sas

Exploring Modern Regression Methods Using SAS®

Special Collection

Foreword by
Phil Gibbs

The correct bibliographic citation for this manual is as follows: Gibbs, Phil. 2019. *Exploring Modern Regression Methods Using SAS®: Special Collection*. Cary, NC: SAS Institute Inc.

Exploring Modern Regression Methods Using SAS® : Special Collection

978-1-64295-487-6 (Paperback)
978-1-64295-448-7 (Web PDF)

SAS Institute Inc., SAS Campus Drive, Cary, NC 27513-2414

June 2019

Table of Contents

Free SAS® e-Books: Special Collection

In this series, we have carefully curated a collection of papers that introduces and provides context to the various areas of analytics. Topics covered illustrate the power of SAS solutions that are available as tools for data analysis, highlighting a variety of commonly used techniques.

Discover more free SAS e-books!
support.sas.com/freesasebooks

sas.com/books
for additional books and resources.

sas
THE POWER TO KNOW®

About This Book

What Does This Collection Cover?

SAS has been implementing methods such as quantile regression and generalized additive models in SAS/STAT procedures for more than 20 years. As newer model building techniques for general linear models and generalized linear models are introduced, the number of methods can be overwhelming.

This special collection of papers demonstrates new and enhanced capabilities and applications of lesser-known SAS/STAT and SAS Viya procedures for regression models. The goal here is to raise awareness of current valuable SAS/STAT content of which the user may not be aware.

SAS offers numerous analytics solutions. The papers in this collection are from the *Proceedings of the SAS Global Forum.* For more from the *Proceedings,* visit the online versions.

Additional helpful resources are available at support.sas.com and sas.com/books.

We Want to Hear from You

Do you have questions about a SAS Press book that you are reading? Contact us at saspress@sas.com.

SAS Press books are written *by* SAS Users *for* SAS Users. Please visit sas.com/books to sign up to request information on how to become a SAS Press author.

We welcome your participation in the development of new books and your feedback on SAS Press books that you are using. Please visit sas.com/books to sign up to review a book

Learn about new books and exclusive discounts. Sign up for our new books mailing list today at https://support.sas.com/en/books/subscribe-books.html.

Foreword

Regression is the process of fitting a model to data. The earliest regression models were simple linear regression models that later evolved into inferential analysis of variance (ANOVA) models based on designed experiments. Now, regression encompasses a variety of complex topics, and the regression landscape can be quite difficult to navigate.

Today, statisticians and data scientists face issues in selecting the best candidate model when the data can contain hundreds, if not thousands, of predictors. These issues surface in situations in which the following might occur:

- The relationship between the predictors and the response might not be linear
- The form of the relationship between the predictors and the response might not be known at all
- The shape of the model could depend on the conditional quantile of the response

This special collection of papers focuses on using the regression model building process to address today's challenges. SAS offers a complete set of tools for building modern regression models:

- SAS/STAT® software can address responses that come from a variety of statistical distributions.
- Recent additions to the software make it possible to easily program complex structures like spline effects.
- SAS® Viya® software makes it easy to fit regression models to very large sets of data.
- The SAS Studio environment empowers novice users to learn the SAS programming skills required to build complex regression models.

We have carefully selected this collection of papers from recent SAS Global Forum presentations to introduce you to some of the lesser-known SAS regression methods and procedures, the benefits of them, and when to use them.

Step Up Your Statistical Practice with Today's SAS/STAT® Software, by Robert N. Rodriguez, Phil Gibbs, and Randy Tobias

Has the rapid pace of SAS/STAT releases left you unaware of powerful enhancements that could make a difference in your work? Are you still using PROC REG rather than PROC GLMSELECT to build regression models? Do you understand how the GENMOD procedure compares with the newer GEE and HPGENSELECT procedures? When should you turn to PROC ICPHREG rather than PROC PHREG for survival modeling?

This paper increases awareness of modern tools in SAS/STAT by providing high-level comparisons with well-established tools and explaining the benefits of enhancements and new procedures. It focuses on regression model building, generalized linear models, survival analysis, and mixed models. It also points out resources that will guide you to new tools in other important areas, such as Bayesian analysis, causal inference, item response theory, methods for missing data, and survey data analysis. When you see the advantages of the modern tools, you will want to put them into practice.

We hope that these selections give you a useful overview of the many tools and techniques that are available in SAS /STAT so that you can choose the best method to build your regression model.

Phil Gibbs
Manager of Advanced Analytics, Technical Support, SAS

Phil Gibbs is the manager of the Advanced Analytics group in SAS Technical Support. His team of statisticians, operations research analysts, econometricians, and data scientists help customers use SAS software to solve their business problems. Phil has been a SAS user for 38 years, 27 of which have been at SAS Institute. He has authored numerous SAS Global Forum papers, with special research interests in mixed models, simulation, and optimization. Phil earned his master of science degree in mathematics and statistics at Clemson University.

Paper SAS521-2017

Step Up Your Statistical Practice with Today's SAS/STAT® Software

Robert N. Rodriguez, Phil Gibbs, and Randy Tobias, SAS Institute Inc.

Abstract

Has the rapid pace of SAS/STAT® releases left you unaware of powerful enhancements that could make a difference in your work? Are you still using PROC REG rather than PROC GLMSELECT to build regression models? Do you understand how the GENMOD procedure compares with the newer GEE and HPGENSELECT procedures? When should you turn to PROC ICPHREG rather than PROC PHREG for survival modeling?

This paper will increase your awareness of modern tools in SAS/STAT by providing high-level comparisons with well-established tools and explaining the benefits of enhancements and new procedures. The paper focuses on new tools in the areas of regression model building, generalized linear models, survival analysis, and mixed models. When you see the advantages of these tools, you will want to put them into practice. The paper also points out resources that will guide you to new tools in other important areas, such as Bayesian analysis, causal inference, item response theory, methods for missing data, and survey data analysis.

Introduction

Are you a creature of habit when it comes to analyzing data? Do you still rely on PROC REG and PROC GLM for your regression studies because those are the procedures you learned about in school? Have you heard about recent releases of SAS/STAT and new procedures—but not found time to check them out? If so, the procedures that you know best might not be your best choices when compared with newer procedures that deliver significant advances in methodology. And you might not be aware of alternatives that could make a difference in your work.

This paper provides that awareness. Each of its four main sections focuses on an area where SAS/STAT has grown significantly in recent years:

- The section on "Regression Model Building" describes new tools for selecting the effects in your model when you have many variables to choose from—continuous or categorical. You might be building traditional explanatory models if your goal is to gain insights. Or you might now be building predictive models if your goal is accurate prediction with new data. Either way, you can build better models by applying modern selection methods, such as the lasso, and you can build a broader range of models in which the response can be categorical or continuous.
- The section on "Inferential Analysis of Generalized Linear Models" describes new tools for different kinds of inference—such as estimation of treatment effects—within the framework of generalized linear models. These tools help you to take advantage of modern Bayesian methods, deal with overdispersion, and handle missingness that is due to dropouts in longitudinal studies.
- The section on "Survival Analysis" describes new tools for estimation and hypothesis testing and for modeling the outcome of interest when you have time-to-event data. These tools are indispensable for valid inference because they are specialized for particular problems that you encounter with right-censored data, interval-censored data, competing risks, and clustered data.
- The section on "Analysis of Mixed Models" describes the various procedures available in SAS/STAT software for handling models with both fixed and random effects. Understanding how these tools compare in terms of flexibility and practical advantages will help you decide which ones to apply in your work.

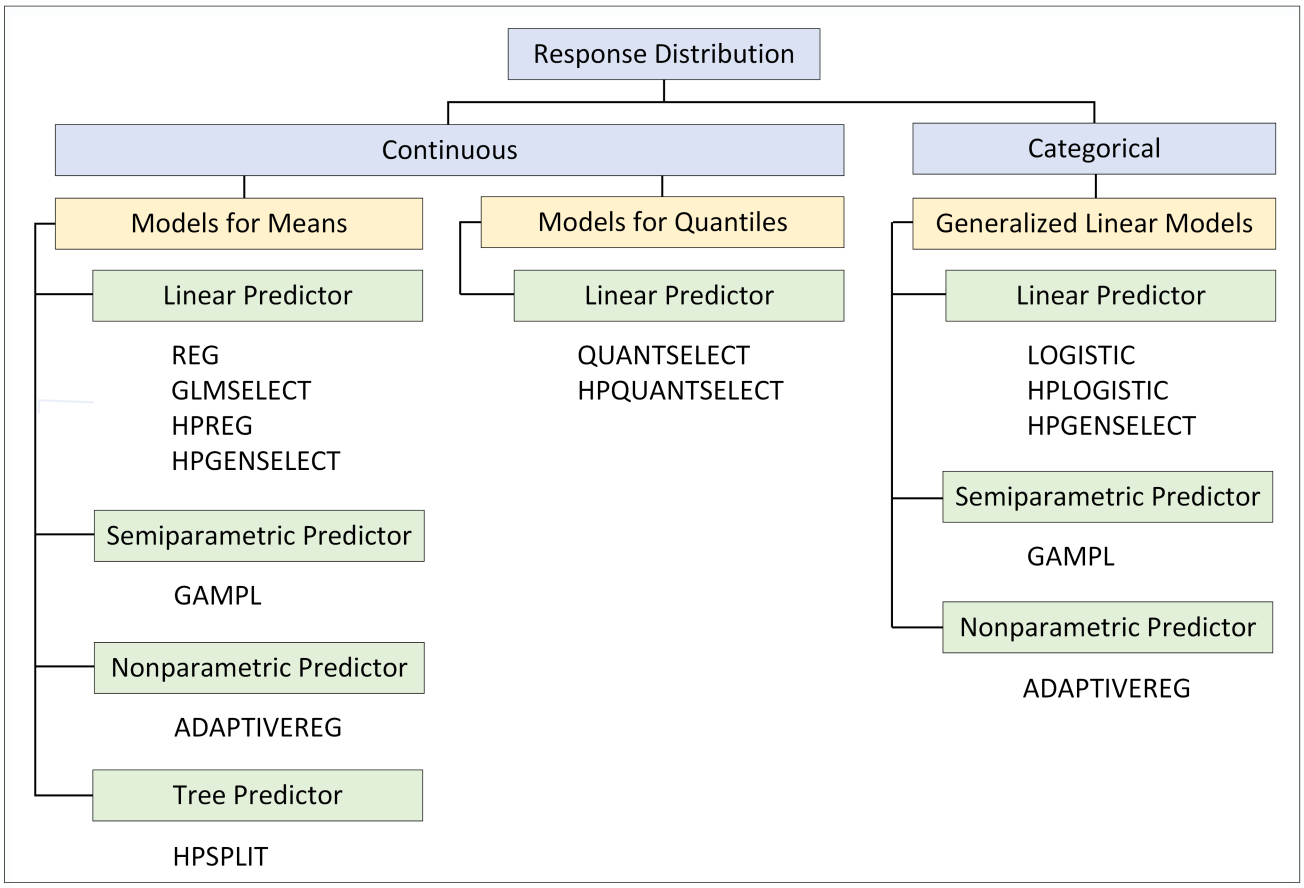
Each section begins by noting the most familiar procedures in that area, and it then presents new tools—enhancements and new procedures—that give you greater flexibility for statistical modeling, specialized inference for complex data, and improved performance for large data. The discussion compares the objectives, assumptions, and benefits of the new tools.

Because this paper presents a high-level view, it does not include examples or explanations of methods. Instead, each section refers to introductory papers that cover these aspects. The final section points out resources that guide you to new tools in areas of SAS/STAT software that are not covered here, such as Bayesian analysis, causal inference, item response theory, methods for missing data, and survey data analysis.

Regression Model Building

One of the most frequently asked questions in statistical practice is the following: "I have hundreds of variables—even thousands, so which ones should I include in my regression model?" Figure 1 presents the various regression model building procedures now available in SAS/STAT, and it shows how they are related to each other.

Figure 1 Procedures for Regression Model Building



The models you can build with these procedures fall into three broad categories: regression models for means of continuous responses, quantile regression models for continuous responses, and generalized linear models for categorical responses. In Figure 1, the term "generalized linear models" refers to regression models for categorical responses that assume a parametric distribution in the exponential family and a corresponding link function.

With the exception of PROC REG, all the procedures in Figure 1 enable you to specify predictors that are continuous or categorical. Most of these procedures build models in which the predictor effects enter the model linearly. The HPSPLIT procedure fits tree models in which the predictor variables enter the model through indicator functions of regions of the predictor space that are defined by variable splits. The GAMPL procedure allows spline functions of continuous predictors in addition to linear predictors, and the ADAPTIVEREG procedure builds predictors that are based entirely on splines. The next sections describe the capabilities and benefits of the model building procedures in each of the three categories.

Building Regression Models for Means of Continuous Responses

The REG procedure has always served the dual purposes of fitting and building standard regressions models, which apply to continuous responses and assume no parametric distribution for the response. However, this procedure is limited to regression models in which the predictors are continuous variables. Longtime users of PROC REG are often surprised to learn that this limitation is overcome by the GLMSELECT procedure, which is now the flagship SAS/STAT procedure for building standard regression models. Keep in mind that the REG procedure is still the preferred tool for fitting standard regression models when you need inferential methods, influence statistics, and diagnostic plots.

A major advantage of PROC GLMSELECT over PROC REG is that it supports effect selection methods for general linear models of the form

$$y_i = \beta_0 + \beta_1 x_{i1} + \cdots + \beta_p x_{ip} + \epsilon_i, \quad i = 1, \ldots, n$$

where the response y_i is continuous and the predictors $x_{i1}, \ldots, x_{ip}$ represent main effects that consist of continuous or classification variables, and interaction effects of these variables. You specify the model by using MODEL and CLASS statements as in the GLM procedure. By using the EFFECT statement, you can include more types of effects—such as polynomial and spline effects—that are constructed from the variables.

Another advantage of PROC GLMSELECT is that it provides lasso methods, introduced by Tibshirani (1996), in addition to the forward, backward, and stepwise selection methods available in the REG procedure. Lasso methods leave all the effects in the model, but they restrict their parameters by setting some to zero while shrinking others toward zero. Thus, they produce models that are sparser and potentially more interpretable (Hastie, Tibshirani, and Wainwright 2015). Table 1 summarizes the selection methods available in the GLMSELECT procedure.

Table 1 Effect Selection Methods in the GLMSELECT Procedure

Method	Description
Forward selection	Starts with no effects and adds effects
Backward elimination	Starts with all effects and deletes effects
Stepwise selection	Starts with no effects; effects are added and can be deleted
Least angle regression	Starts with no effects and adds effects; at each step, estimated βs are shrunk toward 0
Lasso	Constrains sum of absolute βs; some βs set to 0
Elastic net	Constrains sums of absolute and squared βs; some βs set to 0
Adaptive lasso	Constrains sum of absolute weighted βs; some βs set to 0
Group lasso	Constrains sum of Euclidean norms of βs corresponding to effects; all βs for the same effect are set to 0 or are nonzero

The GLMSELECT procedure also provides extensive capabilities for customizing effect selection. You can specify information criteria or criteria based on significance levels. You can also specify criteria based on validation; this approach avoids overfitting the training data by partitioning the data into subsets for training, validation, and testing.

