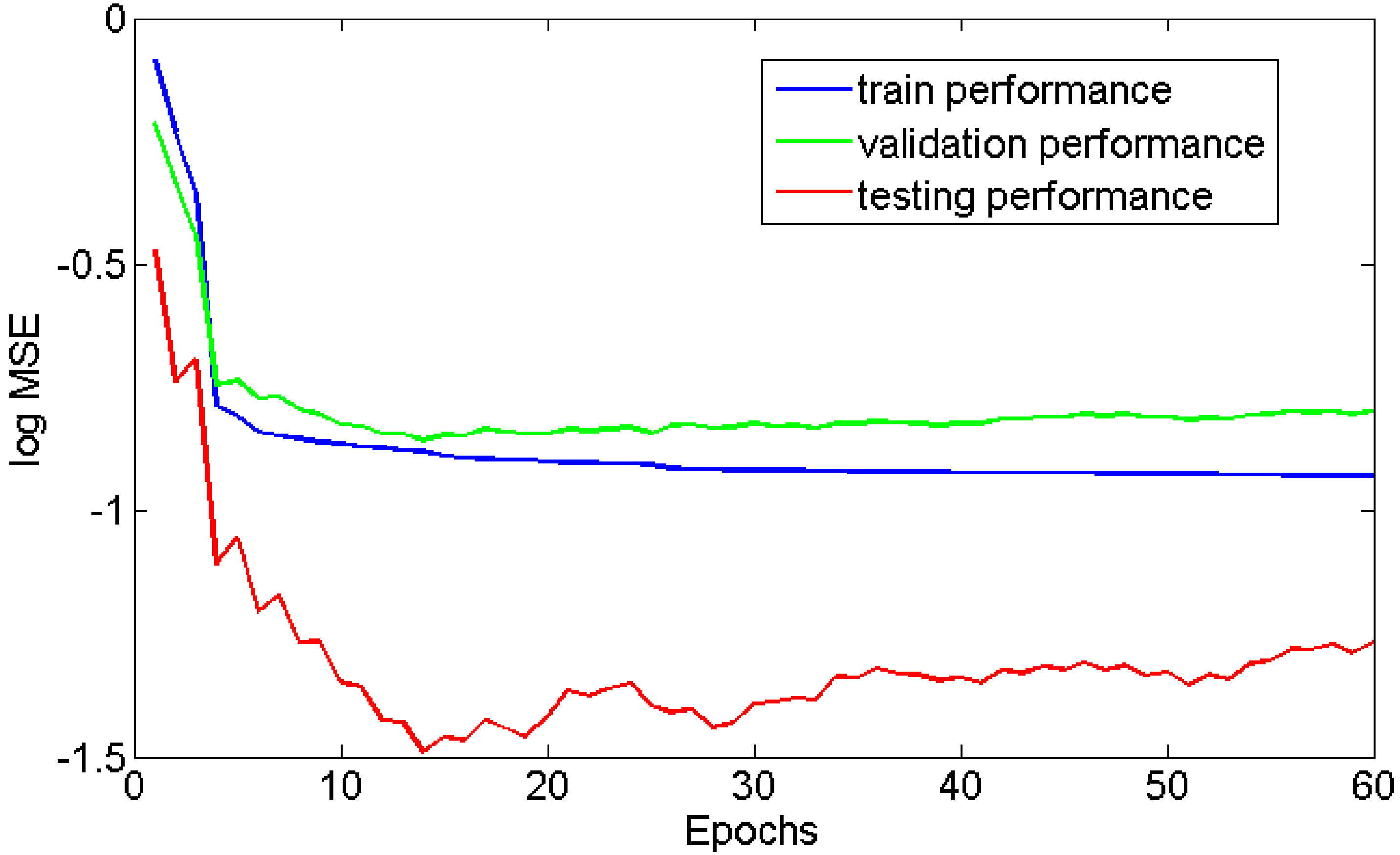
# Early stopping in practice

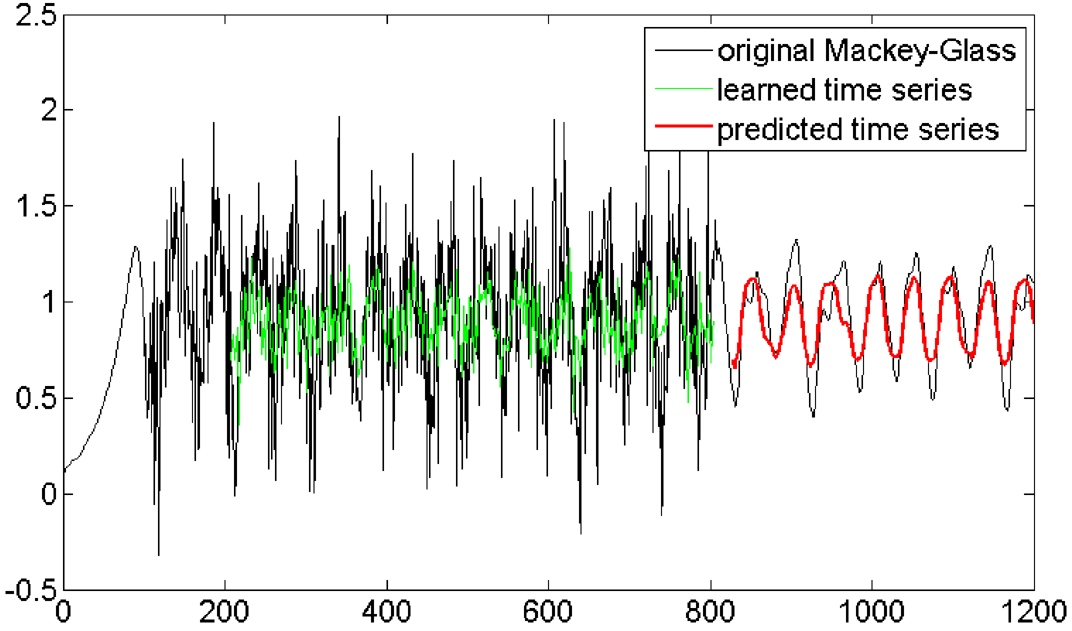
- Validation error can have multiple local minima during the learning process
- Heuristic stopping criteria
  - a certain number of successive local minima
  - generalisation loss - relative increase of the error over the minimum-so-far
- Divided opinions on the most suitable data sizes for early stopping ($30W > N > W$)
- Research on computing the stopping point based on complexity analysis - no need for a separate validation set

