DESIGN AND REGULARIZATION OF NEURAL NETWORKS: THE OPTIMAL USE OF A VALIDATION SET

JAN LARSEN

The Computational Neural Network Center
Section for Digital Signal Processing
Department for Mathematical Modelling, B321
Technical University of Denmark
DK-2800 Lyngby, Denmark

☎ (+45) 45 25 39 08, (+45) 45 25 39 23
Fax: (+45) 45 87 25 99
Email: jl@imm.dtu.dk
Anonymous FTP: eivind.imm.dtu.dk
WWW: http://eivind.imm.dtu.dk

MOTIVATION

- Need a better method than – more or less – systematic search to determine regularization parameters.
- Can we avoid the strong statistical assumptions entering OBD/OBS for architecture optimization?

HIGHLIGHTS

Idea

Use an empirical approach by optimizing the architecture and estimating regularization parameters on a validation set.

Results

- Derivation of an iterative gradient descent scheme for estimating regularization parameters. We study especially:
  - The multiple weight decay regularizer which may be interpreted as a way to perform architecture selection,
  - Single weight decay regularization.
- Architecture selection:
  - Validation set based OBD/OBS,
  - Approximative combinatorial architecture search using mean field annealing.

OUTLINE

- Training, validation and generalization.
- Estimating regularization parameters.
- Validation set based OBD/OBS.
- Combinatorial architecture optimization using mean field annealing.
- Numerical Experiments.
- Conclusions.

LEARNING FROM EXAMPLES

Data Set

$$\mathcal{D} = \{(\boldsymbol{x}(k), \boldsymbol{y}(k))\}_{k=1}^{N}$$

- $\boldsymbol{x}(k)$ is the input vector signal.
- $\boldsymbol{y}(k)$ is the output vector (target) signal.

Training and Validation Sets

Split into disjoint sets:

- *Validation* set $\mathcal{V}$ with $N_v = \lceil \gamma N \rceil$ examples.
- *Training* set $\mathcal{T}$ with $N_t = N - N_v$ examples.

$\boxed{0 < \gamma < 1 \text{ is the split-ratio}}$

COST FUNCTION

Model

$$\widehat{\boldsymbol{y}}(k) = \boldsymbol{f}(\boldsymbol{x}(k); \boldsymbol{w})$$

- $\widehat{\boldsymbol{y}}(k)$ is the prediction of the output $\boldsymbol{y}(k)$.
- $\boldsymbol{f}(\cdot)$ is the nonlinear mapping of the network.
- $\boldsymbol{w}$ is the vector of network parameters.

Training

$$\begin{aligned} C_{\mathcal{T}}(\boldsymbol{w}) &= S_{\mathcal{T}}(\boldsymbol{w}) + R(\boldsymbol{w}, \boldsymbol{\kappa}) \\ &= \frac{1}{N_t} \sum_{k=1}^{N_t} \ell\left(\boldsymbol{y}(k), \widehat{\boldsymbol{y}}(k); \boldsymbol{w}\right) + R(\boldsymbol{w}, \boldsymbol{\kappa}) \end{aligned}$$

- $\ell(\cdot)$ measures the distance between $\boldsymbol{y}(k)$ and $\widehat{\boldsymbol{y}}(k)$.
- $R(\boldsymbol{w}, \boldsymbol{\kappa})$ is the regularization term and $\boldsymbol{\kappa}$ the regularization parameters.

Weight Estimate

$$\widehat{\boldsymbol{w}}_{\mathcal{T}} = \arg \min_{\boldsymbol{w}} C_{\mathcal{T}}(\boldsymbol{w})$$

GENERALIZATION ERROR

Definition

The generalization error, G, is defined as the *expected loss* of the estimated model on a *independent* test sample $(\boldsymbol{x}, \boldsymbol{y})$.

$$G(\widehat{\boldsymbol{w}}_{\mathcal{T}}) = E\left\{\ell(\boldsymbol{y}, \widehat{\boldsymbol{y}}; \widehat{\boldsymbol{w}}_{\mathcal{T}})\right\} = \int \ell(\boldsymbol{y}, \widehat{\boldsymbol{y}}; \widehat{\boldsymbol{w}}_{\mathcal{T}}) \cdot p(\boldsymbol{x}, \boldsymbol{y})\, d\boldsymbol{x}\, d\boldsymbol{y}$$

Dependency On the Training Set

$G(\widehat{\boldsymbol{w}}_{\mathcal{T}})$ depends on $\mathcal{T}$ through the estimated weights, $\widehat{\boldsymbol{w}}_{\mathcal{T}}$.