To address the computational demands of selection from a very large number of effects, the GLMSELECT procedure has added screening approaches that you can combine with selection methods to reduce the number of regressors to a smaller subset on which the selection is then performed.

Cohen (2006) provides an introduction to the GLMSELECT procedure, and Cohen (2009) describes its strengths for building models with large data. Günes (2015) discusses regression methods based on penalization. Gibbs et al. (2013) explain the versatility of the EFFECT statement, which is available in many SAS/STAT modeling procedures.

The HPREG procedure is a high-performance procedure that has many of the same features as the GLMSELECT procedure for fitting and building standard regression models. PROC HPREG is referred to as a high-performance procedure because it runs in either single-machine mode or distributed mode, and it is multi-threaded. Cohen and Rodriguez (2013) describe the design of high-performance statistical modeling procedures and discuss when these procedures provide performance benefits.

The HPSPLIT procedure is a high-performance procedure that builds regression trees, which model continuous responses, and classification trees, which model categorical responses. The predictor variables can be categorical or continuous, and the tree is built by recursively splitting the predictor space into nonoverlapping segments, which define the terminal nodes of the tree. The process begins by growing a large, full tree. To prevent overfitting, the full tree is pruned back to a smaller subtree that balances the goals of fitting the training data and predicting new data. The average response of the training observations in a terminal node serves to predict the response for new observations that fall into that node.

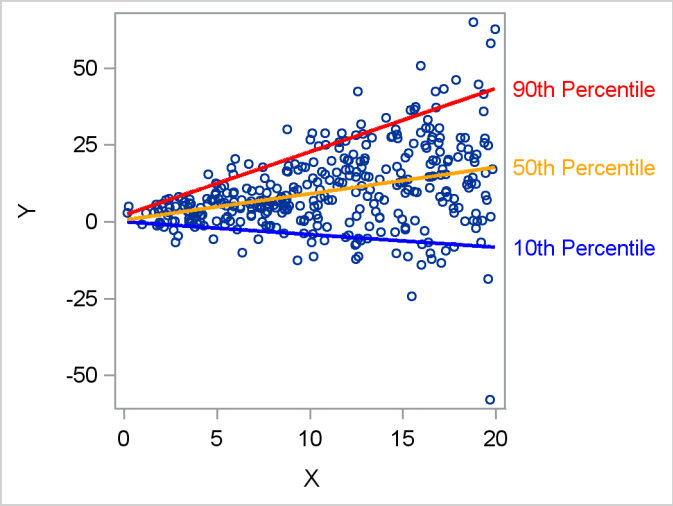
An advantage of regression trees over standard regression models is that they are easy to explain; tree diagrams can be highly interpretable when the tree size is small. On the other hand, because regression trees lack flexibility for capturing smooth relationships between the predictors and the response, they often fail to provide the predictive accuracy of linear regression models.

Building Quantile Regression Models for Continuous Responses

The standard regression models that you build with the GLMSELECT and HPREG procedures predict the conditional mean of the response ($E[Y|X]$), and they assume that the conditional variance of the response is constant ($\text{Var}[Y|X] = \sigma^2$). Models of this type cannot describe data in which the shape of the response distribution depends on the predictors.

For example, consider the data shown in Figure 2, where the variance of Y increases with X. You can use a simple linear regression model to predict $E[Y|X]$, but this model cannot account for the variation in the conditional distribution of Y.

Figure 2 Quantile Regression Models for Three Percentiles



Quantile regression, introduced by Koenker and Bassett (1978), uses a general linear model to fit conditional quantiles—more commonly referred to as percentiles—of the response without assuming a parametric distribution for the response. Figure 2 shows quantile regression lines for the 10th, 50th, and 90th conditional percentiles of Y, fitted with the QUANTREG procedure. By fitting a more extensive set of percentiles, you can describe the entire conditional distribution of Y. When the shape of the conditional distribution varies nonlinearly with the predictors, you can include polynomial or spline effects in the model.

Table 2 summarizes important differences between standard linear regression and quantile regression.

Table 2 Comparison of Linear Regression with Quantile Regression

Linear Regression	Quantile Regression
Predicts the conditional mean $E[Y\|X[$	Predicts conditional quantiles $Q_\tau[Y\|X]$
Applies even with small data	Needs sufficient data
Can assume normality	Does not assume a parametric distribution
Sensitive to outliers	Robust to outliers
Computationally inexpensive	Computationally intensive

For many years, quantile regression was impractical because its computational cost was too high when the number of observations was sufficiently large for accurate prediction of quantiles, especially in the tails. Today, however, quantile regression is quite practical—even for very large data—with the algorithms that are available in the QUANTREG and QUANTSELECT procedures. Quantile regression can reveal the effects of predictors on different parts of the response distribution, and it can yield valuable insights in applications such as risk management, where useful information lies in the tails.

The QUANTSELECT procedure performs effect selection for quantile regression. Like the GLMSELECT procedure, it is designed primarily for effect selection, and it offers similar methods of effect selection. The HPQUANTSELECT procedure is a high-performance procedure that provides functionality similar to that of PROC QUANTSELECT for building quantile regression models. See Rodriguez and Yao (2017) for applications of the QUANTREG and QUANTSELECT procedures.

Building Regression Models for Categorical Responses

The LOGISTIC procedure has long been the primary SAS/STAT procedure for analyzing logistic regression models, and it provides some functionality for building models. In contrast, the HPLOGISTIC procedure is a new—and relatively unknown—high-performance procedure that is designed specifically for fitting and building logistic regression models. In particular, PROC HPLOGISTIC provides advantages for building predictive models from large data, as summarized in Table 3.

Table 3 Comparison of Model Building Capability in PROC HPLOGISTIC and PROC LOGISTIC

HPLOGISTIC Procedure	LOGISTIC Procedure
Provides forward, backward, fast backward, and stepwise methods of effect selection	Provides forward, backward, fast backward, and stepwise methods of effect selection
Provides selection criteria based on information criteria, validation, and significance level of score test	Provides selection criteria based on significance level of score test
Partitions the data into subsets for model training, validation, and testing	Uses all the data for model fitting and inference
Creates SAS code for scoring new data	Creates SAS code for scoring new data and provides a variety of postfitting analysis
Runs in single-machine or distributed mode; is multi-threaded	Runs in single-machine mode; is single-threaded

When you have a response variable that is either categorical or continuous and can be described by a generalized linear model, you can fit or build the model by using the HPGENSELECT procedure. Although the GENMOD procedure is well known as a tool for fitting generalized linear models, it does not provide model selection (note that PROC GENMOD has been enhanced with specialized methods for inferential analysis, as discussed on page 7). Table 4 summarizes the differences between the two procedures.

Table 4 Comparison of PROC HPGENSELECT and PROC GENMOD

HPGENSELECT Procedure	GENMOD Procedure
Fits and builds generalized linear models	Fits generalized linear models
Analyzes large to massive data	Analyzes moderate to large data
Designed for predictive modeling	Designed for inferential analysis
Partitions the data into subsets for model training, validation, and testing	Uses all the data for model fitting and inference
Creates code for scoring new data	Creates code for scoring new data and offers a variety of postfitting analyses
Runs in single-machine or distributed mode; is multi-threaded	Runs in single-machine mode; is single-threaded

The HPGENSELECT procedure provides models for standard response distributions in the exponential family, including the binary, binomial, gamma, inverse Gaussian, normal, Poisson, and Tweedie distributions. In addition, the procedure provides multinomial models for ordinal and nominal responses, and it fits zero-inflated Poisson and negative binomial models for count data. For effect selection, the HPGENSELECT procedure provides backward elimination, forward selection, stepwise regression, and the group lasso method. See Johnston and Rodriguez (2015) for an introduction to the HPGENSELECT procedure.

Generalized additive models are extensions of generalized linear models in which the predictors are semiparametric. This means that, in addition to linear predictors, you can specify additive spline terms that characterize nonlinear dependency structures which are either unknown or too complex to be described by parametric terms.

The GAMPL procedure is a new high-performance procedure that fits generalized additive models by using low-rank regression splines (Wood 2003, 2006). PROC GAMPL does not provide effect selection, but it does produce plots that you can use to explore the additive effects of the spline components. These plots can suggest parametric effects—such as quadratic polynomials—for models that you can then build with the HPGENSELECT procedure.

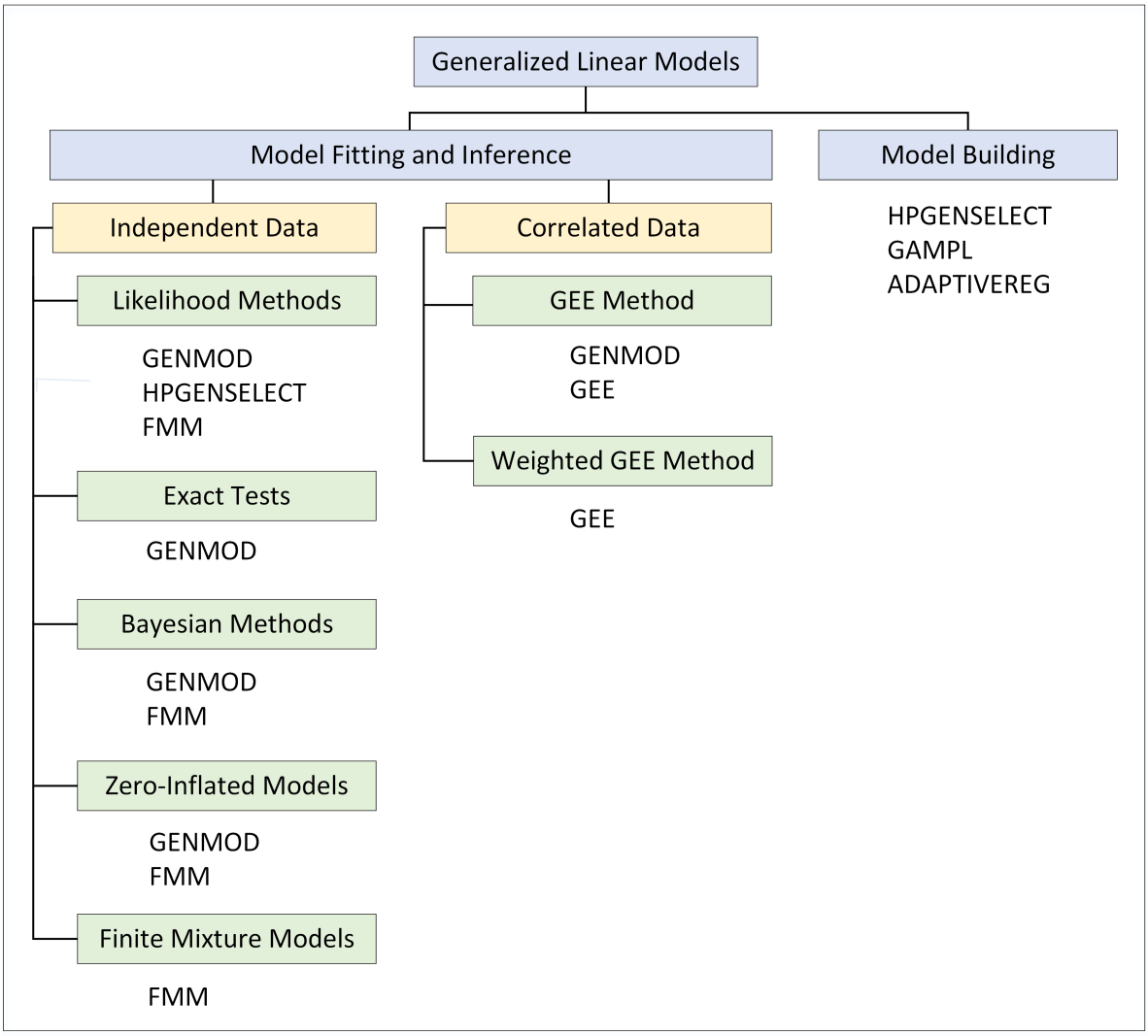
You might be familiar with the earlier GAM procedure for fitting generalized additive models. PROC GAMPL implements newer approaches, such as penalized likelihood estimation, a modified performance iteration method (Wood 2004) and the outer iteration method (Wood 2006). As a result, it provides greatly improved performance for large data.

The ADAPTIVEREG procedure fits response variables with distributions in the exponential family, including the binomial, gamma, inverse Gaussian, normal, negative binomial, and Poisson distributions. The predictor is nonparametric and is constructed from regression splines. The procedure is based on an approach due to Friedman (1991), which constructs spline basis functions in an adaptive way by automatically selecting appropriate knot values for different variables. The approach creates an overfitted model and then prunes it with backward selection. You can use the ADAPTIVEREG procedure to model complex, unknown relationships between the predictors and the response. See Kuhfeld and Cai (2013) for an introduction.

Inferential Analysis of Generalized Linear Models

This section describes procedure enhancements and new procedures for model fitting and inference within the framework of generalized linear models. New procedures for building generalized linear models are described in the preceding section. Figure 3 presents a high-level view of all these procedures.

Figure 3 Procedures for Generalized Linear Models



Generalized linear models assume a parametric response distribution that is in the exponential family. The linear predictor is defined in the same way as for general linear models, and a specified link function expresses how the expected value of the response relates to the linear predictor. Table 5 describes these components.

Table 5 Components of Generalized Linear Models

Component	Description
Linear predictor	Effects involving continuous or classification variables
Link function	Log, logit, inverse, inverse square, and so on
Response distribution	Binary, binomial, gamma, inverse Gaussian, negative binomial, normal, Poisson, Tweedie

A number of widely used statistical models are generalized linear models, including standard linear models with normal errors, logistic regression models for binary data, and log-linear models for multinomial data. You can formulate many other models as generalized linear models by selecting an appropriate response distribution and link function.

The GENMOD procedure is by far the most familiar SAS/STAT procedure for fitting and analyzing generalized linear models. In addition to the models represented in Table 5, PROC GENMOD fits the following extensions:

- models with multinomial response distributions
- models with zero-inflated negative binomial and zero-inflated Poisson response distributions
- models for correlated responses, which the procedure fits by the generalized estimating equation (GEE) method (Liang and Zeger 1986)

Enhancements of the GENMOD Procedure

The GENMOD procedure now provides additional methods of fitting generalized linear models:

- You can request a Bayesian analysis by using the BAYES statement. The model parameters are then treated as random variables, and inferences are based on the posterior distributions of the parameters. The BAYES statement provides a convenient syntax for specifying built-in prior distributions and for requesting credible intervals and summaries of the posterior samples.

  Bayesian analysis does not rely on asymptotic approximations, as do likelihood methods. Another benefit of Bayesian analysis is that the results have intuitive interpretations. On the other hand, you must think carefully about your selection of priors, because these can heavily influence the posterior distributions and there is no single correct way to select a prior. Furthermore, you must assess whether the Markov chain that generated the posterior distribution reached stationarity. The BAYES statement produces convergence diagnostics for making this assessment.