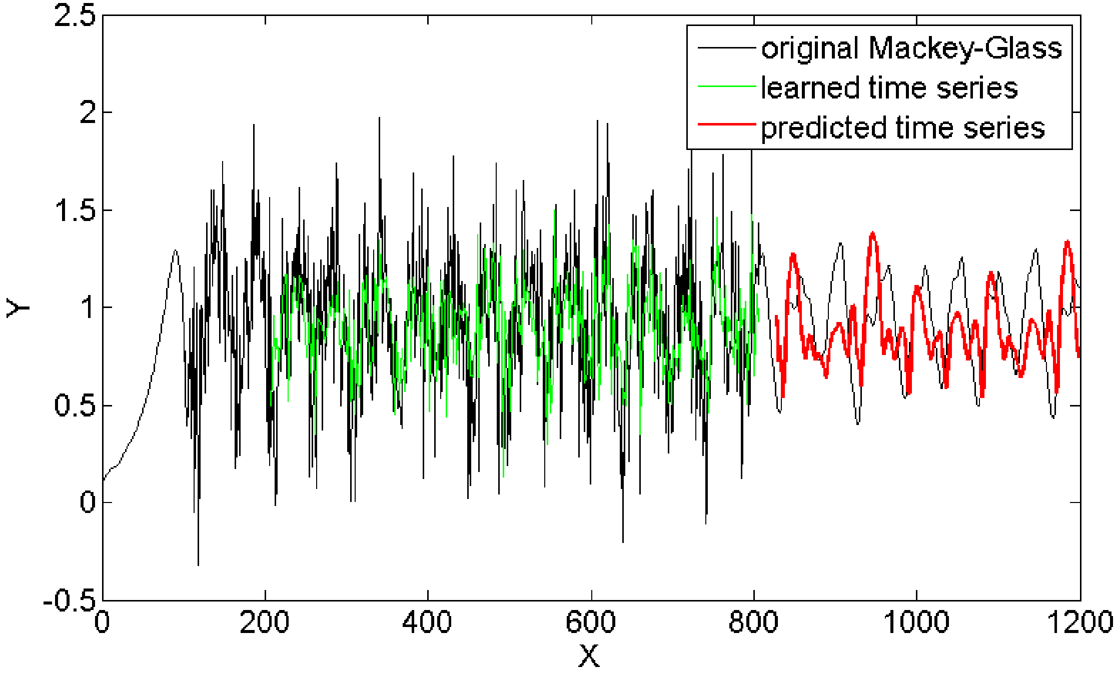

original Mackey-Glass
learned time series
predicted time series
2.5
2
1.5
1
0.5
0
-0.5
0
200
400
600
800
1000
1200
X
Y


0
-0.2
-0.4
-0.6
-0.8
-1
-1.2
-1.4
-1.6
log MSE
0
100
200
300
400
500
600
700
800
Epochs
train performance
testing performance


original Mackey-Glass
learned time series
predicted time series
2.5
2
1.5
1
0.5
0
-0.5
0
200
400
600
800
1000
1200


train performance
validation performance
testing performance
log MSE
0
-0.5
-1
-1.5
Epochs
0
10
20
30
40
50
60


original Mackey-Glass
learned time series
predicted time series
2.5
2
1.5
1
0.5
0
-0.5
0
200
400
600
800
1000
1200


original Mackey-Glass
learned time series
predicted time series
Y
2.5
2
1.5
1
0.5
0
-0.5
0
200
400
600
800
1000
1200
X

# Network growing or pruning

- Problem of network structure optimisation, primarily size
- Major approaches that avoid exhaustive search and training
  - network growing or pruning
  - network topology constructed from a set of simpler networks - network committees and mixture of experts

- In growing algorithms, new units or layers are added when some design specifications are not fulfilled (e.g. error increase)
- Network pruning starts from a large network architecture and iteratively weakens or eliminates weights based on their saliency (e.g. optimal brain surgeon algorithm)

# Model selection and verification

- Empirical assessment of generalisation capabilities
  - allows for model verification, comparison and thus selection
  - separate training and test sets - the simplest approach

- Basic ***hold-out*** (also referred to as *cross-validation*) method often relies on 3 sets and is commonly combined with early stopping
- The cost of sacrificing original data for testing (>10%) can be too high for small data sets

# Re-sampled error rates

- More economical on the data but more computationally demanding
- More robust from a statistical point of view, confidence intervals estimate
- Most popular approaches
  - bootstrap (the bias estimate) with alternative “0.632” estimator
  - *n*-fold cross-validation (*n*CV)
  - leave-one-out estimate (large variance when compared to *n*CV)