Average Generalization Error

$$E_{\mathcal{T}}\left\{G(\widehat{\boldsymbol{w}}_{\mathcal{T}})\right\}$$

GOALS

Minimize

- generalization error, $G(\widehat{\boldsymbol{w}}_{\mathcal{T}})$, or
- average generalization error, $E_{\mathcal{T}}\{G(\widehat{\boldsymbol{w}}_{\mathcal{T}})\}$,

with respect to

- regularization parameters, κ, and/or
- network architecture

ARCHITECTURE OPTIMIZATION

Asymptotic ($o(1/N_t)$) analytical estimates of the average generalization error are available: (*Akaike, 69*), (*Moody, 91*), (*Larsen, 92*), (*Hansen, 93*), (*Larsen, 93*), (*Murata, 94*)

Estimates rely on strong statistical assumptions

Algorithm

1. Choose an initial network structure.
2. Train (retrain) the network.
3. Compute OBD/OBS saliencies or generalization based saliencies (*With, 96*) and remove some low saliency weights.
4. If the estimated average generalization error is minimal stop; otherwise, go to step 2.

OPTIMIZING REGULARIZATION

Method

In principle, solve the equation

$$\frac{\partial E_{\mathcal{T}}\left\{G(\widehat{\boldsymbol{w}}_{\mathcal{T}})\right\}}{\partial \boldsymbol{\kappa}} = \mathbf{0}$$

This is only possible analytically under restricted assumptions

---

Under assumptions (*Larsen, 94*):

- $R(\boldsymbol{w}, \boldsymbol{\kappa}) = \kappa |\boldsymbol{w}|^2$.
- Complete model.
- Input independent model noise with variance σ_ε^2

$$\kappa_{\mathrm{opt}} = \frac{\sigma_\varepsilon^2}{N_t} \cdot \frac{\mathrm{tr} \boldsymbol{H}^{+}(\boldsymbol{w}^\circ)}{(\boldsymbol{w}^\circ)^\top \boldsymbol{H}^{+}(\boldsymbol{w}^\circ) \boldsymbol{w}^\circ} + o(1/N_t)$$

- $\boldsymbol{H}(\boldsymbol{w}) = \partial^2 S_{\mathcal{T}}(\boldsymbol{w}) / \partial \boldsymbol{w} \partial \boldsymbol{w}^\top$.
- $\boldsymbol{w}^\circ$ are the optimal weights.

ADAPTIVE REGULARIZATION

Adaptive regularization suggested in (*Hansen, 93*), (*Hansen, 94*)

Conclusions

- Substituting current estimate $\widehat{\boldsymbol{w}}_{\mathcal{T}}$ for true weights $\boldsymbol{w}^{\circ}$ may sometimes lead to undesirable solutions
- The algorithm is generally sensitive to parameter settings.

VALIDATION ERROR APPROACH

$$S_{\mathcal{V}}(\widehat{\boldsymbol{w}}_{\mathcal{T}}) = \frac{1}{N_v} \sum_{k=1}^{N_v} \ell\left(\boldsymbol{y}(k), \widehat{\boldsymbol{y}}(k); \widehat{\boldsymbol{w}}_{\mathcal{T}}\right)$$

Property

$S_{\mathcal{V}}(\widehat{\boldsymbol{w}}_{\mathcal{T}})$ is an estimate of $G(\widehat{\boldsymbol{w}}_{\mathcal{T}})$ *not* $G(\widehat{\boldsymbol{w}}_{\mathcal{D}})$

Method

Select architecture/regularization by minimizing the validation error

VALIDATION ERROR APPROACH

Requirement

An architecture/regularization optimization procedure using the validation error approach should deliver lower generalization error than that of an *non-optimized* model trained on all data $\mathcal{D}$.

Potential Risk of Overusing the Validation Set

Impose prior constraints on the weights

Optimal Split-Ratio

- The optimal γ is dependent on the particular algorithm and objective.
- Results concerning estimating the generalization error was discussed in (*Larsen, 95*).

TOPIC

- ○ Training, validation and generalization.
- • $\boxed{\text{Estimating regularization parameters.}}$
- ○ Validation set based OBD/OBS.
- ○ Combinatorial architecture optimization using mean field annealing.
- ○ Numerical Experiments.
- ○ Conclusions.