- You can request exact conditional Poisson regression, as well as exact binary logistic regression, by using the EXACT statement. Exact conditional inference is based on generating the conditional distribution for the sufficient statistics for the parameters of interest (Cox 1970). This approach is useful in situations involving small samples or small cell counts, where asymptotic properties of maximum likelihood estimation do not apply. The EXACT statement provides exact tests of the parameters for specified effects.

- You can request zero-inflated Poisson regression models or zero-inflated negative binomial regression models with the ZEROMODEL statement. These models are useful when you encounter overdispersion in count data, assuming it results from a process that produces more zero counts than can be explained by the corresponding standard model. An overdispersion diagnostic plot is available for zero-inflated models; it plots the predicted variance as a function of the predicted mean for a zero-inflated response.

Finite Mixture Models

The FMM procedure fits mixtures of generalized linear models by both maximum likelihood and Bayesian techniques, and it models the effects of covariates on both the component distributions and the mixing probabilities. Finite mixture models enable you to account for heterogeneity and overdispersion in your data with a flexible representation that describes the data distribution as a mixture of known distributions.

The FMM procedure provides CLASS and MODEL statements that are familiar from other procedures such as the GLM and GENMOD procedures, and it provides a BAYES statement for requesting built-in Bayesian analysis. The FMM procedure offers a broad selection of distribution functions and automated model selection methods. Kessler and McDowell (2012) provide an introduction to the FMM procedure.

Weighted Methods of Analyzing Missing Data in Longitudinal Studies

Studies of longitudinal data are prevalent in fields such as public health, medical research, and social science. Multiple measurements are taken on the same subject over time in order to discover changes in the response over time and the relationship of changes to covariates (Fitzmaurice, Laird, and Ware 2011). Marginal models are used when population-averaged effects are of interest, and the regression parameters are commonly estimated by the GEE method, which is implemented in the GENMOD procedure.

Missing observations caused by dropouts are a particular concern in longitudinal studies. When the analysis is restricted to complete cases and missingness of responses depends on previous responses, the standard GEE approach can produce biased parameter estimates. The GEE procedure implements inverse probability-weighted GEE methods that account for dropouts under the assumption that data are missing at random (MAR); see Robins and Rotnitzky (1995) and Preisser, Lohman, and Rathouz (2002). These methods can produce unbiased estimates.

Table 6 summarizes the differences between the standard and weighted GEE methods.

Table 6 Comparison of GEE Methods

	Standard GEE Method	**Weighted GEE Method**
Procedure	GENMOD and GEE	GEE
Data analyzed	Available cases	Available cases
Model specification	Response model Correlation	Response model Correlation Missingness model
Inference assuming data missing completely at random (MCAR)	Valid even when the correlation is misspecified	Valid even when the correlation is misspecified
Inference assuming data missing at random (MAR)	Not generally valid	Valid even when the correlation is misspecified

In addition to GEE methods, the GEE procedure supports the alternating logistic regressions (ALR) algorithm, which is available in the GENMOD procedure and models the association between pairs of responses by using log odds ratios instead of correlations (Carey, Zeger, and Diggle 1993). The GEE procedure provides three methods that are not available in the GENMOD procedure:

- weighted GEE methods
- the ALR method for ordinal multinomial data
- the generalized logit model for nominal multinomial data. Only the independent working correlation structure is supported.

See Lin and Rodriguez (2014) for an introduction to the GEE procedure.

Survival Analysis

Survival analysis deals with time-to-event data that are incomplete due to censoring or competing risks:

- Observations are right-censored when the only information at a given time is that the event of interest has not yet occurred. Likewise, observations are left-censored when the only information at a given time is that the event has already occurred. Observations are interval-censored when the only information is that the event has occurred within a known interval.
- Competing risks are events that impede the observation of the event of interest or that modify the probability that this event will occur. For example, in cardiovascular studies, deaths from other causes such as cancer are considered competing risks.

SAS/STAT software provides specialized procedures for performing survival analysis for right-censored data. Three of these, the LIFETEST, LIFEREG, and PHREG procedures, are particularly well known because they have been available for many years.

The LIFETEST procedure specializes in estimation and hypothesis testing; it computes the Kaplan-Meier estimate of a survivor function and provides the log-rank test for comparing survival curves between groups of observations. The LIFEREG and PHREG procedures specialize in modeling the outcome of interest, but with a clear distinction: PROC LIFEREG fits parametric accelerated failure time (AFT) models, while PROC PHREG fits semiparametric regression models, including the Cox proportional hazards model.

Figure 4 presents a high-level view of all the survival analysis procedures that perform estimation and hypothesis testing.

Figure 4 Survival Analysis Procedures for Estimation and Hypothesis Testing

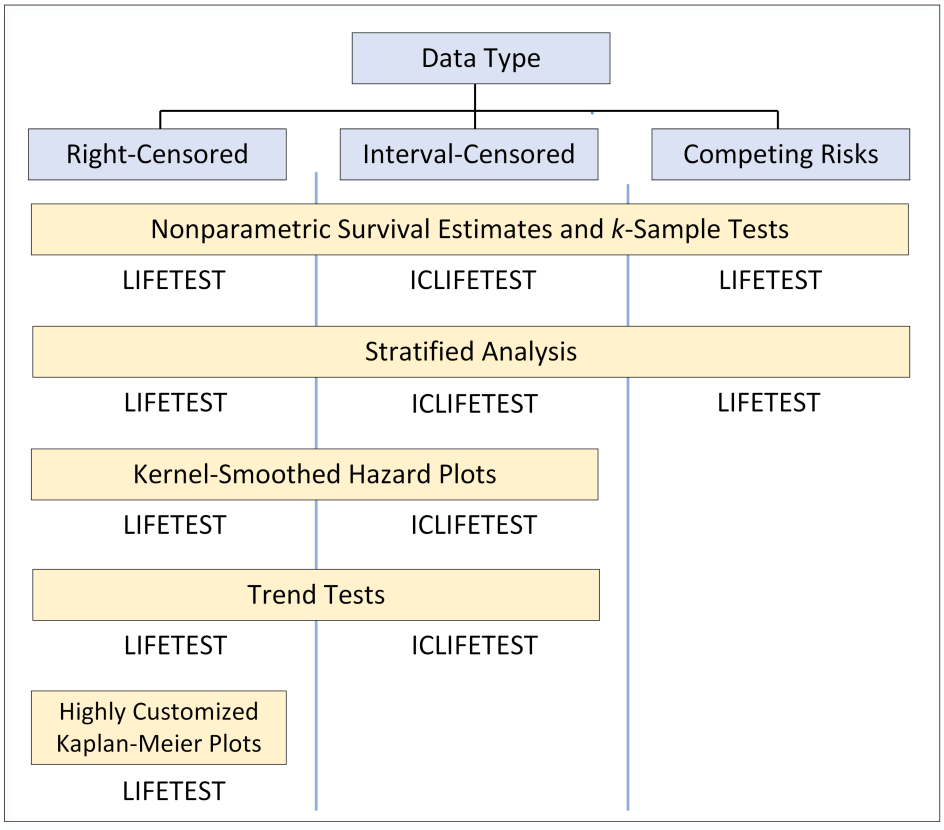
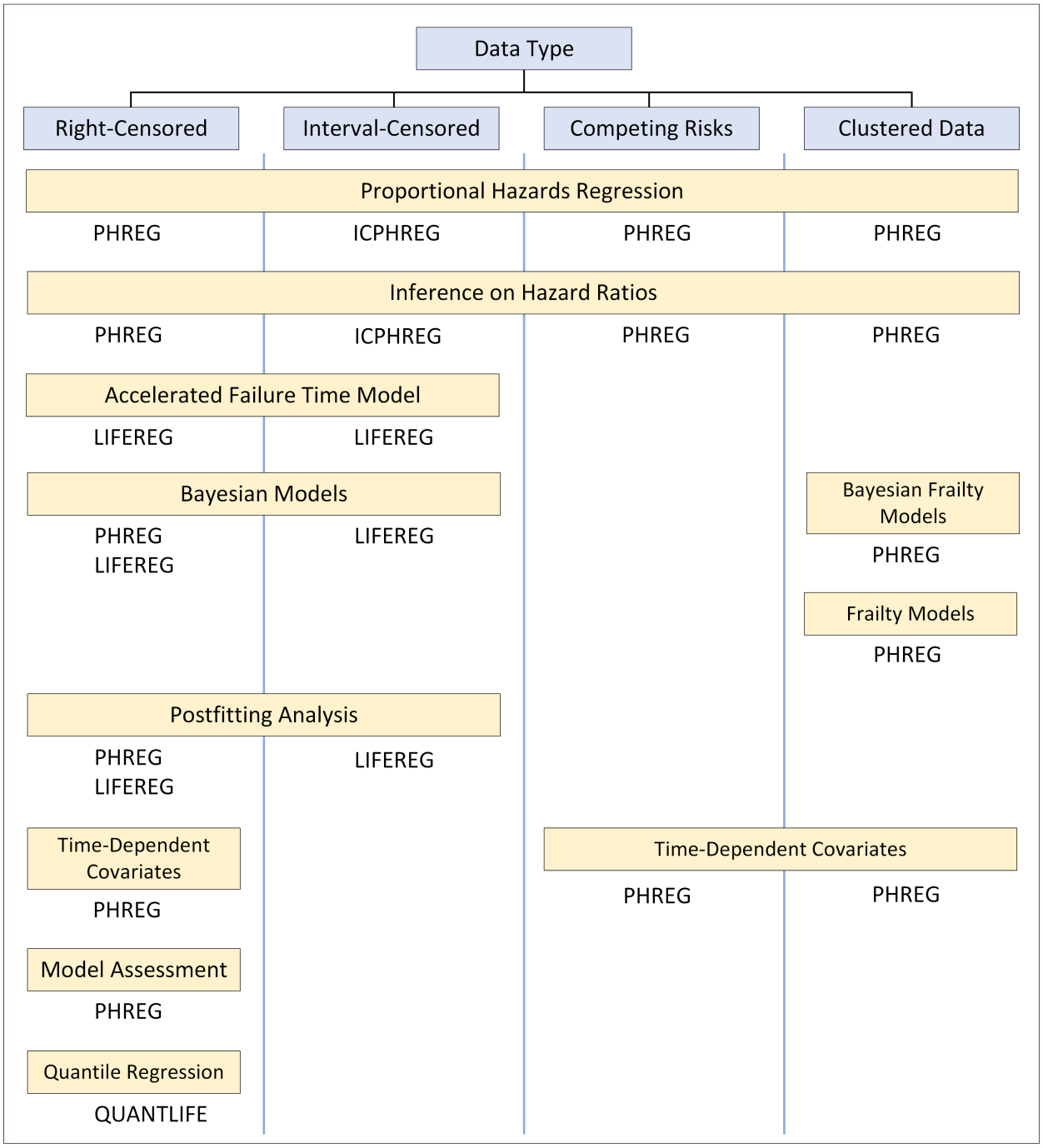


Figure 5 presents a high-level view of all the survival analysis procedures that perform modeling.

Figure 5 Survival Analysis Procedures for Modeling



Enhancements for Survival Analysis

In recent years, the LIFETEST, LIFEREG, and PHREG procedures have been enhanced with specialized methods of analyzing right-censored data:

- The BAYES statement in PROC LIFEREG and PROC PHREG requests a Bayesian analysis of the model; it provides a variety of priors and produces diagnostic plots for convergence assessment.
- The HAZARDRATIO statement in PROC PHREG enables you to request hazard ratios for variables in the model at customized settings, and it provides confidence limits for hazard ratios.

- The PHREG procedure provides methods of model assessment, including the Schemper-Henderson statistic, two versions of concordance statistics, and time-dependent receiver-operator characteristic (ROC) curves.
- The Kaplan-Meier plot that is produced by PROC LIFETEST is now highly customizable through the use of procedure options, graph template modifications, and style template modifications. Kuhfeld and So (2013) provide examples of these approaches.

The survival analysis capabilities of SAS/STAT have also been extended to handle types of time-to-event data other than right-censored data:

- Two new procedures, PROC ICLIFETEST and PROC ICPHREG, specialize in the analysis of interval-censored data and serve as counterparts of PROC LIFETEST and PROC PHREG.
- The LIFETEST and PHREG procedures have been enhanced with specialized methods that analyze the cumulative incidence function (CIF) for competing risks data.
- The RANDOM statement in PHREG provides facilities for fitting frailty models, which handle correlations between failures in clustered data.

Another new procedure, the QUANTLIFE procedure, uses quantile regression to analyze survival data and is particularly useful for modeling heterogeneous data.

Table 7 summarizes the main distinctions among the six procedures that are now available for survival analysis.

Table 7 Comparison of Procedures for Survival Analysis

Procedure	Focus	Inferential Approach	Modeling with Covariates	Censoring Scheme
LIFETEST	Survival function	Nonparametric	No	Right
ICLIFETEST	Survival function	Nonparametric	No	Interval
LIFEREG	Lifetime	Parametric	Yes	Right, left, interval
PHREG	Hazard function	Semiparametric	Yes	Right
ICPHREG	Hazard function	Parametric	Yes	Interval
QUANTLIFE	Lifetime	Semiparametric	Yes	Right

The next three sections explain the benefits of new tools for interval-censored analysis, competing risks analysis, and quantile regression analysis of survival data.

Interval-Censored Analysis

Interval censoring occurs in medical and health studies that involve periodic follow-ups on patients. For example, in acquired immune deficiency syndrome (AIDS) trials, the determination of disease onset is usually based on blood testing, which can only be performed periodically. Interval censoring generalizes left and right censoring. When the left endpoint is zero, the interval represents a left-censored observation. When the right endpoint is infinity, the interval represents a right-censored observation.

Specialized methods of handling interval-censored data have emerged (Turnbull 1976; Finkelstein 1986; Groeneboom and Wellner 1992) and are available in the ICLIFETEST and ICPHREG procedures. These methods offer advantages over midpoint imputation, an ad hoc approach that applies methods for right-censored data to the midpoint of the censoring interval. Chen (2009a) demonstrates that the imputation approach is biased and less efficient than the specialized methods, especially for infrequent or imbalanced assessment.

The ICLIFETEST and ICPHREG procedures resemble the LIFETEST and PHREG procedures in their objectives, but their functionality is relatively limited because fewer methods have been introduced for interval-censored data. The ICLIFETEST procedure provides nonparametric methods of estimating survival functions and statistical testing. The ICPHREG procedure fits proportional hazards regression models and provides inferences based on these models.

For an introduction to the ICLIFETEST procedure, see Guo, So, and Johnston (2014).

Analysis of Competing Risks

Recent enhancements of the LIFETEST and PHREG procedures provide state-of-the-art techniques for the analysis of right-censored data with competing risks. You can use the LIFETEST procedure to perform nonparametric analyses and the PHREG procedure to perform regression analyses.

