# Stopped Training and Other Remedies for Overfitting

To appear in Proceedings of the 27th Symposium on the Interface, 1995

Warren S. Sarle
SAS Institute Inc., SAS Campus Drive, Cary, NC 27513, USA
saswss@unx.sas.com

## Abstract

Overfitting is the most serious problem in neural network training; stopped training is the most common solution to this problem. Yet the neural network literature contains little systematic investigation of the properties of stopped training and no comparisons with statistical methods of dealing with overfitting. This paper presents the results of simulations designed to investigate these issues.

## Introduction

The most important unresolved problem in practical applications of feedforward neural networks (NNs) is how to determine the appropriate complexity of the network. This problem is similar to the problem of choosing the degree of smoothing in nonparametric estimation. But overfitting in NNs can have much more serious consequences than can undersmoothing in most nonparametric estimation methods. NNs, like high-order polynomials, can produce wild predictions far beyond the range of the data.

There are two main approaches to controlling the complexity of a NN: model selection and regularization.

Model selection for a NN requires choosing the number of hidden units (HUs) and connections thereof. Related statistical problems include choosing the order of an autoregressive model and choosing a subset of predictors in a linear regression model. The usual NN approach to model selection is *pruning* (Reed 1993), in which you start with a large model and remove connections or units during training by various ad hoc algorithms. Pruning is somewhat similar to backward elimination in stepwise regression. Wald tests for pruning have been introduced in the NN literature under the name *Optimal Brain Surgeon* by Hassibi and Stork (1993).

The usual statistical approach to model selection is to estimate the generalization error for each model and to choose the model with the minimum such estimate. For nonlinear models, bootstrapping is generally the best way to estimate the generalization error, but Schwarz's Bayesian criterion (SBC) is reasonably effective and much cheaper than bootstrapping. Previous unpublished simulations showed that criteria such as AIC, FPE, and GCV, which choose more complex models than SBC, can overfit badly when applied to NNs.

Regularization involves constraining or penalizing the solution of the estimation problem to improve generalization by smoothing the predictions. Common NN methods for regularization include weight decay and addition of noise to the inputs during training. Both of these methods reduce to ridge regression in the case of linear models. Other forms of statistical regularization include Stein estimation and Bayesian estimation.

## Stopped training

The most popular method of regularizing NNs, which is called *stopped training* or *early stopping* or *optimal stopping*, proceeds as follows:

1. Divide the available data into two separate training and validation sets.
2. Use a large number of HUs.
3. Use small random initial values.
4. Use a slow learning rate.
5. Compute the validation error periodically during training.
6. Stop training when the validation error "starts to go up."

Stopped training is closely related to ridge regression. If the learning rate is sufficiently small, the sequence of weight vectors on each iteration will approximate the path of continuous steepest descent down the error function. Stopped training chooses a point along this path that optimizes an estimate of the generalization error computed from the validation set. Ridge regression also defines a path of weight vectors by varying

the ridge value. The ridge value is often chosen by optimizing an estimate of the generalization error computed by cross-validation, generalized cross-validation, or bootstrapping. There always exists a positive ridge value that will improve the expected generalization error in a linear model. A similar result has been obtained for stopped training in linear models (Wang, Venkatesh, and Judd 1994). In linear models, the ridge path lies close to, but does not coincide with, the path of continuous steepest descent; in nonlinear models, the two paths can diverge widely.

Stopped training has several advantages:

- It is fast.
- It can be applied successfully to networks in which the number of weights far exceeds the sample size. For example, Nelson and Illingworth (1991, p. 165) discuss training a network with 16,219 weights on only 50 training cases!
- It requires only one major decision by the user: what proportion of validation cases to use.

Statisticians tend to be skeptical of stopped training because:

- It appears to be statistically inefficient due to the use of the split-sample technique; i.e., neither training nor validation makes use of the entire sample.
- The usual statistical theory does not apply.

However, there has been recent progress addressing both of the above concerns (Wang 1994).

Despite the widespread use of stopped training, there has been very little research on the subject. Most published studies are seriously flawed. For example, Morgan and Bourlard (1990) generated artificial data sets in such a way that the training set and validation set were correlated, thus invalidating the results. Finnof, Hergert and Zimmermann (1993) generated artificial data with uniformly distributed noise and then trained the networks by least absolute values, a bizarre combination from a statistical point of view (also see Weigend 1994 on least-absolute-value training).

In one of the few useful studies on stopped training, Weigend (1994) claims that:

1. "... fairly small [4HU] networks (that never reach good performance) also, already, show overfitting."
2. "... large [15HU] networks generalize better than small ones"

These conclusions are based on a single data set with 540 training cases, 540 validation cases, 160 inputs, and a 6-category output, trained using the cross-entropy loss function, initial values uniformly distributed between $-.01$ and $+.01$, and a learning rate of .01 with no momentum. A 4HU network has 674 weights, which is greater than the number of training cases. Thus a 4HU network is hardly "fairly small" relative to the training set size, so (1) is not supported by Weigend's results.