ESTIMATING REGULARIZATION PARAMETERS

Fact

The validation error $S_{\mathcal{V}}(\widehat{\boldsymbol{w}}_{\mathcal{T}})$ is an *implicit function* of κ through $\widehat{\boldsymbol{w}}_{\mathcal{T}} = \widehat{\boldsymbol{w}}_{\mathcal{T}}(\boldsymbol{\kappa})$.

Gradient Descent

$$\boldsymbol{\kappa}(j+1) = \boldsymbol{\kappa}(j) - \eta \frac{\partial S_{\mathcal{V}}}{\partial \boldsymbol{\kappa}}(\widehat{\boldsymbol{w}}_{\mathcal{T}}(\boldsymbol{\kappa}(j)))$$

and

$$\frac{\partial S_{\mathcal{V}}}{\partial \boldsymbol{\kappa}}(\widehat{\boldsymbol{w}}_{\mathcal{T}}(\boldsymbol{\kappa}(j))) = \frac{\partial \boldsymbol{w}^{\top}}{\partial \boldsymbol{\kappa}}(\widehat{\boldsymbol{w}}_{\mathcal{T}}(\boldsymbol{\kappa}(j))) \cdot \frac{\partial S_{\mathcal{V}}}{\partial \boldsymbol{w}}(\widehat{\boldsymbol{w}}_{\mathcal{T}}(\boldsymbol{\kappa}(j)))$$

THE WEIGHT GRADIENT

Implicit Dependency

$$\frac{\partial^2 C}{\partial \boldsymbol{\kappa}^\top \partial \boldsymbol{w}}(\widehat{\boldsymbol{w}}_{\mathcal{T}}(\boldsymbol{\kappa}(j))) = \mathbf{0}$$

Assumption

$$R(\boldsymbol{w}, \boldsymbol{\kappa}) = \boldsymbol{\kappa}^\top \boldsymbol{r}(\boldsymbol{w}) = \sum_{i=1}^{q} \kappa_i r_i(\boldsymbol{w})$$

Result

$$\frac{\partial \boldsymbol{w}^\top}{\partial \boldsymbol{\kappa}}(\widehat{\boldsymbol{w}}_{\mathcal{T}}) = -\frac{\partial \boldsymbol{r}}{\partial \boldsymbol{w}^\top}(\widehat{\boldsymbol{w}}_{\mathcal{T}}) \cdot \boldsymbol{J}^{-1}(\widehat{\boldsymbol{w}}_{\mathcal{T}})$$

where $\boldsymbol{J} = \partial^2 C / \partial \boldsymbol{w} \partial \boldsymbol{w}^\top$

TOPIC

- ◦ Training, validation and generalization.
- ◦ Estimating regularization parameters.
- • Validation set based OBD/OBS.
- ◦ Combinatorial architecture optimization using mean field annealing.
- ◦ Numerical Experiments.
- ◦ Conclusions.

VALIDATION SET BASED OBD/OBS

$$S_{\mathcal{V}}(\boldsymbol{w}) \approx$$

$$S_{\mathcal{V}}(\widehat{\boldsymbol{w}}_{\mathcal{T}}) + \frac{\partial S_{\mathcal{V}}}{\partial \boldsymbol{w}^\top}(\widehat{\boldsymbol{w}}_{\mathcal{T}}) \cdot \Delta \boldsymbol{w} + \frac{1}{2} \Delta \boldsymbol{w}^\top \cdot \frac{\partial^2 S_{\mathcal{V}}}{\partial \boldsymbol{w} \partial \boldsymbol{w}^\top}(\widehat{\boldsymbol{w}}_{\mathcal{T}}) \cdot \Delta \boldsymbol{w}$$

where $\Delta \boldsymbol{w} = \boldsymbol{w} - \widehat{\boldsymbol{w}}_{\mathcal{T}}$

Example: vOBD

$$\Delta \boldsymbol{w} = [0 \cdots 0, -\widehat{w}_{\mathcal{T},i}, 0 \cdots 0]^\top$$

gives *saliencies*

$$\varrho_i = -\widehat{w}_{\mathcal{T},i} \cdot \frac{\partial S_{\mathcal{V}}}{\partial w_i}(\widehat{\boldsymbol{w}}_{\mathcal{T}}) + \frac{1}{2} \widehat{w}^2_{\mathcal{T},i} \cdot \frac{\partial^2 S_{\mathcal{V}}}{\partial w_i^2}(\widehat{\boldsymbol{w}}_{\mathcal{T}})$$

TOPIC

- ○ Training, validation and generalization.
- ○ Estimating regularization parameters.
- ○ Validation set based OBD/OBS.
- • Combinatorial architecture optimization using mean field annealing.
- ○ Numerical Experiments.
- ○ Conclusions.