# Bootstrap

Bootstrap estimate:

$$E_{\mathrm{true}}^{boot} = E_{\mathrm{emp}}^{T} + B^{boot}$$

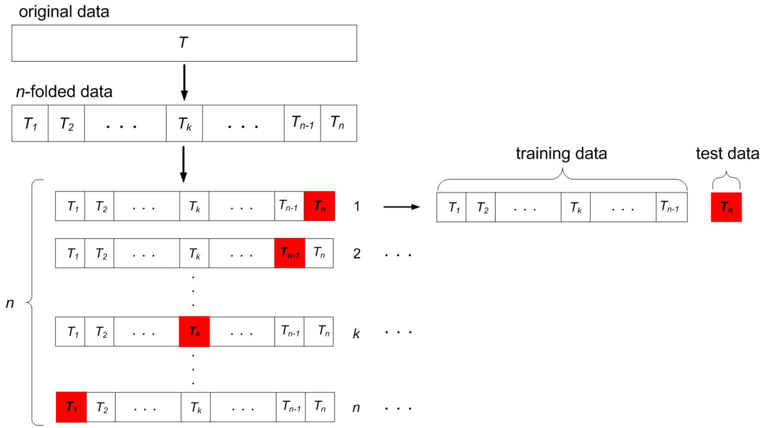
where the bias is estimated according to

$$B^{boot} = \frac{1}{K} \sum \left( E_{\mathrm{emp}}^{T_k} - E_{\mathrm{emp}}^{T_k \to T} \right)$$

Alternatively, “0.632” estimator can be used:

$$E_{\mathrm{true}}^{0.632} = 0.368\, E_{\mathrm{emp}}^{T} + 0.632\, \frac{1}{K} \sum E_{\mathrm{emp}}^{T_k \to T}$$

# *n*-fold cross-validation



original data

$T$

*n*-folded data

$T_1$ $T_2$ . . . $T_k$ . . . $T_{n-1}$ $T_n$

$n$

$T_1$ $T_2$ . . . $T_k$ . . . $T_{n-1}$ $T_n$ 1 → $T_1$ $T_2$ . . . $T_k$ . . . $T_{n-1}$ $T_n$

training data

test data

$T_1$ $T_2$ . . . $T_k$ . . . $T_{n-1}$ $T_n$ 2 . . .

$T_1$ $T_2$ . . . $T_k$ . . . $T_{n-1}$ $T_n$ $k$ . . .

$T_1$ $T_2$ . . . $T_k$ . . . $T_{n-1}$ $T_n$ $n$ . . .

$$E_{\text{true}}^{\text{CV}} = \frac{1}{n}\sum E_{\text{emp}}^{T\setminus T_k \rightarrow T_k}$$

# The concept of regularisation (1)

- Regularisation as an approach to controlling the complexity of the model
    - constraining an ill-posed problem (existence, uniqueness and continuity)
    - striking bias-variance balance (SRM)

- Penalised learning (penalty term in the error function)

$$\tilde{E} = E + \lambda \Omega$$

    - trade-off controlled by the regularisation parameter $\lambda$
    - smooth stabilisation of the solution due to the complexity term
    - in classical “penalised ridging”, $\Omega = \frac{\partial^n Y}{\partial \mathbf{w}^n} \left( \|\mathbf{D}Y\|^2 \right)$

# The concept of regularisation (2)

- Why is smoothness promoted (non-smooth solutions are penalised)
  - heuristic understanding of smooth mapping in regression and classification
  - smooth stabilisation encourages continuity (see *ill-posed problems*)
  - most physical processes are described by smooth functionals
- Theoretical justification of regularisation -> the need to impose Occam's razor on the solution
- The type of regularisation term can be determined using prior knowledge

# Forms of complexity penalty term (1)

- weight decay

$$\Omega = \|\mathbf{w}\|^2 = \sum w_i^2$$

  - a simple approach to forcing some weights (excess weights) to 0
  - limiting the risk of overfitting due to high likelihood of taking on arbitrary values by excess weights
  - reducing large curvatures in the mapping (linearity of the central region of a sigmoidal activation function)

- weight elimination

$$\Omega = \sum \frac{(w_i/w_0)^2}{1+(w_i/w_0)^2}$$

  - $w_0$ is a parameter
  - variable selection algorithm
  - it favours few large weights over a number of small ones