The concepts of a survival function and a hazard function, which form the basis for standard survival analysis, are inadequate for studying competing risks because once a subject experiences an event other than the event of interest, information about the latter can no longer be ascertained reliably. Instead, the analysis of competing risks is based on the analogous concepts of a cumulative incidence function (CIF) and a cause-specific hazard (CSH) function. The CIF, which is defined as the probability subdistribution function of failure from a specific cause, characterizes the occurrence of a cause-specific outcome over time. The CSH function measures the instantaneous rate of failing from a specific cause in the presence of other causes.

By treating observations of other types of events as censored observations of the event of interest, you can analyze the CSH function for the cause of interest by using certain standard methods, such as the log-rank test in the LIFETEST procedure and Cox regression in the PHREG procedure. However, to analyze the CIF, you need specialized methods, such as those recently provided in the LIFETEST procedure and the PHREG procedure.

The model of Fine and Gray (1999), implemented in the PHREG procedure, extends the Cox model to the CIF setting. The test due to Gray (1988) serves as a counterpart of the log-rank test for testing the equality of CIFs, and is available in the LIFETEST procedure along with a nonparametric estimator of the CIF. You can request CIF analyses in the LIFETEST and PHREG procedures by specifying the code that represents the cause of interest with the EVENTCODE= option. So, Lin, and Johnston (2015) explain how to perform competing risks regression by using the PHREG procedure.

Survival Analysis Based on Quantile Regression

The quantile regression approach to survival analysis, now available in the QUANTLIFE procedure, is useful when you are modeling the survival time and the effects of covariates on the lifetime distribution differ with the covariates. You can use PROC QUANTLIFE to explore such effects—for example, when the variation in the lifetime increases with a continuous covariate.

To decide when to use the QUANTLIFE procedure, you should understand how the quantile regression approach compares with standard methods available in the LIFETEST, LIFEREG, and PHREG procedures. Each method has its advantages and limitations.

Quantile regression is a distribution-free approach in the sense that inference about the regression parameters for a particular quantile of the lifetime depends only on the conditional distribution near that quantile. By comparison, the AFT model in the LIFEREG procedure is more restrictive in its parametric assumption.

Both the Cox proportional model in the PHREG procedure and the AFT model involve an iid error assumption under a suitable transformation of the survival time (Koenker and Geling 2001). This means that covariate effects can shift the location but not the shape of the conditional density for the transformed lifetime. The additional flexibility of quantile regression for modeling the shape can be important when, for example, you are concerned about treatment effects on longer lifetimes.

The QUANTLIFE and LIFEREG procedures both use a regression method to model the lifetime. The LIFEREG procedure provides an efficient estimator for the regression parameters if you are willing to assume a parametric distribution for the lifetime. The regression coefficients computed by the LIFEREG procedure are interpreted as the effect on the mean of the lifetime, and the regression coefficients computed by the QUANTLIFE procedure apply to specified quantiles of the lifetime.

Unlike the QUANTLIFE procedure, the PHREG procedure models the hazard function. Both of these procedures are semiparametric, but in different ways. The Cox model requires no parametric assumption about the baseline hazard function. Another advantage of the Cox model is that it can incorporate time-dependent covariates.

Lin and Rodriguez (2013) provide an introduction to the QUANTLIFE procedure.

Analysis of Mixed Models

When you fit statistical models to data, it is common to assume that all the observations are uncorrelated. The standard linear model procedures—GLM, REG, GLMSELECT, and HPREG—all make this assumption. But when that assumption is violated, it can have a huge impact on the validity of the inferences that you make, and that is when you need mixed models. Mixed models incorporate both fixed effects, which affect only the mean of the response, and random effects, which relate to the covariance between observations.

The MIXED procedure is the flagship SAS/STAT procedure for dealing with linear mixed models. PROC MIXED extends the versatile features for specifying fixed effects in linear models that you find in many SAS/STAT procedures with similarly versatile features for specifying how random effects induce correlation. Likewise, PROC MIXED extends the inferential tools for linear models with fixed effects—for example, Type 3 tests, tests for linear contrasts, and LS-means—with tests and methods that are appropriate for correlation structures.

Although PROC MIXED provides the generality that you need for model estimation and postfit inference, it is not computationally efficient for certain important special cases, including the following:

- linear mixed models with thousands of levels for the fixed and/or random effects
- linear mixed models with hierarchically nested fixed and/or random effects, possibly with hundreds or thousands of levels at each stage of the hierarchy

For these models, which are large and sparse, you need specialized methods. The HPMIXED procedure implements these methods by taking advantage of sparse matrix techniques. PROC HPMIXED does sacrifice certain inferential tools that are available in PROC MIXED but cannot be implemented sparsely. However, if your mixed models fall into these special categories, PROC HPMIXED can often run much faster than PROC MIXED. Wang and Tobias (2009) and Kiernan, Tao, and Gibbs (2016) describe situations that call for the methods in PROC HPMIXED, and they discuss the substantial gains in performance that it provides.

Both the MIXED and HPMIXED procedures deal with responses and random effects that are assumed to be normally distributed. If your response has a nonnormal distribution that belongs to the exponential family—for example, if it has a binary logistic or Poisson distribution—then you need the GLIMMIX procedure, which fits generalized linear mixed models. PROC GLIMMIX can use pseudo-likelihood or marginal maximum likelihood estimation to fit mixed models with a variety of nonnormal error distributions. Schabenberger (2005) outlines the capabilities of PROC GLIMMIX and provides examples that demonstrate its great flexibility for modeling correlated data.

If you need to fit linear mixed models, then one of the three procedures discussed so far in this section—MIXED, HPMIXED, and GLIMMIX—is what you need. But what if you need to fit a random coefficients model in which the coefficients enter the model nonlinearly? Or what if you are fitting a nonlinear mixed model for a pharmacokinetic application where the likelihood depends on solving a system of differential equations? The NLMIXED procedure can fit such models; it enables you to specify a distribution for the response, conditional on the random effects, that has a standard form—such as normal, binary, or Poisson—or a general form that you express with SAS programming statements.

Although both PROC GLIMMIX and PROC NLMIXED enable you to fit models for nonnormal responses, the estimation methods they use require the random effects to be normally distributed. If you need to fit models with nonnormal random effects, then you need to move beyond likelihood methods and consider Bayesian methods, for which the MCMC procedure is the most versatile procedure in SAS/STAT.

Instead of maximizing a likelihood, the Bayesian approach treats all unknown quantities in a model, including both the fixed and random effects, as random variables. The objective is to estimate the joint posterior distribution, often by using the Markov chain Monte Carlo approach (Gelfand et al. 1990). The marginal distribution of the fixed-effects parameters is obtained by using a numerical Monte Carlo method that is based on the Markov chain samples.

In PROC MCMC, you specify the details of a Bayesian model with a combination of procedure statements (such as the PARMS, PRIOR, MODEL, and RANDOM statements) and SAS programming statements. Like the NLMIXED procedure, the MCMC procedure does not assume linearity and it handles a wide range of models. The complex Bayesian models that you can fit with PROC MCMC include linear, generalized linear, and nonlinear random-effects models.

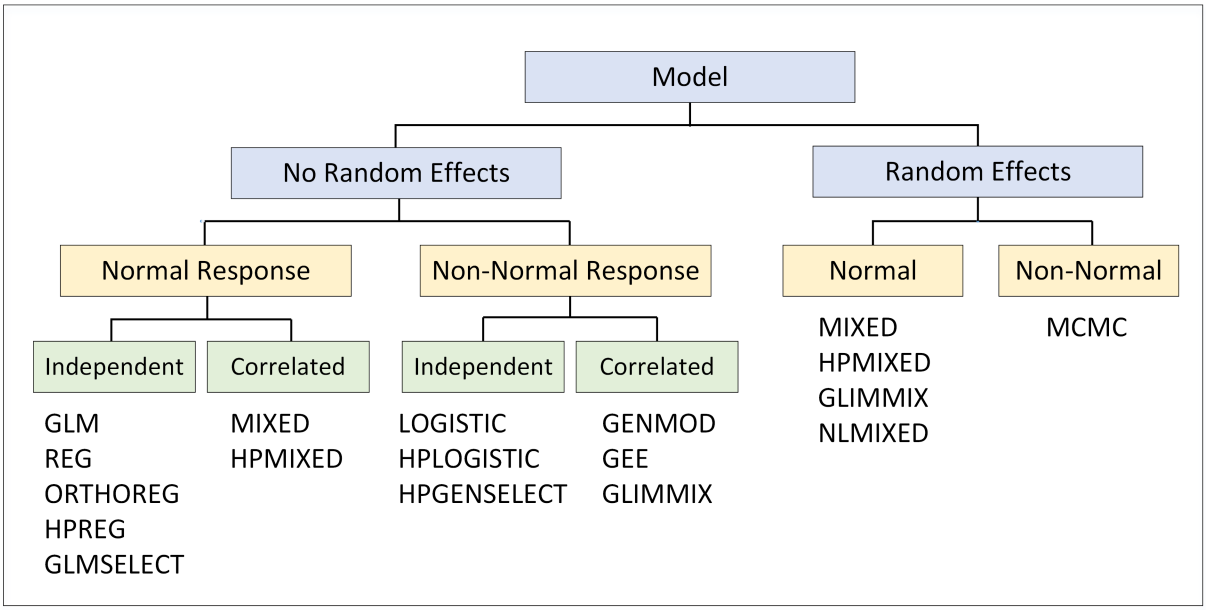
The tutorial papers by Chen (2009b, 2011, 2013) introduce the MCMC procedure. Chen, Brown, and Stokes (2016) offer guidance on using PROC MCMC to perform Bayesian analysis of models for which PROC MIXED and PROC

GLIMMIX implement likelihood methods. Chen and Stokes (2017) illustrate the use of PROC MCMC to fit various multilevel hierarchical models that incorporate complex structures and data dependencies.

In principle, the MCMC procedure is the most general SAS/STAT procedure for analyzing mixed models. It can handle normal data and linear models (like the MIXED procedure), nonnormal data and generalized linear models (like the GLIMMIX, GENMOD, and GEE procedures), and nonlinear models (like the NLMIXED procedure). In addition, the MCMC procedure can handle nonnormal random effects, multilevel random effects, and missing data in ways that are not available with the other procedures. Of course, with this generality you give up much of the convenience and specific analytic features of the more specialized procedures. Thus, the MIXED, HPMIXED, GLIMMIX, and NLMIXED procedures should be your go-to tools for most practical applications of mixed models. The MCMC procedure should be in your toolbox for applications that the other procedures cannot handle, and for situations in which Bayesian modeling is appropriate.

Figure 6 presents a high-level view of procedures for mixed models along with procedures for linear models.

Figure 6 Procedures for Analysis of Linear Models and Mixed Models



Procedures are listed in categories where they provide strengths in practice (in principle, some procedures could be listed in other categories). Procedures within a category have their own unique strengths. For example, the GLM procedure is distinguished from the ORTHOREG, HPREG, and GLMSELECT procedures by its capability for multivariate analysis of variance (MANOVA), which can be regarded as a forerunner to mixed modeling.

Postfitting Analysis of Linear Mixed Models

There is a general-purpose feature for handling linear mixed models that you may not be familiar with—but should be. By using the PLM procedure after you fit the model, you can do additional analysis without rerunning the procedure (either PROC MIXED or PROC GLIMMIX) that you originally used to fit the model and without access to the original data. PROC PLM provides you with two important features:

- Additional analyses, which you can request with the ESTIMATE, LSMEANS, LSMESTIMATE, SLICE, and TEST statements in PROC PLM—even if those statements are not available in the procedure that fit the model.
- Plots of your model results, which you can request with the EFFECTPLOT statement in PROC PLM or the plots that are available through the ESTIMATE, LSMEANS, LSMESTIMATE, and SLICE statements.

In short, PROC PLM enables you to explore the analysis of your model without refitting it—and that makes it a procedure worth learning about. Tobias and Cai (2010) explain how PROC PLM offers these same features not just for linear mixed models, but for a wide spectrum of linear models that you can fit with other procedures, including the GEE, GENMOD, GLM, GLMSELECT, LIFEREG, LOGISTIC, ORTHOREG, and PHREG procedures.

Summary

This paper describes important new tools that SAS/STAT has added in four areas:

- Regression model building
- Inferential analysis of generalized linear models
- Survival analysis
- Analysis of mixed models

If you are over-relying on the basic tools in these areas—perhaps the well-known procedures that you learned about in school—then you will want to explore the enhancements and new procedures discussed in this paper. Table 8 summarizes the many benefits of adopting these tools in your statistical practice.

Table 8 Benefits of Procedure Enhancements and New Procedures in SAS/STAT Software

Benefit	Method	Procedure
Improved predictive ability and interpretability of regression models	Data partitioning	GLMSELECT, HPREG, HPSPLIT, QUANTSELECT, ADAPTIVEREG, HPLOGISTIC, HPGENSELECT
	Lasso methods and information criteria	GLMSELECT, QUANTSELECT, HPGENSELECT
Regression model building for a variety of response types and for complex dependence structures	Categorical responses	HPLOGISTIC, HPGENSELECT, GAMPL, ADAPTIVEREG
	Quantile regression	QUANTSELECT, HPQUANTSELECT
	Regression trees	HPSPLIT
	Spline effects	GLMSELECT, GAMPL, ADAPTIVEREG
Advantages of Bayesian methods, including model versatility, highly interpretable results, and no requirement of a large sample	Generalized linear models Survival analysis models Finite mixture models Mixed models General Bayesian models	GENMOD LIFEREG, PHREG, MCMC FMM MCMC MCMC
Inference for special generalized linear models	Models for overdispersion Exact methods for small samples	GENMOD, FMM GENMOD
	Weighted GEE methods for dropouts in longitudinal data	GEE
Inference for special types of time-to-event data	Methods for interval-censored data Analysis of competing risks Analysis of heterogeneous data	ICLIFETEST, ICPHREG LIFETEST, PHREG QUANTLIFE
High-performance computing for large data	Regression model building	HPREG, HPSPLIT, HPQUANTSELECT, HPLOGISTIC, HPGENSELECT
	Generalized additive models Regression trees Large, sparse mixed models	GAMPL HPSPLIT HPMIXED

Keeping Up with New Releases of SAS/STAT

The new tools discussed in this paper are only a portion of the many enhancements that you will find in recent releases of SAS/STAT software, which are listed in Table 9. The current release is SAS/STAT 14.2.

Table 9 Recent Releases of SAS/STAT Software

Release	Year	Overview	Base SAS Version
SAS/STAT 12.1	2012	Stokes et al. (2012)	SAS 9.3
SAS/STAT 12.3	2013	Stokes (2013)	SAS 9.4
SAS/STAT 13.1	2013	Rodriguez (2014)	SAS 9.4M1
SAS/STAT 13.2	2014	Stokes, Güneş, and Chen (2014)	SAS 9.4M2
SAS/STAT 14.1	2015	Stokes and Statistical R&D Staff (2015)	SAS 9.4M3
SAS/STAT 14.2	2016	support.sas.com/statistics	SAS 9.4M4

The best place to find out about the enhancements in the release that you have is the chapter "What's New in SAS/STAT" in the online documentation at http://support.sas.com/statdoc/. Also, be sure to visit the Statistics and Operations Research focus area at http://support.sas.com/statistics. There you can watch helpful videos, download overview papers, and subscribe to a quarterly e-newsletter.