COMBINATORIAL OPTIMIZATION

Goal

Perform an approximative search among the $2^m - 1$ architectures where $m = \dim(\boldsymbol{w})$.

Method

- Pick a binary state vector $\boldsymbol{s} \in \{0, 1\}^m$ indicating pruning or not, i.e., $\boldsymbol{f}(\boldsymbol{x}; \boldsymbol{w} \odot \boldsymbol{s})$.
- Estimate $\widehat{\boldsymbol{w}}_{\mathcal{T}}$ on the training set.
- Estimate $\boldsymbol{s}$ from the validation set using mean field annealing.

$\boxed{\text{Crucial problem: Retraining is not invoked}}$

TOPIC

- Training, validation and generalization.
- Estimating regularization parameters.
- Validation set based OBD/OBS.
- Combinatorial architecture optimization using mean field annealing.
- Numerical Experiments.
- Conclusions.

EXPERIMENTS WITH A LINEAR MODEL

Data Generation

$$y(k) = \boldsymbol{x}^\top(k)\boldsymbol{w}^\circ + \varepsilon(k)$$

- $\boldsymbol{x} \in \mathbb{R}^m$, $m = 10$ and $\boldsymbol{x}(k) \sim \mathcal{N}(\mathbf{0}, \boldsymbol{H})$ where $\boldsymbol{H}$ covariance matrix. $\boldsymbol{x}(k)$ is i.i.d. i.e., time-independent.
- $\varepsilon(k) \sim \mathcal{N}(0, \sigma_\varepsilon^2)$ is i.i.d. and independent of $\boldsymbol{x}(k)$. $\sigma_\varepsilon^2 = \mathrm{NSR} \cdot V\{y\}$ where $\mathrm{NSR} = 0.2$.
- "True" weights are $\boldsymbol{w}^\circ = [0000000111]^\top$.

Algorithm Evaluation

- Pick Q independent data sets of size $N = 60$.
- Choose the split-ratio γ.
- Train $m = 10$ dimensional linear models using the mean square error loss plus a regularization term as the cost function to obtain $\widehat{\boldsymbol{w}}_{\mathcal{T}}^{(i)}$, $i \in [1; Q]$.
- Calculate the true generalization error as:

$$G(\widehat{\boldsymbol{w}}_{\mathcal{T}}^{(i)}) = \sigma_\varepsilon^2 + (\widehat{\boldsymbol{w}}^{(i)\mathcal{T}} - \boldsymbol{w}^\circ)^\top \boldsymbol{H} (\widehat{\boldsymbol{w}}^{(i)\mathcal{T}} - \boldsymbol{w}^\circ)$$

MEASURES OF ALGORITHM PERFORMANCE

Probability of improvement

$$P_{\mathrm{imp}} = Q^{-1} \sum_{i=1}^{Q} \mu(G(\boldsymbol{w}_{\mathcal{D}}^{(i)}) - G(\widehat{\boldsymbol{w}}_{\mathcal{T}}^{(i)}))$$

- $\mu(x) = 1$ for $x > 0$, and zero otherwise.
- $G(\widehat{\boldsymbol{w}}_{\mathcal{D}})$ is the generalization error training on all data *without* architecture optimization.
- $G(\widehat{\boldsymbol{w}}_{\mathcal{T}})$ is the generalization error using the particular architecture optimization.

Relative Improvement in Generalization Error

$$\mathrm{RGI}^{(i)} = 100\% \cdot \frac{G(\boldsymbol{w}_{\mathcal{D}}^{(i)}) - G(\widehat{\boldsymbol{w}}_{\mathcal{T}}^{(i)})}{G(\boldsymbol{w}_{\mathcal{D}}^{(i)})}$$

SIMPLE WEIGHT DECAY

Regularizer

$$R(\boldsymbol{w}, \kappa) = \kappa \cdot |\boldsymbol{w}|^2$$

Observations

- $\partial S_{\mathcal{V}} / \partial \kappa = 0$ is a 3rd order equation in κ. The coefficients are very complicated functions of the data set.
- κ is found by using the MATLAB function *fmin.m*.
- The optimal κ might be negative (asymptotically positive).
- There is no potential risk of overusing the validation set.