# Forms of complexity penalty term (2)

- curvature-driven smoothing

$$\Omega = \sum_{in} \sum_{out} \left( \frac{\partial^2 y_{out}}{\partial x_{in}^2} \right)^2$$

  - direct approach to penalising high curvature
  - derivatives can be obtained via extension of back-propagation

- approximate smoother for MLP with a single hidden layer and a single output neuron

$$\Omega = \sum w_{out_j}^2 \left\| \mathbf{w}_j \right\|^p$$

  - a more accurate method than weigh decay and weight elimination
  - distinguishes the roles of weights in the hidden and the output layer, captures the interactions between them

# Fundamentals of Bayesian inference

- Statistical inference of the probability that a given hypothesis may be true based on collected (accumulated) evidence or observations
- Framework for adjusting probabilities stems from *Bayes' theorem*

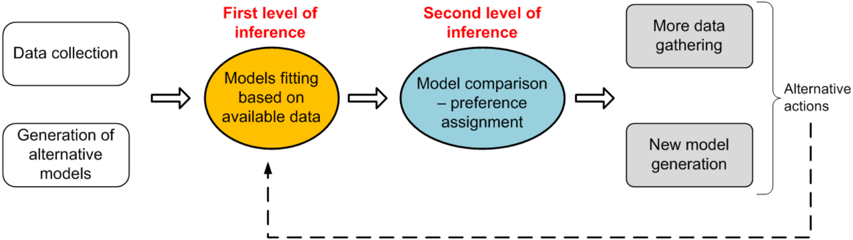
$$P(\mathscr{H}_i \mid D) = \frac{P(\mathscr{H}_i)\, P(D \mid \mathscr{H}_i)}{P(D)}$$

  - $D$ refers to data and serves as *source of **evidence*** in this framework
  - $P(D \mid \mathscr{H}_i)$ is the conditional probability of seeing the evidence (data) $D$ if the hypothesis $\mathscr{H}_i$ is true - ***evidence for the model*** $\mathscr{H}_i$
  - $P(D)$ serves as ***marginal probability*** of $D$ - a priori probability of seeing the new data $D$ under all posible hypotheses (does not depend on a particular model)
  - $P(\mathscr{H}_i \mid D)$ is the ***posterior probability*** of $\mathscr{H}_i$ given $D$ (assigned after the evidence is taken into account)

# Philosophy of Bayesian inference

- All quantities are treated as random variables, e.g. parameters, data, model
- A prior distribution over the unknown parameters (model $\mathscr{H}$ description) captures our beliefs about the situation before we see the data
- Once the data is observed, Bayes' rule produces a **posterior distribution** for these parameters - prior and evidence is taken into account
- Then predictive distributions for future observations can be computed

# Bayesian inference in a scientific process


Data collection
Generation of alternative models
First level of inference
Models fitting based on available data
Second level of inference
Model comparison – preference assignment
More data gathering
New model generation
Alternative actions

- model fitting (weights in ANNs) based on likelihood function and priors (1st level)
- model comparison by evaluating the evidence (2nd level)

# Bayesian learning of network weights - model fitting

- Instead of minimising the error function as in the maximum likelihood approach, $(\max_{\mathbf{w}} L(\mathbf{w}) = \max_{\mathbf{w}}\{P(D|\,\mathbf{w})\})$, the Bayesian approach looks at pdf over **w** space
- For the given architecture (defining $\mathcal{H}_i$), a prior distribution $P(\mathbf{w})$ is assumed and when the data has been observed the posterior probability is evaluated

$$P(\mathbf{w}|\,D,\mathcal{H}_i) = \frac{P(\mathbf{w}|\mathcal{H}_i)\,P(D|\,\mathbf{w},\mathcal{H}_i)}{P(D|\mathcal{H}_i)} = \frac{\text{prior x likelihood}}{\text{evidence}}$$