Weigend's point (2) is clearly supported by the results, but generalizing from a single case is risky!

# Practical regularization

Ridge regression is the most popular statistical regularization method. Ridge regression can be extended to nonlinear models in a variety of ways:

- Penalized ridging, in which the estimation function is augmented by a penalty term, usually equal to the sum of squared weights; this method is called *weight decay* in the NN literature.
- Smoothed ridging, in which noise is added to the inputs, which amounts to an approximate convolution of the training data with the noise distribution; this method is used in the NN literature but does not have a widely recognized name.
- Constrained ridging, in which a norm of the weights is constrained not to exceed a specified value; this method has not appeared in the NN literature to my knowledge.

Penalized ridging has the clearest motivation, being a form of maximum-posterior-density (MPD) Bayesian estimation in which the weights are distributed $N(0, \text{ridge} \times I)$. The Bayesian approach has the great advantage of providing a natural way to estimate the ridge value: you make the ridge value a hyperparameter and estimate it along with everything else. Typically, the ridge value is given a noninformative prior distribution.

Full Bayesian estimation has been found very effective for NN training (Neal 1995) but is prohibitively expensive except for small data sets. Empirical Bayes methods (called the *evidence framework* in the NN literature, MacKay 199?), in which you first estimate the ridge value by integrating over the weights, are also effective but still expensive. Williams (1995) suggested first integrating over the ridge value and then maximizing with respect to the weights. This method is very efficient since the integration can be done analytically, but the global optimum is infinite, with all the weights

equal to zero, a solution that is rarely of much practical use.

Hierarchical Bayesian models with noninformative hyperprior distributions generally have an infinite posterior density along a ridge corresponding to a variance of zero for the hyperparameters. Sometimes there is a local maximum of the posterior density that yields an approximation to a full Bayesian analysis, but there is often only a *shoulder* rather than a local maximum (O'Hagan 1985).

For Bayesian estimation to be competitive with stopped training for practical applications, a computationally efficient approach is required. One simple and fast approach is to maximize the posterior density with respect to both the weights and the ridge value, using a slightly informative prior for the ridge value. A sufficiently informative prior will convert a shoulder into a local maximum, but how much prior information is required depends on the particular application.

The maximum posterior density approach yields a single model equation, so predictions can be computed quickly. But MPD estimation can be statistically inefficient compared to full Bayesian estimation in small samples when the mode of the posterior density is not representative of the entire posterior distribution. Hence the usefulness of MPD estimation in small samples requires investigation.

## Simulations

NNs are typically used with many inputs. But it is difficult to comprehend the nature of high-dimensional functions. The simulations presented in this paper were run using functions of a single input to make it possible to plot the estimated functions.

Six functions were used, named *flat, line, curve, wiggle, saw,* and *step*. Each figure at the end of this paper is devoted to one of these functions. Data were generated by selecting an input value from a uniform distribution on $[-2, 2]$ and adding Gaussian noise with standard deviation *sigma* as shown in the figures. Fifty samples were generated from the saw function, while twenty samples were generated from each of the other functions.

The figures are composed of four quadrants containing the following plots:

**Upper left:** The true regression function, pooled data from all samples, and separate data from each of the first two samples.

**Upper right:** The estimated regression functions for each sample with each of four estimation methods:

- SBC: Ordinary least squares with the number of HUs chosen by Schwarz's Bayesian criterion.
- DG: Ordinary least squares with the number of HUs chosen by Divine Guidance, i.e. the true generalization error.
- MPD: Maximum posterior density with 10 HUs and a slightly informative prior.
- Stopped training with 20 HUs and 25% validation cases.

**Lower left:** The mean MSE of prediction (above) and the median MSE of prediction (below) as functions of the number of HUs (on the left) and the percentage of validation cases (on the right).

**Lower right:** Box-and-whisker plots of the MSE of prediction for each estimation method.

In all simulations, the HUs had logistic activation functions and the output unit had an identity activation function.

Stopped training was done with 5, 10, 20, or 40 HUs. Each sample was randomly split into training and validation sets. The network was trained to convergence by a conjugate gradient algorithm. The validation error was computed on each iteration, and the weights that minimized the validation error were used to compute the estimated functions. This approach circumvents the question of when the validation error "starts to go up."

OLS and MPD estimation were done by a Levenberg-Marquardt algorithm. Ten HUs were used for MPD estimation.

Generalization error was estimated by computing the MSE for predicting the mean response for 201 data points evenly spaced from -2 to 2, which gives a close approximation to the mean integrated squared error.

Since the distribution of the MSE of prediction is highly skewed and possibly multimodal, a rank-transform analysis of variance was used to evaluate the effects of the number of HUs and the validation percentage and to compare estimation methods. Since this is a small, exploratory study, no adjustments were made for multiple tests. Results are summarized in terms of the conventional 5% level of significance.

There was no significant interaction between the number of HUs and the validation percentage for any of the functions.