RESULTS

γ	P_{imp}	$avr(\mathrm{RGI})$
0.4	0.846 ± 0.007	9.49
0.5	0.892 ± 0.006	9.61
0.6	0.911 ± 0.006	9.87
0.7	0.939 ± 0.005	9.88

- $Q = 10^4$.
- $\boldsymbol{H} = \boldsymbol{A}\boldsymbol{A}^\top$ where $\boldsymbol{A}$ is a randomly chosen Vandermonde matrix with large eigenvalue spread.
- Only weak dependency with split-ratio.

MULTIPLE WEIGHT DECAYS

Regularizer

$$R(\boldsymbol{w}, \boldsymbol{\kappa}) = \sum_{i=1}^{m} \kappa_i \cdot w_i^2$$

- Potential risk of overusing the validation set: a solution of $\partial S_{\mathcal{V}} / \partial \boldsymbol{\kappa} = 0$ is to choose $\boldsymbol{\kappa}$ such that $\widehat{\boldsymbol{w}}$ corresponds to training on the validation set!

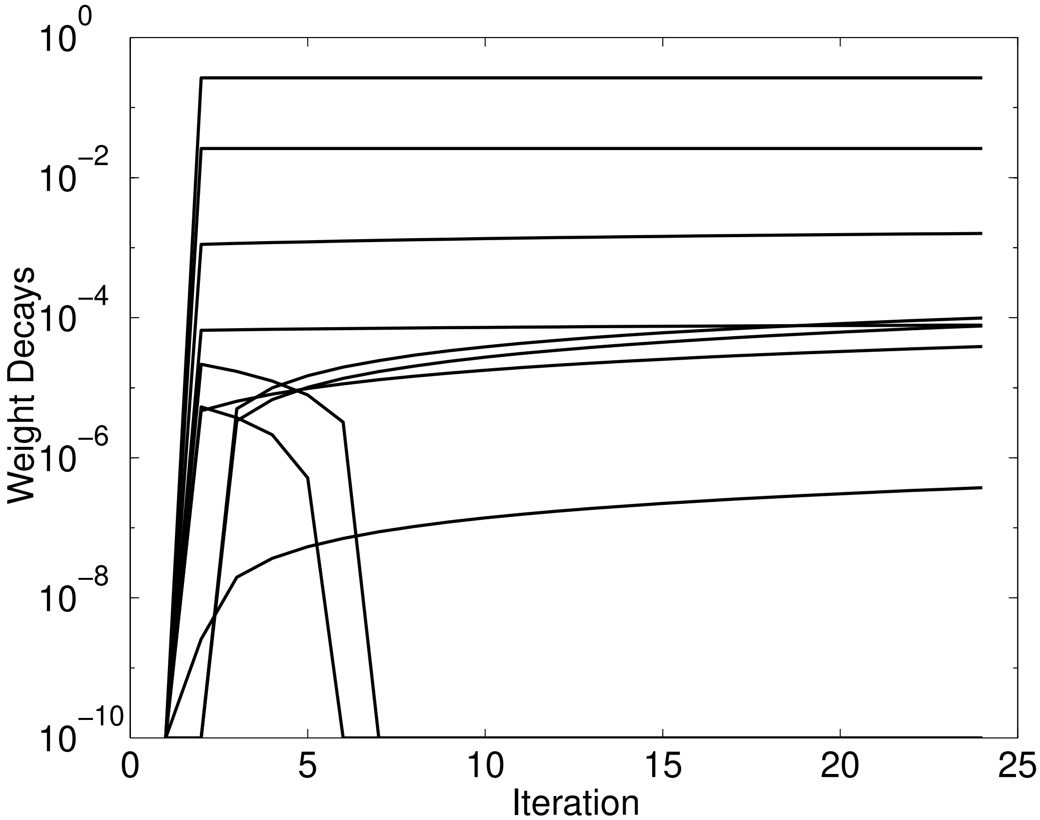
- Imposing constraints $\kappa_i \geq 0$ eliminates overuse and provides the interpretation as a *pruning regularizer*.

Algorithm

1. Choose step-size $\eta = 10^{-5}$ and initial $\kappa_i = 10^{-10}$ and the split-ratio γ.
2. Estimate (reestimate) weights.
3. If relative change in $S_{\mathcal{V}}$ is the less than $2 \cdot 10^{-4}$ go to step 5; otherwise, continue.
4. Update $\boldsymbol{\kappa}$ using gradient descent enforcing $\kappa_i \geq 0$, and go to step 2.
5. Retrain on *all* data using the regularization achieved.