- The normalisation factor, $P(D|\mathcal{H}_i)$, does not depend on **w** and thus it is not taken into account at this stage of inference (first level) for the given architecture

# Model fitting - distribution of weights and targets

- The most common Gaussian prior for weights $P(\mathbf{w})$ that encourages smooth mappings (with $E_\mathbf{w} = \|\mathbf{w}\|^2 = \sum w_i^2$)

$$P(\mathbf{w}) = \frac{\exp\left(-\alpha E_\mathbf{w}\right)}{Z_\mathbf{w}(\alpha)} = \frac{\exp\left(-\alpha \|\mathbf{w}\|^2\right)}{\int \exp(-\alpha \|\mathbf{w}\|^2) d\mathbf{w}}$$

- The likelihood function $P\left(D|\,\mathbf{w}\right)$ can also be expressed in an exponential form with the error function $E_D = \sum_{i=1}^{N} \left(y\left(x_i, \mathbf{w}\right) - t_i\right)^2$

$$P\left(D|\,\mathbf{w}\right) = \frac{\exp\left(-\beta E_D\right)}{Z_D(\beta)}$$

- This corresponds to the Gaussian noise model of the target data, $\mathrm{pdf}(t \mid x, \mathbf{w}) \propto \exp\left(-\left(y\left(x, \mathbf{w}\right) - t\right)^2\right)$
- $\alpha, \beta$ are hyperparameters that describe the distribution of network parameters

# Posterior distribution of weights

$$P(\mathbf{w}|\ D) \propto P(\mathbf{w})\ P(D|\ \mathbf{w}) = \frac{\exp(-\alpha E_{\mathbf{w}})}{Z_{\mathbf{w}}(\alpha)} * \frac{\exp(-\beta E_D)}{Z_D(\beta)} = \frac{\exp(-M(\mathbf{w}))}{Z_M(\alpha,\beta)}$$

with $M(\mathbf{w}) = \alpha E_{\mathbf{w}} + \beta E_D = \alpha \sum w_j^2 + \beta \sum (y(x_i,\mathbf{w}) - t_i)^2$

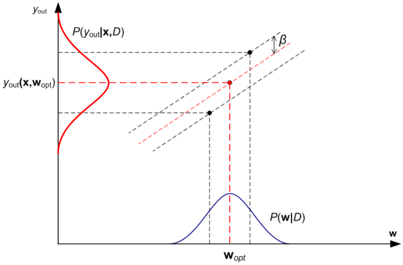
- optimal $\mathbf{w}_{\mathrm{opt}}$ should minimise $M$, which has analogous form to the square error function with weight decay regularisation
- it can be found in the course of gradient descent (for known $\alpha, \beta$)
- Bayesian framework allows not only for identification of $\mathbf{w}_{\mathrm{opt}}$, but also posteriori distribution of weights $P(\mathbf{w}|\ D)$

# Distribution of network ouputs

- In consequence, the distribution of network outputs follows

$$P(y_{\mathrm{out}} \mid \mathbf{x},\, D) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left( -\frac{\left(y_{out} - y(\mathbf{x}, \mathbf{w}_{\mathrm{opt}})\right)^2}{2\sigma^2} \right)$$

- $\sigma$ depends on $\beta$ and the width of $P\,(\mathbf{w}|\; D)$


$y_{out}$
$P(y_{out}|\mathbf{x},D)$
$\beta$
$y_{out}(\mathbf{x},\mathbf{w}_{opt})$
$P(\mathbf{w}|D)$
$\mathbf{w}$
$\mathbf{w}_{opt}$