REFERENCES

Carey, V., Zeger, S. L., and Diggle, P. J. (1993). "Modelling Multivariate Binary Data with Alternating Logistic Regressions." *Biometrika* 80:517–526.

Chen, C. (2009a). "Empirical Comparison between Conventional Approach and Finkelstein's Method." Paper presented at DIA/FDA/PHRMA PFS Workshop, Oct. 7–9, Bethesda, MD.

Chen, F. (2009b). "Bayesian Modeling Using the MCMC Procedure." In *Proceedings of the SAS Global Forum 2009 Conference*. Cary, NC: SAS Institute Inc. http://support.sas.com/resources/papers/proceedings09/257-2009.pdf.

Chen, F. (2011). "The RANDOM Statement and More: Moving On with PROC MCMC." In *Proceedings of the SAS Global Forum 2011 Conference*. Cary, NC: SAS Institute Inc. http://support.sas.com/resources/papers/proceedings11/334-2011.pdf.

Chen, F. (2013). "Missing No More: Using the MCMC Procedure to Model Missing Data." In *Proceedings of the SAS Global Forum 2013 Conference*. Cary, NC: SAS Institute Inc. https://support.sas.com/resources/papers/proceedings13/436-2013.pdf.

Chen, F., Brown, G., and Stokes, M. (2016). "Fitting Your Favorite Mixed Models with PROC MCMC." In *Proceedings of the SAS Global Forum 2016 Conference*. Cary, NC: SAS Institute Inc. https://support.sas.com/resources/papers/proceedings16/SAS5601-2016.pdf.

Chen, F., and Stokes, M. (2017). "Advanced Hierarchical Modeling with the MCMC Procedure." In *Proceedings of the SAS Global Forum 2017 Conference*. Cary, NC: SAS Institute Inc. https://support.sas.com/resources/papers/proceedings17/SAS478-2017.pdf.

Cohen, R. (2006). "Introducing the GLMSELECT Procedure for Model Selection." In *Proceedings of the Thirty-First Annual SAS Users Group International Conference*. Cary, NC: SAS Institute Inc. http://www2.sas.com/proceedings/sugi31/207-31.pdf.

Cohen, R. (2009). "Applications of the GLMSELECT Procedure for Megamodel Selection." In *Proceedings of the SAS Global Forum 2009 Conference*. Cary, NC: SAS Institute Inc. http://support.sas.com/resources/papers/proceedings09/259-2009.pdf.

Cohen, R., and Rodriguez, R. N. (2013). "High-Performance Statistical Modeling." In *Proceedings of the SAS Global Forum 2013 Conference*. Cary, NC: SAS Institute Inc. http://support.sas.com/resources/papers/proceedings13/401-2013.pdf.

Cox, D. R. (1970). *Analysis of Binary Data*. London: Metheun.

Fine, J. P., and Gray, R. J. (1999). "A Proportional Hazards Model for the Subdistribution of a Competing Risk." *Journal of the American Statistical Association* 94:496–509.

Finkelstein, D. M. (1986). "A Proportional Hazards Model for Interval-Censored Failure Time Data." *Biometrics* 42:845–854.

Fitzmaurice, G. M., Laird, N. M., and Ware, J. H. (2011). *Applied Longitudinal Analysis*. 2nd ed. Hoboken, NJ: John Wiley & Sons.

Friedman, J. H. (1991). "Multivariate Adaptive Regression Splines." *Annals of Statistics* 19:1–67.

Gelfand, A. E., Hills, S. E., Racine-Poon, A., and Smith, A. F. M. (1990). "Illustration of Bayesian Inference in Normal Data Models Using Gibbs Sampling." *Journal of the American Statistical Association* 85:972–985.

Gibbs, P., Tobias, R., Kiernan, K., and Tao, J. (2013). "Having an EFFECT: More General Linear Modeling and Analysis with the New EFFECT Statement in SAS/STAT Software." In *Proceedings of the SAS Global Forum 2013 Conference*. Cary, NC: SAS Institute Inc. https://support.sas.com/resources/papers/proceedings13/437-2013.pdf.

Gray, R. J. (1988). "A Class of K-Sample Tests for Comparing the Cumulative Incidence of a Competing Risk." *Annals of Statistics* 16:1141–1154.

Groeneboom, P., and Wellner, J. A. (1992). *Information Bounds and Nonparametric Maximum Likelihood Estimation*. Basel: Birkhäuser.

Günes, F. (2015). "Penalized Regression Methods for Linear Models in SAS/STAT." In *Proceedings of the SAS Global Forum 2015 Conference*. Cary, NC: SAS Institute Inc. http://support.sas.com/rnd/app/stat/papers/2015/PenalizedRegression_LinearModels.pdf.

Guo, C., So, Y., and Johnston, G. (2014). "Analyzing Interval-Censored Data with the ICLIFETEST Procedure." In *Proceedings of the SAS Global Forum 2014 Conference*. Cary, NC: SAS Institute Inc. http://support.sas.com/resources/papers/proceedings14/SAS279-2014.pdf.

Hastie, T., Tibshirani, R., and Wainwright, M. (2015). *Statistical Learning with Sparsity: The Lasso and Generalizations*. Boca Raton, FL: CRC Press.

Johnston, G., and Rodriguez, R. N. (2015). "Introducing the HPGENSELECT Procedure: Model Selection for Generalized Linear Models and More." In *Proceedings of the SAS Global Forum 2015 Conference*. Cary, NC: SAS Institute Inc. http://support.sas.com/resources/papers/proceedings15/SAS1742-2015.pdf.

Kessler, D., and McDowell, A. (2012). "Introducing the FMM Procedure for Finite Mixture Models." In *Proceedings of the SAS Global Forum 2012 Conference*. Cary, NC: SAS Institute Inc. http://support.sas.com/resources/papers/proceedings12/328-2012.pdf.

Kiernan, K., Tao, J., and Gibbs, P. (2016). "Tips and Strategies for Mixed Modeling with SAS/STAT Procedures." In *Proceedings of the SAS Global Forum 2016 Conference*. Cary, NC: SAS Institute Inc. http://support.sas.com/resources/papers/proceedings16/SAS6403-2016.pdf.

Koenker, R., and Bassett, G. W. (1978). "Regression Quantiles." *Econometrica* 46:33–50.

Koenker, R., and Geling, O. (2001). "Reappraising Medfly Longevity: A Quantile Regression Survival Analysis." *Journal of the American Statistical Association* 96:458–468.

Kuhfeld, W., and Cai, W. (2013). "Introducing the New ADAPTIVEREG Procedure for Adaptive Regression." In *Proceedings of the SAS Global Forum 2013 Conference*. Cary, NC: SAS Institute Inc. https://support.sas.com/resources/papers/proceedings13/457-2013.pdf.

Kuhfeld, W. F., and So, Y. (2013). "Creating and Customizing the Kaplan-Meier Survival Plot in PROC LIFETEST." In *Proceedings of the SAS Global Forum 2013 Conference*. Cary, NC: SAS Institute Inc. https://support.sas.com/resources/papers/proceedings13/427-2013.pdf.

Liang, K.-Y., and Zeger, S. L. (1986). "Longitudinal Data Analysis Using Generalized Linear Models." *Biometrika* 73:13–22.

Lin, G., and Rodriguez, R. N. (2013). "Using the QUANTLIFE Procedure for Survival Analysis." In *Proceedings of the SAS Global Forum 2013 Conference*. Cary, NC: SAS Institute Inc. https://support.sas.com/resources/papers/proceedings13/421-2013.pdf.

Lin, G., and Rodriguez, R. N. (2014). "Weighted Methods for Analyzing Missing Data with the GEE Procedure." In *Proceedings of the SAS Global Forum 2014 Conference*. Cary, NC: SAS Institute Inc. http://support.sas.com/resources/papers/proceedings14/SAS166-2014.pdf.

Preisser, J. S., Lohman, K. K., and Rathouz, P. J. (2002). "Performance of Weighted Estimating Equations for Longitudinal Binary Data with Drop-Outs Missing at Random." *Statistics in Medicine* 21:3035–3054.

Robins, J. M., and Rotnitzky, A. (1995). "Semiparametric Efficiency in Multivariate Regression Models with Missing Data." *Journal of the American Statistical Association* 90:122–129.

Rodriguez, R. N. (2014). "SAS/STAT 13.1: Round-Up." In *Proceedings of the SAS Global Forum 2014 Conference*. Cary, NC: SAS Institute Inc. http://support.sas.com/resources/papers/proceedings14/SAS181-2014.pdf.

Rodriguez, R. N., and Yao, Y. (2017). "Five Things You Should Know about Quantile Regression." In *Proceedings of the SAS Global Forum 2017 Conference*. Cary, NC: SAS Institute Inc. http://support.sas.com/resources/papers/proceedings17/SAS525-2017.pdf.

Schabenberger, O. (2005). "Introducing the GLIMMIX Procedure for Generalized Linear Mixed Models." In *Proceedings of the Thirtieth Annual SAS Users Group International Conference*. Cary, NC: SAS Institute Inc. http://www2.sas.com/proceedings/sugi30/196-30.pdf.

So, Y., Lin, G., and Johnston, G. (2015). "Using the PHREG Procedure to Analyze Competing-Risks Data." In *Proceedings of the SAS Global Forum 2015 Conference*. Cary, NC: SAS Institute Inc. http://support.sas.com/resources/papers/proceedings15/SAS1855-2015.pdf.

Stokes, M. (2013). "Current Directions in SAS/STAT Software Development." In *Proceedings of the SAS Global Forum 2013 Conference*. Cary, NC: SAS Institute Inc. http://support.sas.com/resources/papers/proceedings13/432-2013.pdf.

Stokes, M., Chen, F., Yuan, Y., and Cai, W. (2012). "Look Out: After SAS/STAT 9.3 Comes SAS/STAT 12.1!" In *Proceedings of the SAS Global Forum 2012 Conference*. Cary, NC: SAS Institute Inc. http://support.sas.com/resources/papers/proceedings12/313-2012.pdf.

Stokes, M., Güneş, F., and Chen, F. (2014). "An Introduction to Bayesian Analysis with SAS/STAT Software." In *Proceedings of the SAS Global Forum 2014 Conference*. Cary, NC: SAS Institute Inc. http://support.sas.com/resources/papers/proceedings14/SAS400-2014.pdf.

Stokes, M., and Statistical R&D Staff (2015). "SAS/STAT 14.1: Methods for Massive, Missing, or Multifaceted Data." In *Proceedings of the SAS Global Forum 2015 Conference*. Cary, NC: SAS Institute Inc. http://support.sas.com/resources/papers/proceedings15/SAS1940-2015.pdf.

Tibshirani, R. (1996). "Regression Shrinkage and Selection via the Lasso." *Journal of the Royal Statistical Society, Series B* 58:267–288.

Tobias, R. D., and Cai, W. (2010). "Introducing PROC PLM and Postfitting Analysis for Very General Linear Models in SAS/STAT 9.22." In *Proceedings of the SAS Global Forum 2010 Conference*. Cary, NC: SAS Institute Inc. http://support.sas.com/resources/papers/proceedings10/258-2010.pdf.

Turnbull, B. W. (1976). "The Empirical Distribution Function with Arbitrarily Grouped, Censored, and Truncated Data." *Journal of the Royal Statistical Society, Series B* 38:290–295.

Wang, T., and Tobias, R. D. (2009). "All the Cows in Canada: Massive Mixed Modeling with the HPMIXED Procedure in SAS 9.2." In *Proceedings of the SAS Global Forum 2009 Conference*. Cary, NC: SAS Institute Inc. https://support.sas.com/resources/papers/proceedings09/256-2009.pdf.

Wood, S. (2003). "Thin Plate Regression Splines." *Journal of the Royal Statistical Society, Series B* 65:95–114.

Wood, S. (2004). "Stable and Efficient Multiple Smoothing Parameter Estimation for Generalized Additive Models." *Journal of the American Statistical Association* 99:673–686.

Wood, S. (2006). *Generalized Additive Models*. Boca Raton, FL: Chapman & Hall/CRC.

Acknowledgments

The authors are grateful to Susan Rodriguez for editorial assistance and to Weijie Cai, Fang Chen, Changbin Guo, Michael Lamm, and Maura Stokes for helpful suggestions.

Paper SAS4900-2016

Statistical Model Building for Large, Complex Data: Five New Directions in SAS/STAT® Software

Robert N. Rodriguez, SAS Institute Inc.

Abstract

The increasing size and complexity of data in research and business applications require a more versatile set of tools for building explanatory and predictive statistical models. In response to this need, SAS/STAT® software continues to add new methods.

This paper provides a high-level tour of five modern approaches to model building that are available in recent releases of SAS/STAT: building sparse regression models with the GLMSELECT procedure, building generalized linear models with the HPGENSELECT procedure, building quantile regression models with the QUANTSELECT procedure, fitting generalized additive models with the GAMPL procedure, and building classification and regression trees with the HPSPLIT procedure. The paper reviews the key concepts of each approach and illustrates the syntax and output of each procedure with a basic example.

Introduction

One of the most frequently asked questions in statistical practice is the following: "I have hundreds of variables—even thousands. Which should I include in my regression model?" This paper presents overviews of five modern approaches to selecting the effects in a regression model when you need a model that is interpretable or that accurately predicts future data. When interpretability is the goal, you need inferential results, such as standard errors and p-values, to decide which effects are important. When prediction is the goal, you need to evaluate the accuracy of prediction and assess whether it could be improved by a sparser, more parsimonious model.

The paper is organized into five main sections, one for each approach:

- Building Sparse Regression Models with the GLMSELECT Procedure
- Building Generalized Linear Models with the HPGENSELECT Procedure
- Building Quantile Regression Models with the QUANTSELECT Procedure
- Fitting Generalized Additive Models with the GAMPL Procedure
- Building Classification and Regression Tree Models with the HPSPLIT Procedure

These approaches are implemented in new or enhanced procedures that are available in recent releases of SAS/STAT software. The paper introduces each procedure, explains key concepts, and illustrates syntax and output with a basic example.

SAS has accelerated the pace of SAS/STAT releases in order to meet customer requirements for versatile statistical methods that are driven by data needs and by advances in methodology. SAS/STAT 14.1, the current production release, is the fifth release of SAS/STAT software during the past four years. As indicated in Table 1, these releases have their own numbering scheme, because they occur more frequently than new versions of Base SAS®.