The number of HUs had no significant effect on the generalization of stopped training for the flat or step functions. For the line function, 40 HUs generalized significantly better than 5 or 10 HUs. For the other functions, 20 HUs were better than 5 or 10 HUs. For the saw function, 40 HUs generalized significantly worse than 20

HUs. For the other functions, the difference between 20 and 40 HUs was not significant.

The effect of validation percentage on stopped training was not as clear-cut as the effect of the number of HUs. For the flat function, 33% validation cases generalized best but not significantly better than any other percentage except 14%. For the line function, the effect of validation percentage was not significant. For the curve function, 25% was best but was not significantly better than any other percentage except 10%. For the wiggle function, 14% was nonsignificantly better than 20% and significantly better than any other validation percentage. For the saw function, 14% was best but was not significantly better than any other percentage except 50%. For the step function, 50% was significantly worse than any other percentage. Thus the data suggest a trend in which the more complex functions require a smaller validation percentage, but the results are far from conclusive.

To compare stopped training with the other methods, 20 HUs and 25% validation cases were chosen to give good results for all five functions.

Of the three practical methods (DG is not, of course, practical), SBC was significantly best for the flat and line functions, but MPD was best for the other functions, significantly so for the wiggle and saw functions. For the curve and step functions, MPD was significantly better than stopped training but not significantly better than SBC. For the flat function, MPD was significantly worse than SBC or stopped training, while SBC and stopped training did not differ significantly. For the line function, both MPD and stopped training were significantly worse than SBC, but MPD and stopped training did not differ significantly from each other. Thus, for applications in which nonlinear relationships are anticipated, MPD is clearly the method of choice. Ironically, stopped training, which is touted as a fool-proof method for fitting complicated nonlinear functions, works well compared to MPD only for the two linear functions.

DG was significantly better than MPD for the flat and line functions and was extremely close to MPD for the other functions, so improved methods of model selection deserve further investigation. However, DG sometimes yields estimates with near discontinuities for all six functions, and for some applications this tendency could be a disadvantage. It is interesting that for the step function, SBC appears to the human eye to produce better estimates than DG. But even though SBC detects the discontinuities reliably, it does not position them accurately.

# References


Bernardo, J.M., DeGroot, M.H., Lindley, D.V. and Smith, A.F.M., eds., (1985), *Bayesian Statistics 2*, Amsterdam: Elsevier Science Publishers B.V. (North-Holland).

Finnof, W., Hergert, F., and Zimmermann, H.G. (1993), "Improving model selection by nonconvergent methods," *Neural Networks*, 6, 771-783.

Hassibi, B. and Stork, D.G. (1993), "Second order derivatives for network pruning: Optimal Brain Surgeon," *Advances in Neural Information Processing Systems*, 5, 164-171.

MacKay, D.J.C. (199?), "Probable networks and plausible predictions—a review of practical Bayesian methods for supervised neural networks," ftp://mraos.ra.phy.cam.ac.uk/pub/mackay/network.ps.Z

Morgan, N. and Bourlard, H. (1990), "Generalization and parameter estimation in feedforward nets: Some experiments," *Advances in Neural Information Processing Systems*, 2, 630-637.

Neal, R.M. (1995), *Bayesian Learning for Neural Networks,* Ph.D. thesis, University of Toronto, ftp://ftp.cs.toronto.edu/pub/radford/thesis.ps.Z

Nelson, M.C. and Illingworth, W.T. (1991), *A Practical Guide to Neural Nets*, Reading, MA: Addison-Wesley.

O'Hagan, A. (1985), "Shoulders in hierarchical models," in Bernardo et al. (1985), 697-710.

Reed, R. (1993), "Pruning Algorithms—A Survey," IEEE Transactions on Neural Networks, 4, 740-747.

Wang, C. (1994), *A Theory of Generalisation in Learning Machines with Neural Network Application*, Ph.D. thesis, University of Pennsylvania.

Wang, C., Venkatesh, S.S., and Judd, J.S. (1994), "Optimal Stopping and Effective Machine Complexity in Learning," *Advances in Neural Information Processing Systems*, 6, 303-310.

Weigend, A. (1994), "On overfitting and the effective number of hidden units," *Proceedings of the 1993 Connectionist Models Summer School,* 335-342.

Williams, P.M. (1995), "Bayesian regularization and pruning using a Laplace prior," *Neural Computation*, 7, 117-143.