# Evidence framework for $\alpha$ and $\beta$

- Hyperparameters are found by maximising their posterior distribution $P(\alpha,\beta \mid D) = \dfrac{P(D \mid \alpha,\beta)\, P(\alpha,\beta)}{P(D)}$
- Maximisation of $P(D \mid \alpha,\beta)$ - *the evidence for* $\alpha,\beta$
- Calculations lead to an approximate solution that can be evaluated iteratively

$$\alpha^{(n+1)} = \frac{\gamma^{(n)}}{2\left\|\mathbf{w}^{(n)}\right\|^2}, \quad \beta^{(n+1)} = \frac{N-\gamma^{(n)}}{2\sum_{i=1}^{N}\left(y\left(x_i,\mathbf{w}^{(n)}\right)-t_i\right)^2}$$

- $\gamma$ is the effective number of weights (*well-determined parameters*) estimated based on the data
- For very large data sets $\gamma$ is the number of all weights

# Model comparison (1)

- Posterior probabilities of various models $\mathscr{H}_i$

$$P(\mathscr{H}_i \mid D) = \frac{P(\mathscr{H}_i)\, P(D \mid \mathscr{H}_i)}{P(D)}$$

- It normally amounts to comparing evidence, $P(D \mid \mathscr{H}_i)$
- $P(D \mid \mathscr{H}_i)$ can be approximated around $\mathbf{w}_{\mathrm{opt}}$ as follows

$$\int P(D \mid \mathbf{w}, \mathscr{H}_i)\, P(\mathbf{w} \mid \mathscr{H}_i)\, d\mathbf{w} \backsimeq \left\{P\left(D \mid \mathbf{w}_{\mathrm{opt}}, \mathscr{H}_i\right)\right\}\left\{P\left(\mathbf{w}_{\mathrm{opt}} \mid \mathscr{H}_i\right) \triangle\mathbf{w}_{\mathrm{posterior}}\right\}$$

- For the uniform distribution of the prior $P\left(\mathbf{w}_{\mathrm{opt}} \mid \mathscr{H}_i\right) = \dfrac{1}{\triangle\mathbf{w}_{\mathrm{prior}}}$

$$\text{Occam factor} = \frac{\triangle\mathbf{w}_{\mathrm{posterior}}}{\triangle\mathbf{w}_{\mathrm{prior}}}$$

# Model comparison (2)

- The model with the highest evidence strikes the balance between the best likelihood fit and a large Occam factor (low complexity)
- Occam factor reflects the penalty for the model with a given posterior distribution of weights
- The evidence can be estimated more precisely using the calculations at the first inference level
- However, the correlation between the evidence and the generalisation capability is not straightforward
  - noisy nature of the test error
  - evidence framework provides only a relative ranking
  - maybe, only poor models are considered
  - inaccuraccies in the estimation of the evidence

# Practical approach to Bayesian regularisation

1. Initialisation of values for $\alpha$ and $\beta$.
2. Initialisation of weights from prior distributions.
3. Network training using gradient descent to minimise $M(\mathbf{w})$.
4. Update of $\alpha$ and $\beta$ every few iterations.
5. Steps 1-4 can be repeated fo different initial weights.
6. The algorithm can be repeated for different network models $\mathscr{H}_i$ to choose the model with the largest evidence.

# Practical implication of Bayesian methods

- An intuitive interpretation of regularisation
- There is no need for validation data for parameter selection
- Computationally effective handling of higher number of parameters
- Confidence intervals can be used with the network predictions
- Scope for comparing different network models using only training data
- Bayesian methods provide an objective framework to deal with complexity issues

# Summary and practical hints

1. What is generalisation and overfitting phenomena?
2. How to prevent from overfitting and boost generalisation?
3. What does regularisation contribute wrt. generalisation?
4. How can the Bayesian framework benefit the process of NN design?
5. What are the practical Bayesian techniques and their implications?

# References

- Christopher M. Bishop, *Neural Networks for Pattern Recognition*, 1995
- David J.C. MacKay, A Practical Bayesian Framework for Backpropagation Networks, *Neural Computation*, vol.4, 1992
- Voijslav Kecman, *Learning and Soft Computing*
- Simon Haykin, *Neural Networks. A Comprehensive Foundation*, 2nd edition, 1999