RESULTS

Q	γ	P_{imp}	$avr(\mathrm{RGI})$
5000	0.5	0.964 ± 0.005	9.99
5000	0.65	0.959 ± 0.006	10.2
5000	0.7	0.939 ± 0.007	10.1



MULTIPLE WEIGHT DECAYS, OBD AND vOBD

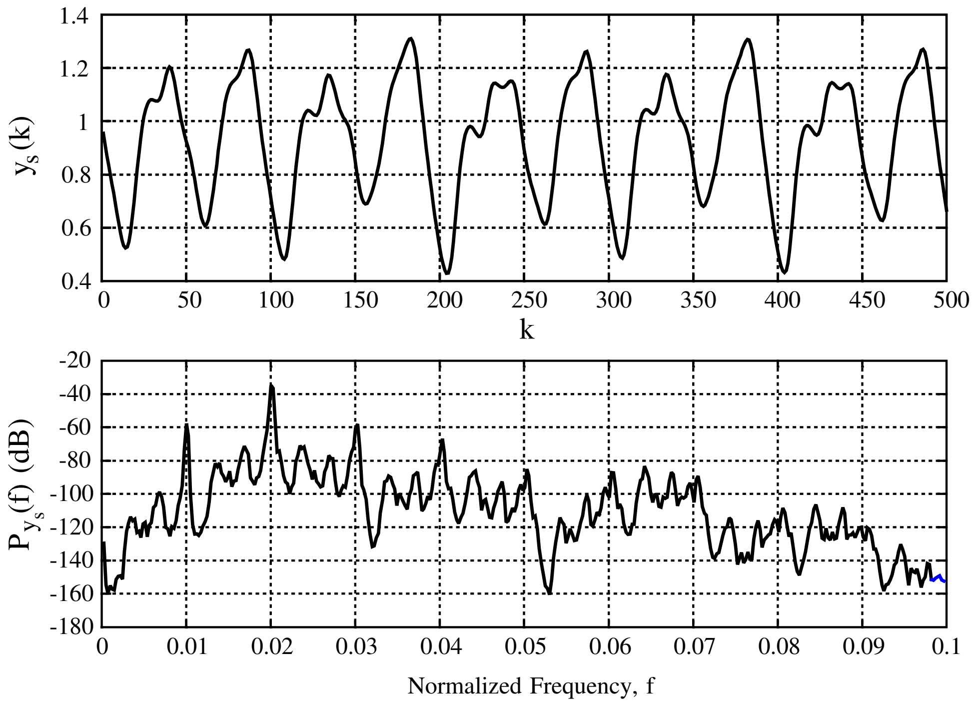
Q	Method	P_{imp}	$avr(\mathrm{RGI})$
5000	MWD	0.959 ± 0.006	10.2
10000	vOBD	0.863 ± 0.007	4.57
10000	OBD	0.835 ± 0.007	3.65

vOBD and OBD was run without regularization

EXPERIMENTS WITH FEED-FORWARD NEURAL NETWORK

Model

- Mackey-Glass chaotic time-series prediction.
- Input: $\boldsymbol{x}(k) = [x(k), x(k-6), x(k-12), x(k-18)]$.
- 15 hidden hyperbolic tangent neurons.
- Linear output: $\widehat{y}(k)$ which predict $y(k) = x(k+100)$.
- Data set of $N = 500$ examples.
- Test set of 8500 examples.