Figure 1

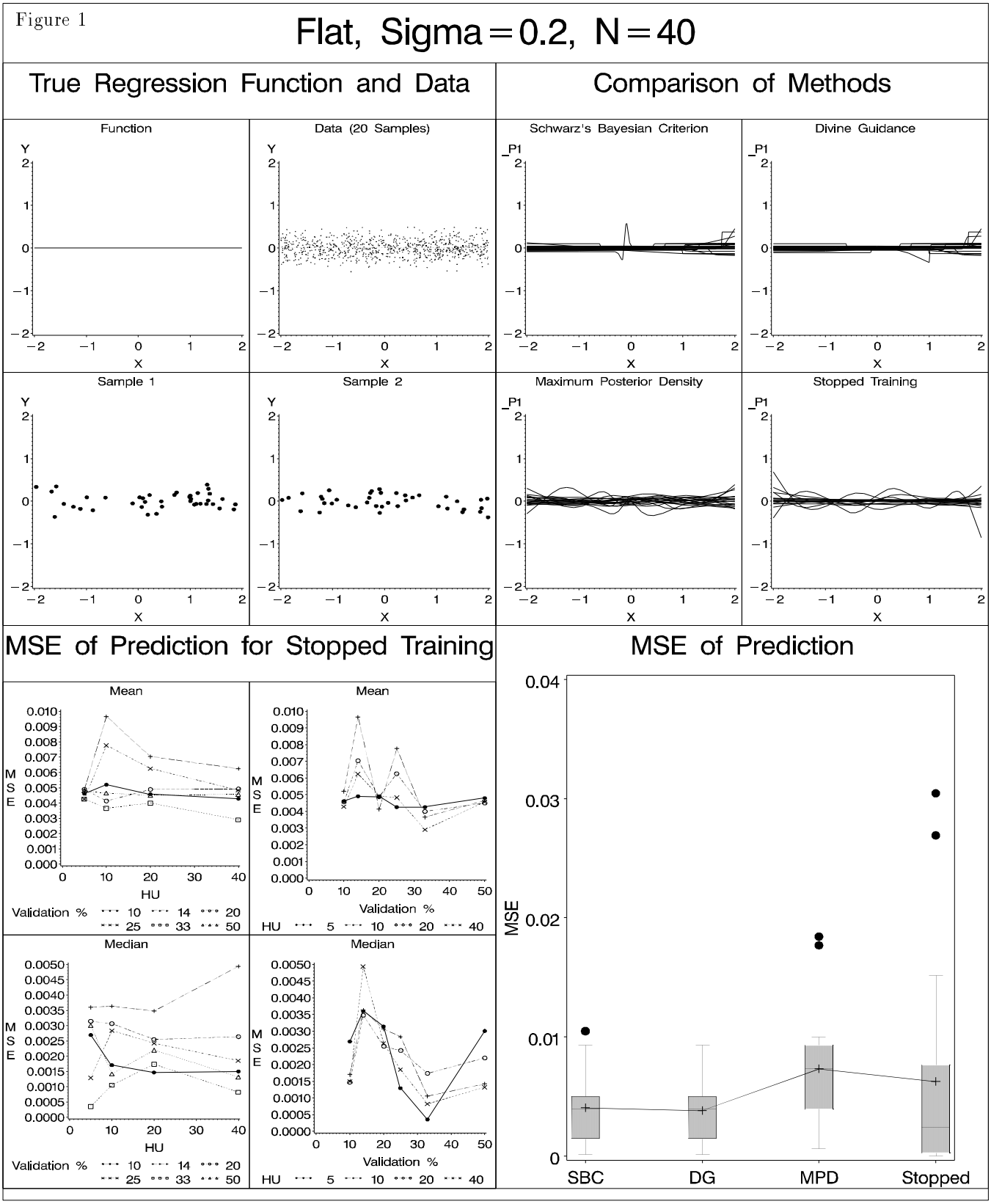

Flat, Sigma = 0.2, N = 40
True Regression Function and Data
Comparison of Methods
Function
Data (20 Samples)
Schwarz's Bayesian Criterion
Divine Guidance
Sample 1
Sample 2
Maximum Posterior Density
Stopped Training
MSE of Prediction for Stopped Training
MSE of Prediction
Mean
Median
HU
Validation %
SBC
DG
MPD
Stopped