Table 1 Recent Releases of SAS/STAT Software

Release	Year	Overview Paper	Base SAS Version
SAS/STAT 12.1	2012	Stokes et al. (2012)	SAS 9.3
SAS/STAT 12.3	2013	Stokes (2013)	SAS 9.4
SAS/STAT 13.1	2013	Rodriguez (2014)	SAS 9.4M1
SAS/STAT 13.2	2014	Stokes and Statistical R&D Staff (2015)	SAS 9.4M2
SAS/STAT 14.1	2015	Stokes and Statistical R&D Staff (2015)	SAS 9.4M3

Building Sparse Regression Models with the GLMSELECT Procedure

The GLMSELECT procedure selects effects in general linear models of the form

$$y_i = \beta_0 + \beta_1 x_{i1} + \cdots + \beta_p x_{ip} + \epsilon_i, \quad i = 1, \ldots, n$$

where the response y_i is continuous and the predictors $x_{i1}, \ldots, x_{ip}$ represent main effects that consist of continuous or classification variables, and interaction effects or constructed effects of these variables. With too many predictors, the model can overfit the training data, leading to poor prediction with future data. To deal with this problem, the GLMSELECT procedure supports the model selection methods summarized in Table 2.

Table 2 Effect Selection Methods in the GLMSELECT Procedure

Method	Description
Forward selection	Starts with no effects and adds effects
Backward elimination	Starts with all effects and deletes effects
Stepwise selection	Starts with no effects; effects are added and can be deleted
Least angle regression	Starts with no effects and adds effects; at each step, estimated βs are shrunk toward 0
Lasso	Constrains sum of absolute βs; some βs set to 0
Elastic net	Constrains sums of absolute and squared βs; some βs set to 0
Adaptive lasso	Constrains sum of absolute weighted βs; some βs set to 0
Group lasso	Constrains sum of Euclidean norms of βs corresponding to effects; all βs for the same effect are set to 0 or are non-zero

Forward selection, backward elimination, and stepwise regression reduce the number of effects in the model. In contrast, the lasso, elastic net, adaptive lasso, and group lasso methods are based on regularization. These methods leave all the effects in the model, but they restrict their parameters by setting some to zero while shrinking others toward zero.

Whereas the classical regression estimator solves the least squares problem

$$\min_{\beta_0,\ldots,\beta_p} \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j \right)^2$$

the lasso estimator solves the least squares problem by placing an ℓ_1 penalty on the parameters:

$$\min_{\beta_0,\ldots,\beta_p} \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j \right)^2$$

$$\text{subject to } \sum_{j=1}^{p} |\beta_j| \leq t$$

Provided that the lasso parameter t is small enough, some of the regression coefficients will be exactly zero. Increasing t in discrete steps leads to a sequence of regression coefficients, where the nonzero coefficients at each step correspond to selected parameters. Thus the lasso method produces sparser and potentially more interpretable models than traditional methods such as forward selection. The following example illustrates this distinction.

Example: Predicting the Close Rate for Retail Stores

The close rate for a retail store is the percentage of shoppers who enter the store and make a purchase. Understanding what factors predict close rate is critical to the profitability and growth of large retail companies, and a regression model is constructed to study this question.

The close rates for 500 stores are saved in a data set named **Stores**. Each observation provides information about a store. The variables available for the model are the response **Close_Rate** and the following candidate predictors:

- **X1**, . . . , **X20**, which measure 20 general characteristics of stores, such as floor size and number of employees
- **P1**, . . . , **P6**, which measure six promotional activities, such as advertising and sales
- **L1**, . . . , **L6**, which measure special layouts of items in six departments

In practice, close rate data can involve hundreds of candidate predictors. A small set is used here for illustrative purposes.

Results with the Forward Selection Method

The following statements use the GLMSELECT procedure to build a model with the forward selection method:

```
proc glmselect plots=coefficient data=Stores;
   model Close_Rate = X1-X20 L1-L6 P1-P6 / selection=forward(choose=aic);
run;
```

The SELECTION= option requests the forward method, and the CHOOSE= suboption specifies that the selected model minimize Akaike's information criterion (AIC). The settings for the selection process are listed in Figure 1.

Figure 1 Model Information

The GLMSELECT Procedure

Data Set	WORK.STORES
Dependent Variable	Close_Rate
Selection Method	Forward
Select Criterion	SBC
Stop Criterion	SBC
Choose Criterion	AIC
Effect Hierarchy Enforced	None

At each step of the forward selection process, AIC is evaluated, and the model that yields the minimal value of AIC is chosen. By default, the GLMSELECT procedure uses the Schwarz Bayesian information criterion (SBC) as the select criterion for determining the order in which effects enter at each step. The effect that is selected is the effect whose addition maximizes the decrease in SBC. By default, the procedure also uses SBC as the stop criterion. Selection stops at the step where the next step yields a model with a larger value of SBC. Both AIC and SBC guard against overfitting by penalizing the model for having a large number of parameters.

As shown in Figure 2, the minimum value of AIC is reached at Step 9, when **P1** enters the model.

Figure 2 Selection Summary with Forward Selection

The GLMSELECT Procedure

Forward Selection Summary

Step	Effect Entered	Number Effects In	AIC	SBC
0	**Intercept**	1	545.6009	47.8155
1	**X2**	2	466.3833	-27.1875
2	**X4**	3	436.8566	-52.4996
3	**P3**	4	424.5035	-60.6381
4	**P4**	5	413.4923	-67.4347
5	**L1**	6	402.9892	-73.7232
6	**L3**	7	393.1296	-79.3681
7	**P5**	8	385.0985	-83.1847
8	**L2**	9	377.8229	-86.2457
9	**P1**	10	371.2472*	-88.6068*

*** Optimal Value of Criterion**

The coefficient progression plot in Figure 3, requested using the PLOTS= option, visualizes the selection process.

Figure 3 Coefficient Progression with Forward Selection

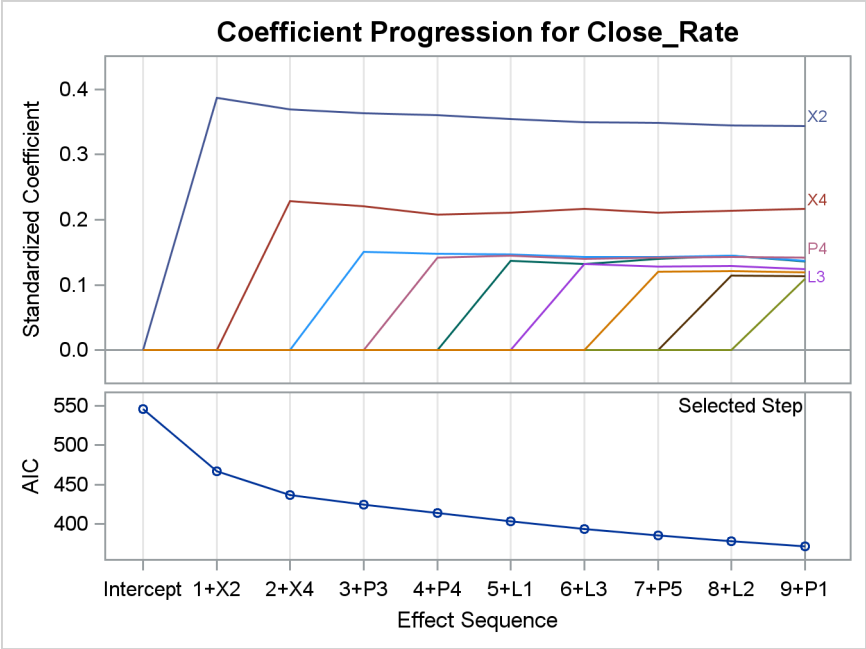


Figure 4 shows the parameter estimates for the final model. The estimates for **X2** and **X4** are larger than the estimates for the seven other predictors, and all the standard errors are comparable in size.

Figure 4 Parameter Estimates with Forward Selection

Parameter Estimates

Parameter	DF	Estimate	Standard Error	t Value
Intercept	1	60.412202	0.119136	507.09
X2	1	1.225952	0.133595	9.18
X4	1	0.798252	0.138799	5.75
L1	1	0.496037	0.137290	3.61
L2	1	0.379632	0.125270	3.03
L3	1	0.438092	0.131785	3.32
P1	1	0.400154	0.137440	2.91
P3	1	0.479429	0.131241	3.65
P4	1	0.520183	0.136973	3.80
P5	1	0.420284	0.132103	3.18

Results with the Lasso Method

The following statements build a model with the lasso method:

```
proc glmselect plots=coefficient data=Stores;
   model Close_Rate = X1-X20 L1-L6 P1-P6 / selection=lasso(choose=aic);
run;
```

The settings for the selection process are listed in Figure 5. As with the settings for the forward method in Figure 1, the choose criterion is AIC and the stop criterion is SBC. However, for the lasso method the GLMSELECT procedure uses the least angle regression algorithm, introduced by Efron et al. (2004), to produce a sequence of regression models in which one parameter is added at each step.

Figure 5 Model Information

The GLMSELECT Procedure

Data Set	WORK.STORES
Dependent Variable	Close_Rate
Selection Method	LASSO
Stop Criterion	SBC
Choose Criterion	AIC
Effect Hierarchy Enforced	None

In contrast to the forward method, which selects a model with nine variables, the lasso method selects a sparse model with two variables, **X2** and **X4**, as shown in Figure 6 and Figure 7.

Figure 6 Selection Summary with Lasso

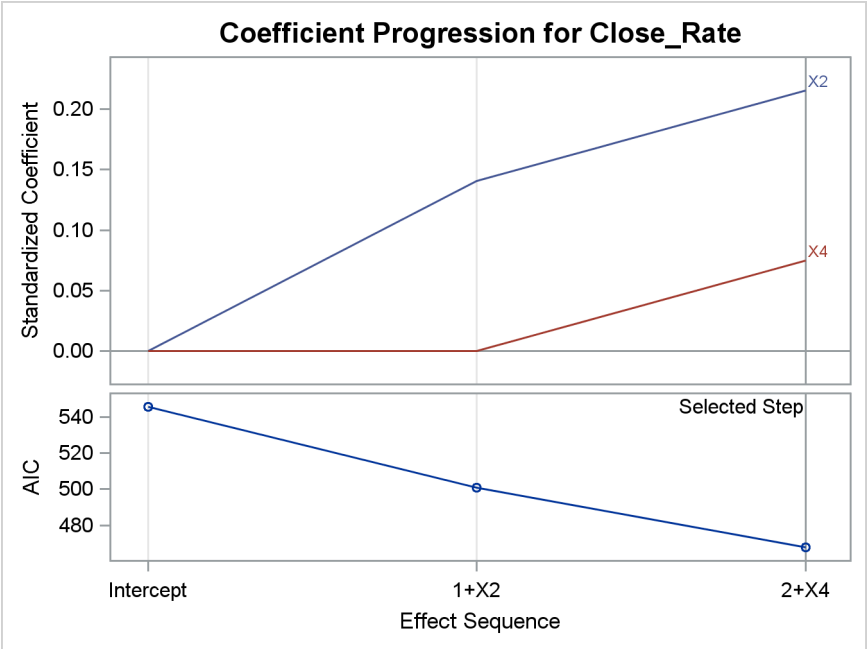
The GLMSELECT Procedure

LASSO Selection Summary

Step	Effect Entered	Effect Removed	Number Effects In	AIC	SBC
0	**Intercept**		1	545.6009	47.8155
1	**X2**		2	500.9692	7.3984
2	**X4**		3	467.7680*	-21.5882*

*** Optimal Value of Criterion**

Figure 7 Coefficient Progression with Lasso



The parameter estimates for the sparse model are shown in Figure 8. Note that these estimates are closer to zero than the corresponding estimates in Figure 4.

Figure 8 Parameter Estimates with Lasso

Parameter Estimates		
Parameter	**DF**	**Estimate**
Intercept	1	61.089916
X2	1	0.767684
X4	1	0.276289

The Elastic Net Method

The elastic net method is a generalization of the lasso method that estimates regression coefficients by solving the doubly penalized least squares problem:

$$\min_{\beta_0,\ldots,\beta_p} \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j \right)^2$$

$$\text{subject to } \sum_{j=1}^{p} |\beta_j| \le t_1 \text{ and } \sum_{j=1}^{p} \beta_j^2 \le t_2$$

In other words, the elastic net method balances between the ℓ_1 lasso penalty and the ℓ_2 penalty for ridge regression. If t_1 is a large value, the elastic net method reduces to ridge regression. If t_2 is a large value, the elastic net method reduces to the lasso method.

The elastic net method offers advantages over the lasso method in three situations (Zou and Hastie 2005; Hastie, Tibshirani, and Wainwright 2015):

- The elastic net method can select more than n variables when the number of parameters p exceeds n. The lasso method can select at most n variables.
- The elastic net method can achieve better prediction when the predictors are highly correlated and $n > p$.
- The elastic net method can handle groups of highly correlated variables more effectively. For an illustration, see Hastie, Tibshirani, and Wainwright (2015, chap. 4).

The following statements use the elastic net method to build a model for **Close_Rate**:

```
proc glmselect plots=coefficient data=Stores;
   model Close_Rate = X1-X20 L1-L6 P1-P6 / selection=elasticnet(choose=aic);
run;
```

In this example, the predictors are not highly correlated, and the selected model (not shown) is identical to the model that is selected with the lasso method.

Other Recent Enhancements

To address the computational demands of model selection when you have a very large number of effects, the GLMSELECT procedure has added screening approaches that you can combine with variable selection methods to reduce the number of regressors to a smaller subset on which the selection is performed.

The procedure provides the SASVI safe screening method proposed by Liu et al. (2014), for which the resulting solution is the same as the solution when no screening is performed. The procedure also provides sure independence screening, proposed by Fan and Lv (2008), a heuristic method that is faster but is not guaranteed to reproduce the true lasso or elastic net solution.

The GLMSELECT procedure has also added the group lasso selection method (Yuan and Lin 2006), which requires groups of parameters to enter the model together. This method is especially useful when the model includes classification effects or spline effects.

For more information, see the chapter on the GLMSELECT procedure in the *SAS/STAT 14.1 User's Guide*.

Building Generalized Linear Models with the HPGENSELECT Procedure

The HPGENSELECT procedure provides model fitting and model building for generalized linear models. It fits models with standard response distributions in the exponential family, such as the normal, Poisson, and Tweedie distributions. In addition, PROC HPGENSELECT fits multinomial models for ordinal and unordered multinomial responses, and it fits zero-inflated Poisson and negative binomial models for count data. For all these models, the HPGENSELECT procedure provides forward, backward, stepwise, and lasso variable selection. The procedure estimates the parameters of a generalized linear model by using maximum likelihood techniques.