EXPERIMENTS

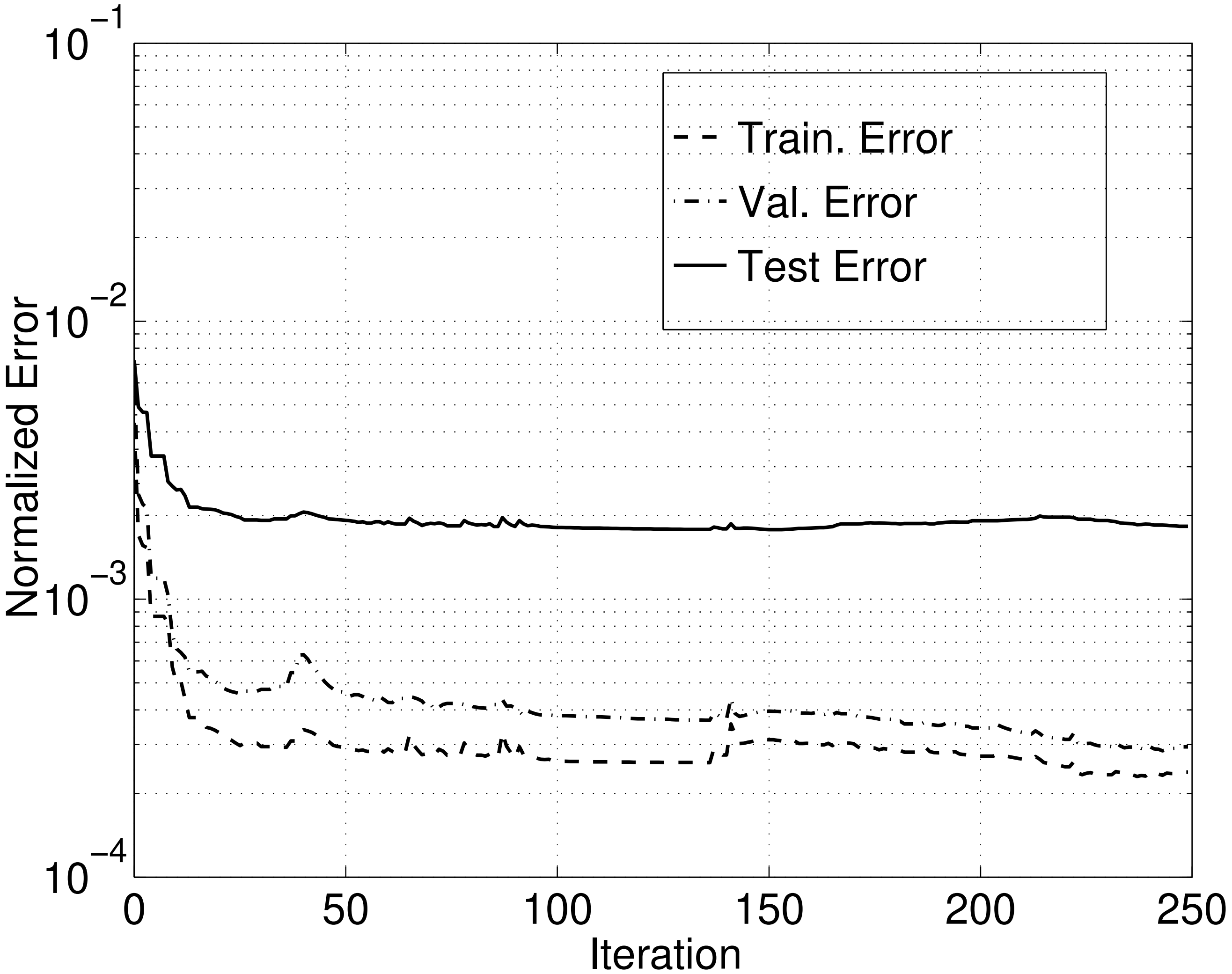
Multiple Weight Decay

- Split-ratio $\gamma = 0.5$
- Gauss-Newton training with initial $\kappa_i = 4 \cdot 10^{-6}$
- Gradient descent for κ enforcing $\kappa_i \geq 0$ with step-size initialized at $\eta = 10^{-3}$ and steadily adapted.
- Ensemble of 5 networks with different initial weights and data splitting.