Figure 2

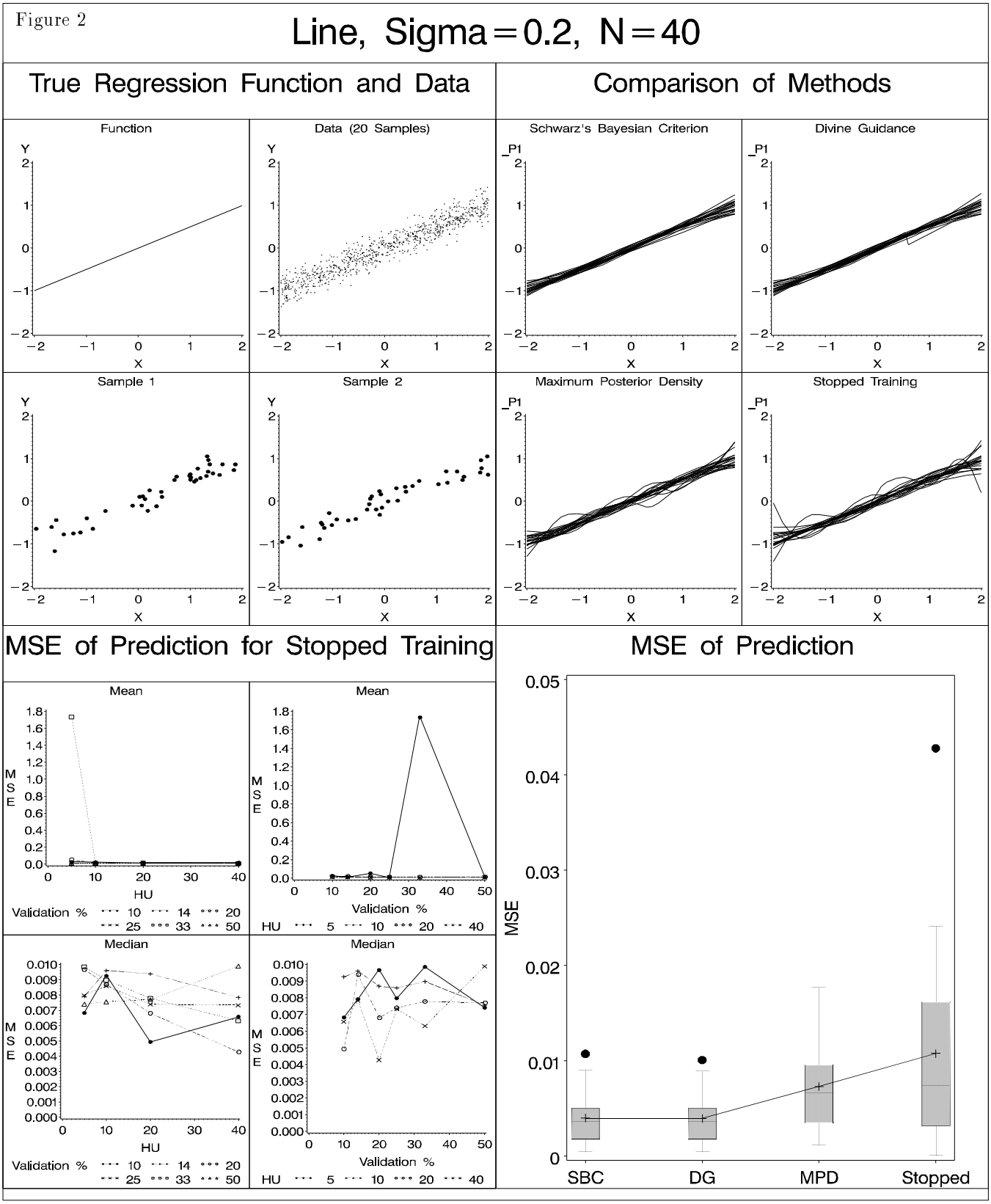

Line, Sigma = 0.2, N = 40
True Regression Function and Data
Comparison of Methods
Function
Data (20 Samples)
Schwarz's Bayesian Criterion
Divine Guidance
Sample 1
Sample 2
Maximum Posterior Density
Stopped Training
MSE of Prediction for Stopped Training
MSE of Prediction
Mean
Median
HU
Validation %
SBC
DG
MPD
Stopped