Generalized linear models offer versatility for analyzing many types of responses. A generalized linear model consists of three components:

- A linear predictor, which is defined in the same way as for general linear models:

  $$\eta_i = \beta_0 + \beta_1 x_{i1} + \cdots + \beta_p x_{ip}, \quad i = 1, \ldots, n$$

- A specified link function g, which describes how μ_i, the expected value of y_i, is related to η_i:

  $$g(\mu_i) = \eta_i = \beta_0 + \beta_1 x_{i1} + \cdots + \beta_p x_{ip}$$

- An assumed distribution for the responses y_i. For distributions in the exponential family, the variance of the response depends on the mean μ through a variance function V,

  $$\mathrm{Var}(y_i) = \frac{\phi V(\mu_i)}{w_i}$$

  where ϕ is a constant and w_i is a known weight for each observation. The dispersion parameter ϕ is either estimated or known (for example, $\phi = 1$ for the binomial distribution).

Table 3 summarizes these three components.

Table 3 Components of Generalized Linear Models

Component	Description
Linear predictor	Effects involving continuous or classification variables
Link function	Log, logit, inverse, and so on
Distribution	Normal, binomial, Poisson, gamma, Tweedie, and so on

What Is the Difference between the HPGENSELECT and GENMOD Procedures?

Both PROC HPGENSELECT and PROC GENMOD fit generalized linear models. However, there are important design differences in the statistical capabilities of these procedures, as summarized in Table 4.

Table 4 Comparison of PROC HPGENSELECT and PROC GENMOD

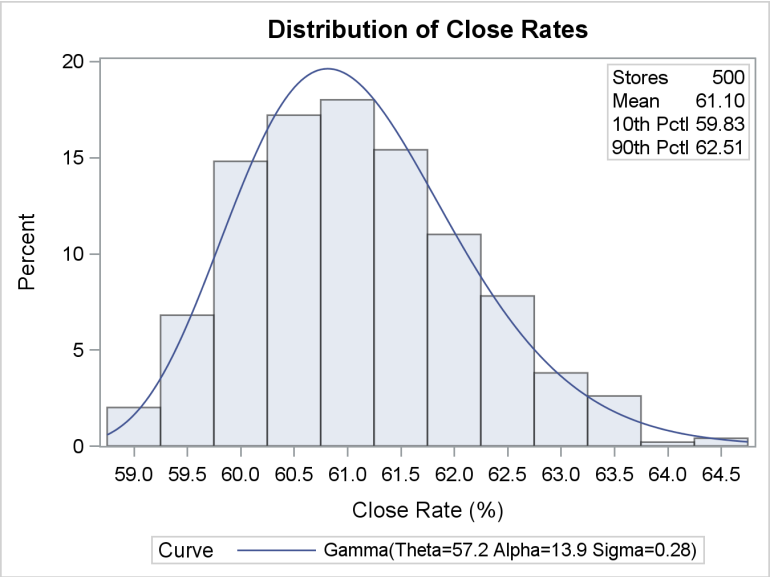
HPGENSELECT Procedure	GENMOD Procedure
Fits and builds generalized linear models	Fits generalized linear models
Analyzes large to massive data	Analyzes moderate to large data
Designed for predictive modeling	Designed for inferential analysis
Runs in single-machine or distributed mode	Runs in single-machine mode

PROC HPGENSELECT is referred to as a high-performance procedure, because it runs in either single-machine mode or distributed mode. For a discussion of these modes, see Cohen and Rodriguez (2013) and Johnston and Rodriguez (2015).

Example: Predicting the Close Rate for Retail Stores (continued)

Figure 9 shows the marginal distribution of the close rates in **Stores**. A gamma distribution provides a good fit, suggesting that a gamma regression model for the conditional mean of close rate is worth exploring.

Figure 9 Distribution of Close Rates for 500 Stores



The following statements use the HPGENSELECT procedure to build a gamma regression model for **Close_Rate**. A preliminary shift transformation is applied to **Close_Rate** because the gamma distribution has a threshold at zero.

```
data Stores; set Stores;
   Close_Rate_0 = Close_Rate - 58;
run;

proc hpgenselect data=Stores;
   model Close_Rate_0 = X1-X20 L1-L6 P1-P6 / distribution = gamma;
   selection method=forward(choose=aic);
run;
```

The METHOD= option requests the forward selection method, and the CHOOSE= suboption specifies that the selected model minimize Akaike's information criterion.

Results with the Forward Selection Method

The settings for the selection process are listed in Figure 10.

Figure 10 Selection Information with Forward Method

The HPGENSELECT Procedure

Selection Information	
Selection Method	Forward
Select Criterion	Significance Level
Stop Criterion	Significance Level
Choose Criterion	AIC
Effect Hierarchy Enforced	None
Entry Significance Level (SLE)	0.05
Stop Horizon	1

Figure 11 shows that the minimum value of AIC is reached at Step 10, when **L5** enters the model. Note that the selected variables are the same as those selected by the GLMSELECT procedure with the forward method (see Figure 2), with the addition of **L5**.

Figure 11 Selection Summary with Forward Method

The HPGENSELECT Procedure

Selection Summary

Step	Effect Entered	Number Effects In	AIC	p Value
0	Intercept	1	1448.2155	.
1	X2	2	1372.9559	<.0001
2	X4	3	1345.6873	<.0001
3	P3	4	1333.3930	0.0002
4	L3	5	1322.5714	0.0004
5	P4	6	1312.2416	0.0005
6	L1	7	1304.9794	0.0025
7	P5	8	1297.9234	0.0027
8	L2	9	1291.8963	0.0048
9	P1	10	1286.2800	0.0061
10	L5	11	1282.0650*	0.0129

*** Optimal Value of Criterion**

Figure 12 shows the fit statistics for the selected model.

Figure 12 Fit Statistics for Gamma Regression Model Selected with Forward Method

Fit Statistics	
-2 Log Likelihood	1258.06
AIC (smaller is better)	1282.06
AICC (smaller is better)	1282.71
BIC (smaller is better)	1332.64
Pearson Chi-Square	41.4567
Pearson Chi-Square/DF	0.08478

Figure 13 shows the parameter estimates for the selected model. As in Figure 4, the estimates for **X2** and **X4** are larger in magnitude than the estimates for the other predictors.

Figure 13 Parameter Estimates for Gamma Regression Model Selected with Forward Method

Parameter Estimates

Parameter	DF	Estimate	Standard Error	Chi-Square	Pr > ChiSq
Intercept	1	0.421938	0.015141	776.6306	<.0001
X2	1	-0.129234	0.014444	80.0555	<.0001
X4	1	-0.083540	0.014834	31.7168	<.0001
L1	1	-0.048919	0.014309	11.6878	0.0006
L2	1	-0.035614	0.013278	7.1939	0.0073
L3	1	-0.049864	0.013921	12.8299	0.0003
L5	1	-0.034887	0.013950	6.2544	0.0124
P1	1	-0.040273	0.014554	7.6575	0.0057
P3	1	-0.049916	0.013947	12.8092	0.0003
P4	1	-0.051448	0.014473	12.6367	0.0004
P5	1	-0.039721	0.013947	8.1112	0.0044
Dispersion	1	12.053493	0.752016	.	.

Results with the Lasso Method

The following statements build a gamma regression model with the lasso method:

```
proc hpgenselect data=Stores;
   model Close_Rate_0 = X1-X20 L1-L6 P1-P6 / distribution = gamma;
   selection method=lasso(choose=aic);
run;
```

The lasso again selects a sparse model with two variables, **X2** and **X4**. The regularization parameter that minimizes AIC is shown in Figure 14.

Figure 14 Lasso Regularization Parameter

The HPGENSELECT Procedure

Maximum Regularization Parameter	0.118143
Chosen Regularization Parameter	0.060489

The lasso estimates for **X2** and **X4** in Figure 15 are shrunk toward zero, compared with the estimates in Figure 13.

Figure 15 Parameter Estimates for Gamma Regression Model Selected with Lasso Method

Parameter Estimates

Parameter	DF	Estimate
Intercept	1	0.324793
X2	1	-0.069242
X4	1	-0.014639
Dispersion	0	1.000000

Building Quantile Regression Models with the QUANTSELECT Procedure

The QUANTSELECT procedure performs effect selection in the framework of quantile regression, which models the quantiles (percentiles) of a response variable conditional on covariates. Quantile regression models, introduced by Koenker and Bassett (1978), can potentially describe the entire conditional distribution of the response. By comparison, general linear models and generalized linear models describe only the conditional mean of the response but are computationally less expensive.

Quantile regression does not assume a particular distribution for the response, nor does it assume a constant variance for the response, unlike ordinary least squares regression. Figure 16 illustrates data in which the variance of the response **Y** increases with the covariate **X**. Simple linear regression models the conditional mean $E[Y|X]$, but it does not capture the conditional variance $\mathrm{Var}[Y|X]$.

Figure 16 Variance in Y Increases with X

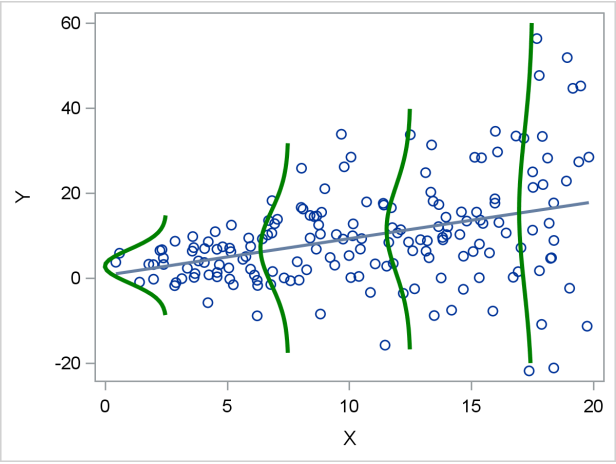
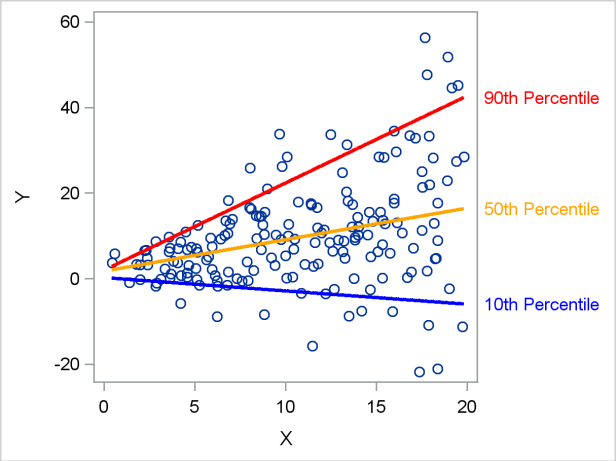


Figure 17 shows quantile regression lines for the 10th, 50th, and 90th conditional percentiles of **Y**. These are formally referred to as the quantile regression lines that correspond to the quantile levels 0.10, 0.50, and 0.90.

Figure 17 Regression Models for Three Percentiles



Fitting a Quantile Regression Model

The regression model for quantile level τ is

$$Q_\tau(Y|X) = X\beta(\tau), \quad 0 < \tau < 1$$

where $\beta(\tau)$ is estimated by solving the minimization problem

$$\min_{\beta_0,\ldots,\beta_p} \sum_{i=1}^{n} \rho_\tau \left(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j \right)$$

and $\rho_\tau(r) = \tau \max(r, 0) + (1 - \tau) \max(-r, 0)$. The function $\rho_\tau(r)$ is referred to as the check loss, because its shape resembles a check mark.

For each quantile level τ, the solution to the minimization problem yields a distinct set of regression coefficients. Note that $\tau = 0.5$ corresponds to median regression, and $2\rho_{0.5}(r)$ is the absolute value function.

Using the QUANTSELECT Procedure

The QUANTSELECT procedure fits and builds quantile regression models. It is designed primarily as an effect selection procedure and does not include regression diagnostics and hypothesis testing, which are provided by the QUANTREG procedure.

The QUANTSELECT procedure supports the model selection methods summarized in Table 5.

Table 5 Effect Selection Methods in the QUANTSELECT Procedure

Method	Description
Forward selection	Starts with no effects and adds effects
Backward elimination	Starts with all effects and deletes effects
Stepwise selection	Starts with no effects; effects are added and can be deleted
Lasso	Adds and deletes effects based on a constrained version of estimated check risk where the ℓ_1 norm of the βs is penalized
Adaptive lasso	Constrains sum of absolute weighted βs; some βs set to 0

Example: Predicting the Close Rate for Retail Stores (continued)

The examples in the preceding sections show how you can build a standard regression model and a gamma regression model for the close rate data. These models answer the following questions:

How can I predict the close rate for a new store?

Which variables explain the average close rate of a store?

By building a quantile regression model, you can answer a different question:

Are there variables that differentiate low and high close rates?

The following statements use the QUANTSELECT procedure to build quantile regression models for levels 0.1, 0.5, and 0.9:

```
proc quantselect data=Stores plots=Coefficients seed=15531;
   model Close_Rate = X1-X20 L1-L6 P1-P6 / quantile = 0.1 0.5 0.9
                                           selection=lasso(sh=3);
   partition fraction(validate=0.3);
run;
```

The SELECTION= option specifies the lasso method with a stop horizon of 3. The PARTITION statement reserves 30% of the data for validation, leaving the remaining 70% for training.

Figure 18 summarizes the effect selection process for quantile level 0.1. The lasso method generates a sequence of candidate models, and the process chooses the model that minimizes the average validation check loss (ACL). The process stops at Step 14.

Figure 18 Selection Summary for Quantile Level 0.1

The QUANTSELECT Procedure
Quantile Level = 0.1

Selection Summary

Step	Effect Entered	Effect Removed	Number Effects In	Validation ACL
0	Intercept		1	0.1578
1	X2		2	0.1667
2	X4		3	0.1566
3	P3		4	0.1380
4	P1		5	0.1326
5	P2		6	0.1119
6	P4		7	0.1104
7	X20		8	0.1113
8	X3		9	0.1111
9	P5		10	0.1096
10		P5	9	0.1111
11	P5		10	0.1096
12		X3	9	0.1083*
13	L1		10	0.1105
14	X3		11	0.1117

The coefficient progression plot in Figure 19 visualizes the selection process, and it is similar to the coefficient progression plot that is constructed by the GLMSELECT procedure in Figure 3. In both plots, **X2** and **X4** are the first two variables that enter the model.

Figure 19 Coefficient Progression for Quantile Level 0.1

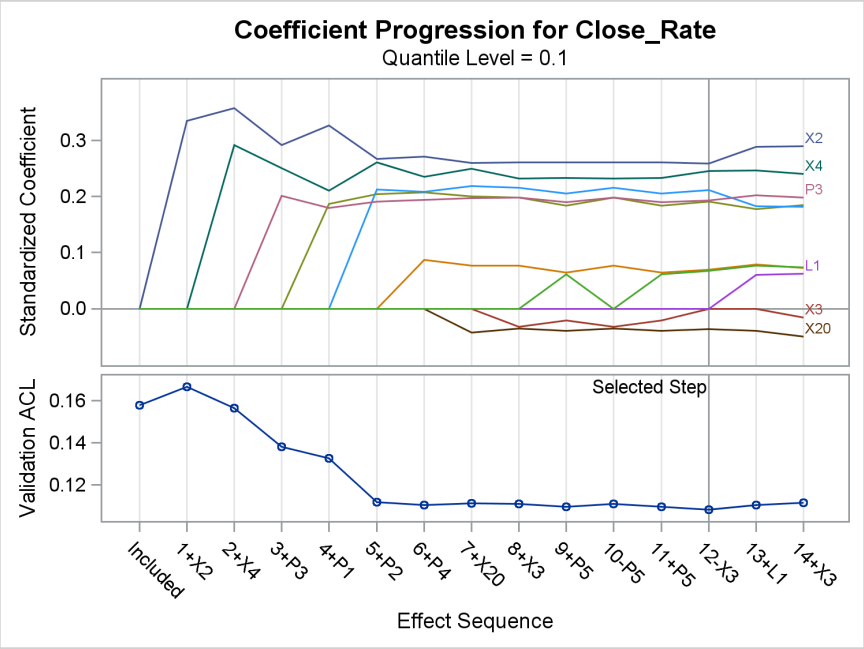


Figure 20 shows the fit statistics for the final model for quantile level 0.1.