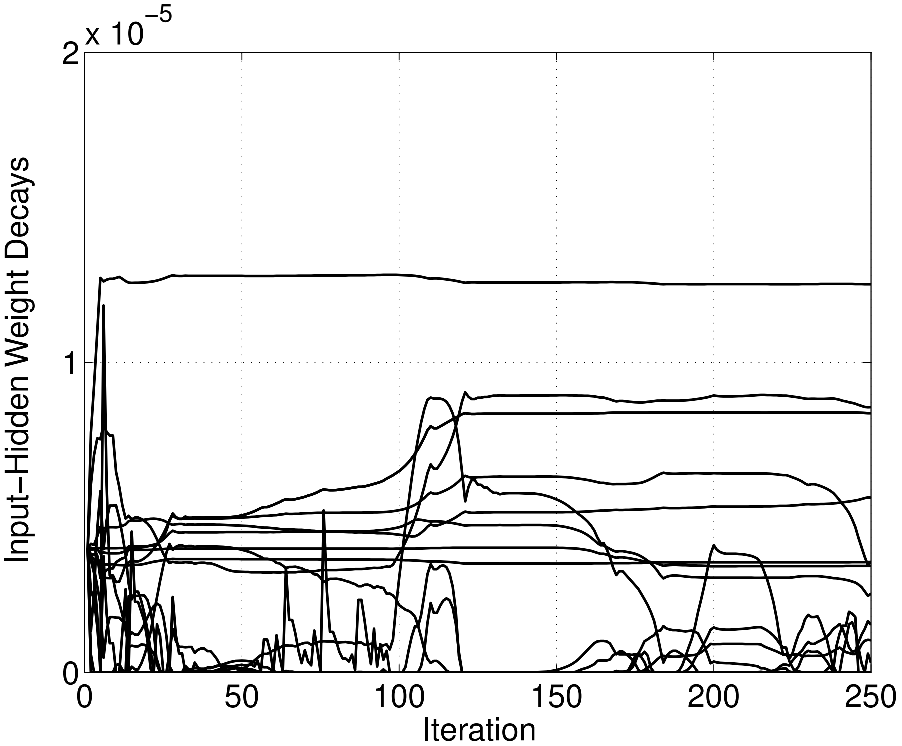
OBD

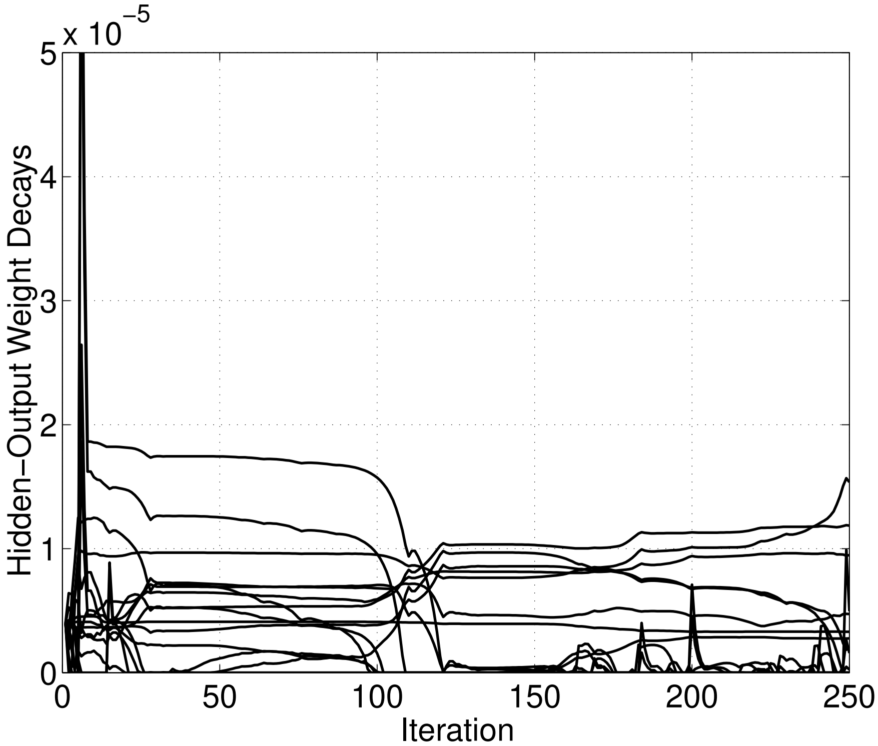
11 repetitions of OBD performed as reported in (*Svarer, 93*).

ALGORITHM RUN



WEIGHT DECAY EVOLUTION





RESULTS

		OBD	Indiv. Wgt. Decay
Norm. Test Error	avr	$2.07 \cdot 10^{-3}$	$2.61 \cdot 10^{-3}$
	min	$0.87 \cdot 10^{-3}$	$1.78 \cdot 10^{-3}$
	max	$3.68 \cdot 10^{-3}$	$3.26 \cdot 10^{-3}$
m_{eff}		83	58

Best results reported $\approx 2.5 \cdot 10^{-3}$.

CONCLUSIONS

- We suggest to optimize the network architecture by minimizing error on a validation set.
- We derived a gradient descent scheme for optimizing regularization parameters.
- Suggested to use validation set based OBD/OBS saliencies.
- Discussed the possibility of approximative combinatorial architecture search using mean field annealing.
- Numerical examples with linear models and neural networks demonstrated the viability of the framework.