Figure 3

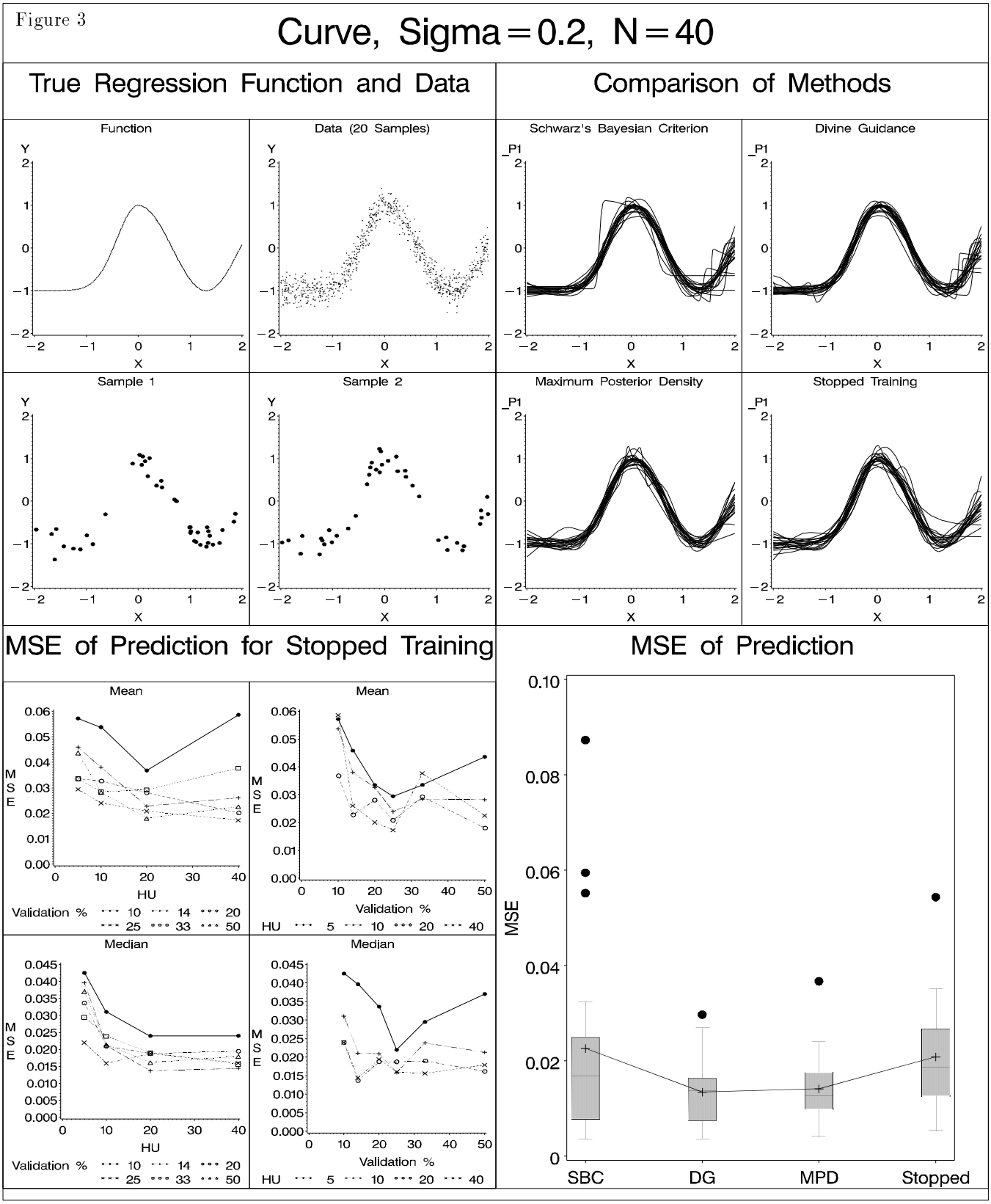

Curve, Sigma = 0.2, N = 40
True Regression Function and Data
Comparison of Methods
Function
Data (20 Samples)
Schwarz's Bayesian Criterion
Divine Guidance
Sample 1
Sample 2
Maximum Posterior Density
Stopped Training
MSE of Prediction for Stopped Training
Mean
Median
Validation %
HU
MSE of Prediction
SBC
DG
MPD
Stopped

Figure 4

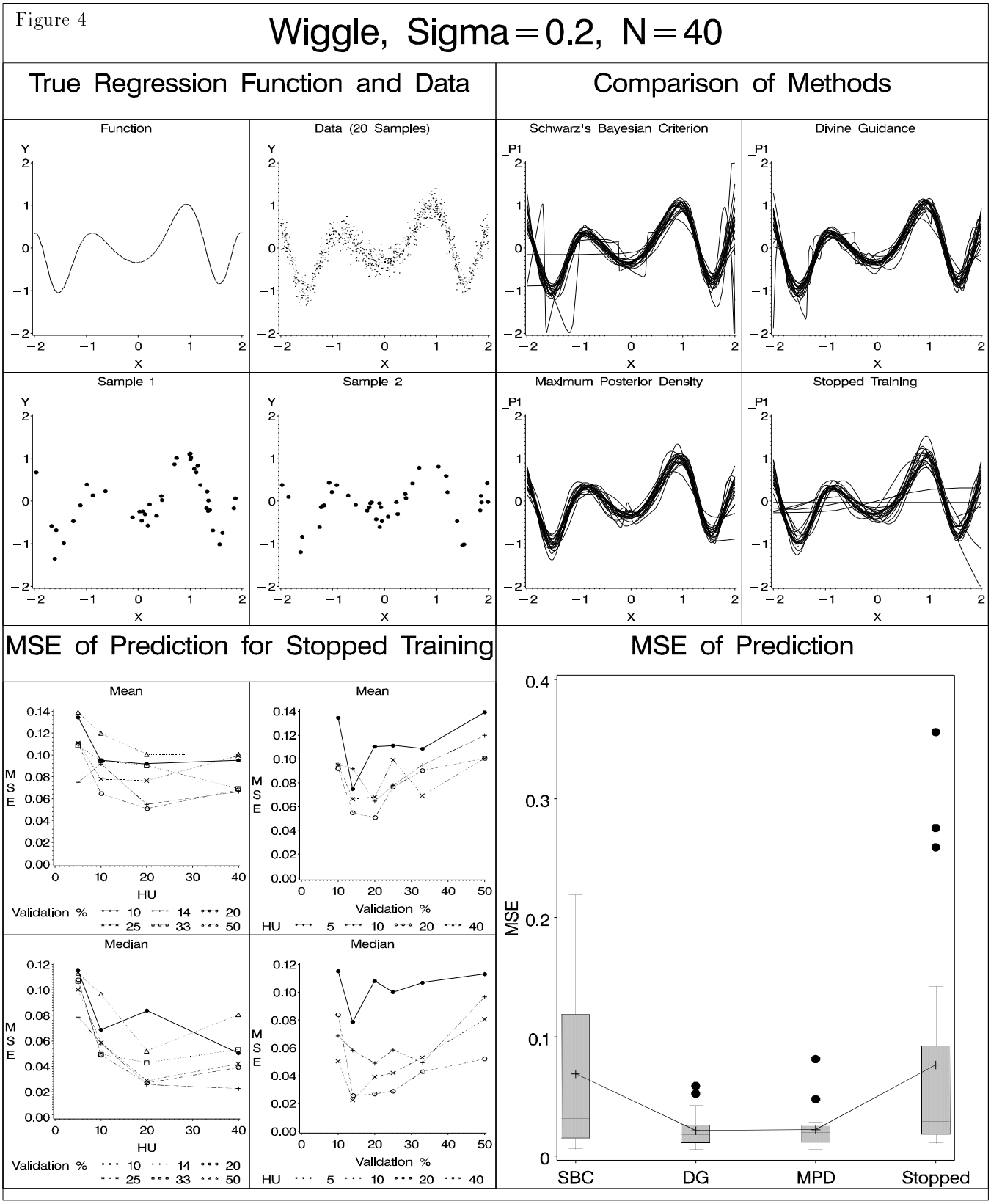

Wiggle, Sigma = 0.2, N = 40
True Regression Function and Data
Comparison of Methods
Function
Data (20 Samples)
Schwarz's Bayesian Criterion
Divine Guidance
Sample 1
Sample 2
Maximum Posterior Density
Stopped Training
MSE of Prediction for Stopped Training
MSE of Prediction
Mean
Median
HU
Validation %
SBC
DG
MPD
Stopped

Figure 5

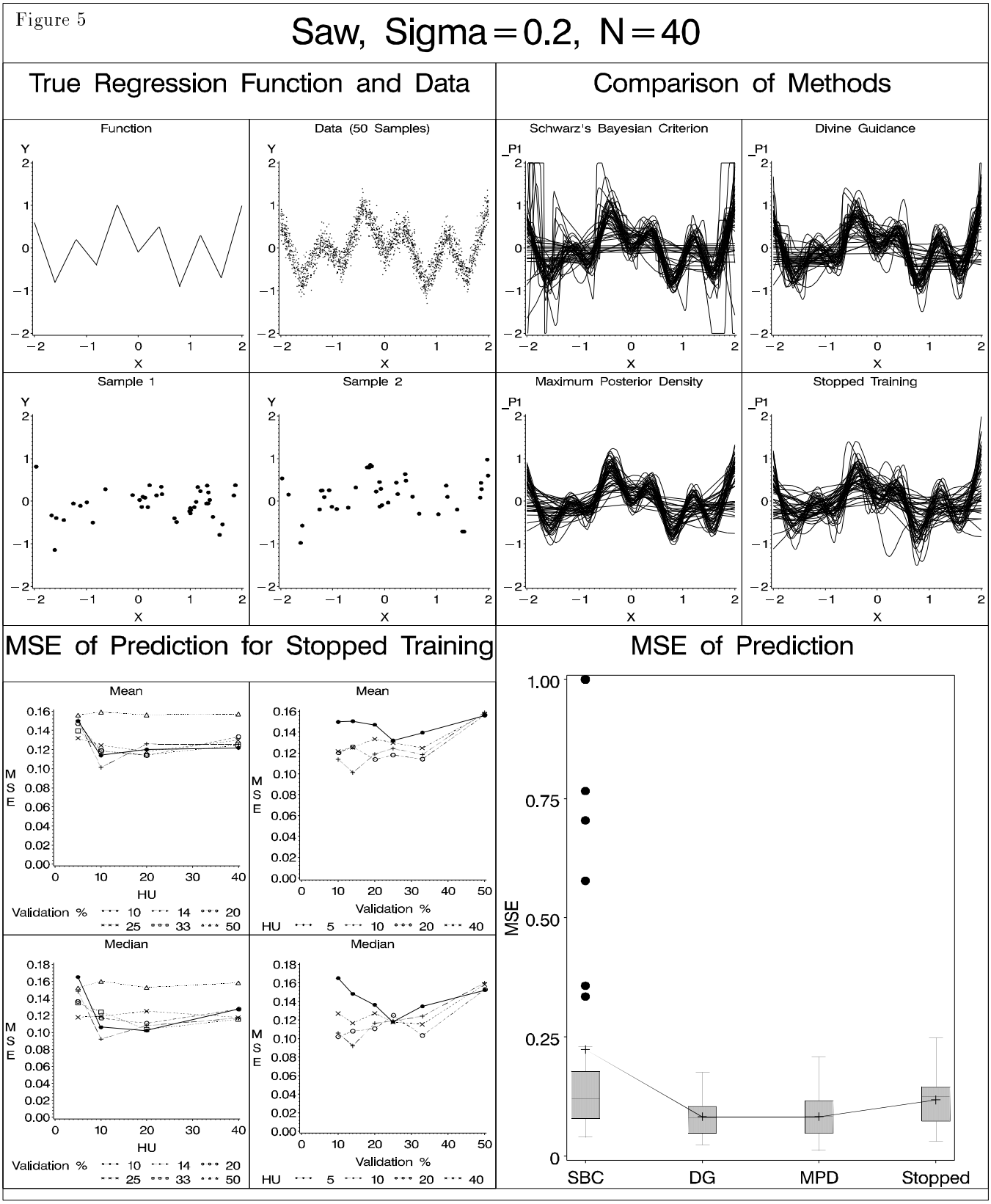
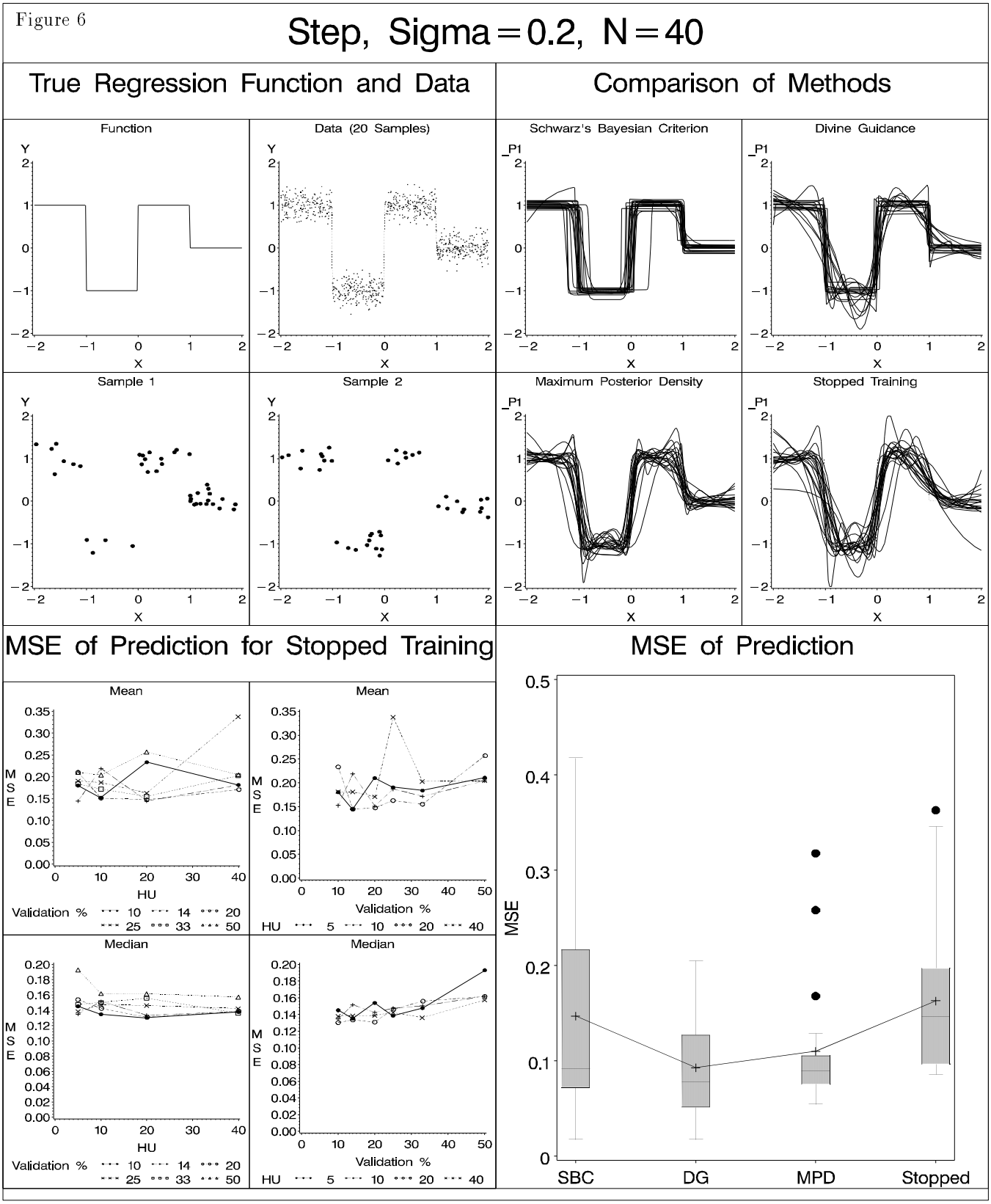

Saw, Sigma = 0.2, N = 40
True Regression Function and Data
Comparison of Methods
Function
Data (50 Samples)
Schwarz's Bayesian Criterion
Divine Guidance
Sample 1
Sample 2
Maximum Posterior Density
Stopped Training
MSE of Prediction for Stopped Training
MSE of Prediction
Mean
Median
HU
Validation %
SBC
DG
MPD
Stopped

Figure 6


Step, Sigma = 0.2, N = 40
True Regression Function and Data
Comparison of Methods
Function
Data (20 Samples)
Schwarz's Bayesian Criterion
Divine Guidance
Sample 1
Sample 2
Maximum Posterior Density
Stopped Training
MSE of Prediction for Stopped Training
MSE of Prediction
Mean
Median
HU
Validation %
SBC
DG
MPD
Stopped
MSE