Figure 20 Fit Statistics for Model Selected for Quantile Level 0.1

The QUANTSELECT Procedure
Quantile Level = 0.1

Fit Statistics	
Objective Function	36.17929
R1	0.38327
Adj R1	0.36909
AIC	-1616.52369
AICC	-1616.00496
SBC	-1581.62407
ACL (Train)	0.10134
ACL (Validate)	0.10826

Figure 21 shows the parameter estimates for the final model for quantile level 0.1.

Figure 21 Parameter Estimates for Model Selected for Quantile Level 0.1

Parameter Estimates			
Parameter	**DF**	**Estimate**	**Standardized Estimate**
Intercept	1	60.097618	0
X2	1	0.953402	0.258498
X4	1	0.933705	0.245902
X20	1	-0.140895	-0.035981
P1	1	0.724145	0.190798
P2	1	0.783880	0.211752
P3	1	0.696274	0.193163
P4	1	0.260641	0.069442
P5	1	0.242147	0.067135

The QUANTSELECT procedure produces a parallel but distinct set of results for quantile levels 0.5 and 0.9. The parameter estimates for the final models are shown in Figure 22 and Figure 23.

Figure 22 Parameter Estimates for Model Selected for Quantile Level 0.5

Parameter Estimates			
Parameter	DF	Estimate	Standardized Estimate
Intercept	1	60.950579	0
X2	1	1.508595	0.409029
X4	1	0.710687	0.187168
P3	1	0.361047	0.100163
P4	1	0.669943	0.178491
P5	1	0.544278	0.150902

Figure 23 Parameter Estimates for Model Selected for Quantile Level 0.9

Parameter Estimates			
Parameter	DF	Estimate	Standardized Estimate
Intercept	1	61.079231	0
X2	1	0.982776	0.266463
X4	1	1.118507	0.294572
L2	1	1.027725	0.297930
L3	1	0.859988	0.240257
L5	1	0.672210	0.186588
P5	1	0.192967	0.053500

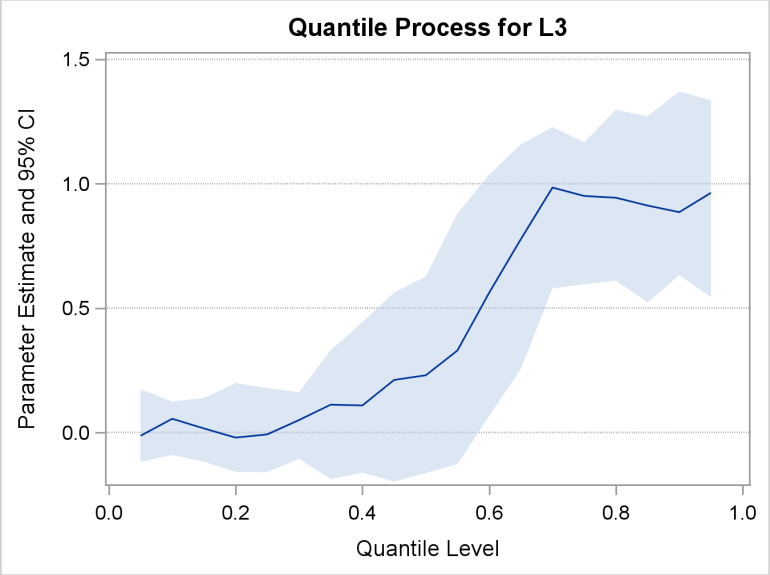
A sparse model with only six variables (**X2**, **X4**, **L2**, **L3**, **L5**, and **P5**) is selected as the best conditional model for predicting the 90th percentile. The layout variables **L2**, **L3**, and **L5** are in this model, but not in the models for the 10th and 50th percentiles. The variables **X2** and **X4** are common to the models for all three percentiles. These results give you insights about store performance that you would not obtain directly from standard regression methods.

You can create quantile process plots that show how the estimated regression coefficients for a covariate change as a function of the quantile level τ in the interval (0,1). The following program creates a process plot for **L3**. First the QUANTSELECT procedure is used to build a quantile process regression model. Then the QUANTREG procedure is used to compute 95% confidence limits for the coefficients.

```
proc quantselect data=Stores plots=Coefficients seed=15531;
   model Close_Rate = X1-X20 L1-L6 P1-P6 / quantile=process(ntau=10)
                                           selection=forward(sh=3);
run;

proc quantreg data=Stores;
   ods output ParameterEstimates=ParmEst;
   model Close_Rate = &_QRSIND / quantile=0.05 to 0.95 by 0.05;
run;

data ParmEstPlot; set ParmEst; if Parameter EQ "L3"; run;

title "Quantile Process for L3";
proc sgplot data=ParmEstPlot noautolegend;
   band upper=UpperCL lower=LowerCL x=Quantile / transparency=0.5;
   series y=Estimate x=Quantile;
   yaxis label='Parameter Estimate and 95% CI'
         grid gridattrs=(thickness=1px color=gray pattern=dot);
   xaxis label='Quantile Level';
run;
```

The process plot, shown in Figure 24, reveals that **L3** affects the upper half of the close rate distribution. Again, this is an insight that you would not obtain with standard regression methods.

Figure 24 Quantile Process Plot for L3



Fitting Generalized Additive Models with the GAMPL Procedure

The GAMPL procedure is a high-performance procedure that fits generalized additive models that are based on low-rank regression splines (Wood 2006).

Generalized additive models are extensions of generalized linear models. In addition to allowing linear predictors, they allow spline terms in order to capture nonlinear dependency that is either unknown or too complex to be characterized with a parametric effect such as a linear or quadratic term.

Each spline term is constructed using the thin-plate regression spline technique (Wood 2003). A roughness penalty is applied to each spline term by a smoothing parameter that controls the balance between goodness of fit and roughness of the spline curve.

Table 6 summarizes the components of a generalized additive model.

Table 6 Components of Generalized Additive Models

Component	Description
Linear predictor	Effects involving continuous or classification variables
Nonparametric predictor	Spline terms involving one or more continuous variables
Link function	Log, logit, inverse, and so on
Distribution	Normal, binomial, Poisson, gamma, and so on

Because a generalized additive model allows both linear and nonparametric predictors, it is useful for problems involving unknown—possibly nonlinear—relationships between the response and the predictors, as well as relationships that can be assumed to be linear. Frigo and Osterloo (2016) describe a problem of this type in the context of insurance pricing and propose solutions that use the GAMPL procedure and the HPGENSELECT procedure.

Strictly speaking, the GAMPL procedure does model fitting rather than model building. Unlike the GLMSELECT, HPGENSELECT, and QUANTSELECT procedures, the GAMPL procedure does not select variables. However, in some situations the results of spline fits that you obtain using PROC GAMPL suggest parametric effects in a model that you can then fit with the HPGENSELECT procedure, as illustrated in the following example.

Example: Predicting Claim Rates for Loans

This example is drawn from the mortgage insurance industry, where analysts create models to predict conditional claim rates for specific types of loans. Understanding how claim rates depend on predictors is critical, because the model is used to assess risk and allocate funds for potential claims.

Claim rates for 10,000 mortgages are saved in a data set named **Claims**. The response variable **Rate** is the number of claims per 10,000 contracts in a policy year, and it is assumed to follow a Poisson distribution whose mean depends on the predictors listed in Table 7.

Table 7 Predictors for Claim Rate

Predictor	Description	Contribution
Age	Age of loan	Unknown, possibly quadratic
Price	Price of house	Unknown, nonlinear
RefInd	Indicator if loan is refinanced	Linear
PayIncmRatio	Payment-to-income ratio	Linear
RefInctvRatio	Refinance incentive ratio	Linear
UnempRate	Unemployment rate	Linear

In practice, models of this type involve many more predictors. A subset is used here for illustrative purposes.

The following statements use the GAMPL procedure to fit a generalized additive model for **Rate**:

```
proc gampl data=Claims plots=components;
   class RefInd;
   model Rate = param(RefInd PayIncmRatio RefInctvRatio UnempRate)
                spline(Age) spline(Price) / dist=poisson;
run;
```

The PARAM( ) option specifies parametric linear terms for **RefInd**, **PayIncmRatio**, **RefInctvRatio**, and **UnempRate**. The SPLINE options specify spline effects for **Age** and **Price**.

Figure 25 displays information about the model fitting process. The Poisson mean of **Rate** is modeled by a log link function. The performance iteration algorithm (Gu and Wahba 1991) is used to obtain optimal smoothing parameters for the spline effects. The unbiased risk estimator (UBRE) criterion is used for model evaluation during the process of selecting smoothing parameters for the spline effects.

Figure 25 Model Information

The GAMPL Procedure

Model Information	
Data Source	WORK.CLAIMS
Response Variable	Rate
Class Parameterization	GLM
Distribution	Poisson
Link Function	Log
Fitting Method	Performance Iteration
Fitting Criterion	UBRE
Optimization Technique for Smoothing	Newton-Raphson
Random Number Seed	1990293722

Figure 26 shows the fit statistics. You can use effective degrees of freedom to compare generalized additive models with generalized linear models, which do not involve spline terms. You can also use the information criteria, AIC, AICC, and BIC, for model comparisons, and you can use the GCV criterion for comparisons with other generalized additive models or penalized models.

Figure 26 Fit Statistics with GAMPL Procedure

Fit Statistics	
Penalized Log Likelihood	-26776
Roughness Penalty	7.83354
Effective Degrees of Freedom	16.54759
Effective Degrees of Freedom for Error	9982.63719
AIC (smaller is better)	53578
AICC (smaller is better)	53578
BIC (smaller is better)	53697
UBRE (smaller is better)	-0.00355

Figure 27 and Figure 28 show estimates for the components of the model.

Figure 27 Estimates for Parametric Terms

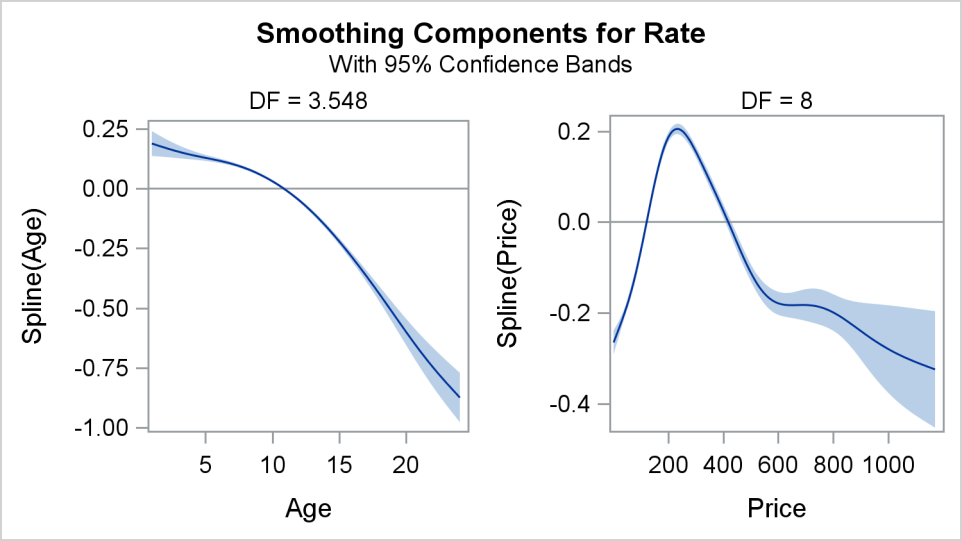
Parameter Estimates					
Parameter	**DF**	**Estimate**	**Standard Error**	**Chi-Square**	**Pr > ChiSq**
Intercept	1	2.484711	0.020877	14164.8501	<.0001
RefInd 0	1	-0.008901	0.005571	2.5532	0.1101
RefInd 1	0	0	.	.	.
PayIncmRatio	1	0.035740	0.009740	13.4642	0.0002
RefInctvRatio	1	-0.031276	0.009627	10.5555	0.0012
UnempRate	1	0.008048	0.002764	8.4778	0.0036

Figure 28 Estimates for Smoothing Components

Estimates for Smoothing Components						
Component	**Effective DF**	**Smoothing Parameter**	**Roughness Penalty**	**Number of Parameters**	**Rank of Penalty Matrix**	**Number of Knots**
Spline(Age)	3.54759	35754.3	7.8335	9	10	24
Spline(Price)	8.00000	1.0000	1.045E-6	9	10	2000

Figure 29 displays plots of the fitted splines for **Age** and **Price**.

Figure 29 Spline Components for Age and Price



The plots suggest quadratic polynomials to characterize the nonlinearity in **Age** and **Price**. The following statements incorporate these polynomials in a generalized linear model that is fitted with the HPGENSELECT procedure (you could also use the GENMOD procedure):

```
proc hpgenselect data=Claims;
   class RefInd;
   model Rate = RefInd PayIncmRatio RefInctvRatio UnempRate
                Age Age*Age Price Price*Price / dist=poisson;
run;
```

Fit statistics for the model that is fitted with PROC HPGENSELECT are given in Figure 30.

Figure 30 Fit Statistics with HPGENSELECT Procedure

The HPGENSELECT Procedure

Fit Statistics	
-2 Log Likelihood	54754
AIC (smaller is better)	54772
AICC (smaller is better)	54772
BIC (smaller is better)	54837
Pearson Chi-Square	11284
Pearson Chi-Square/DF	1.1294

The AIC, AICC, and BIC statistics in Figure 26 are smaller even though the generalized additive model involves more parameters for the splines.

Building Classification and Regression Tree Models with the HPSPLIT Procedure

The HPSPLIT procedure is a high-performance procedure that builds tree-based statistical models for classification and regression. The procedure produces classification trees, which model a categorical response, and regression trees, which model a continuous response. Both types of trees are referred to as decision trees, because the model is expressed as a series of if-then statements.

The predictor variables for tree models can be categorical or continuous. The model is based on a partition of the predictor space into nonoverlapping segments, which correspond to the leaves (terminal nodes) of the tree. Partitioning is done recursively, starting with the root node, which contains all the data. At each step, the parent node is split into child nodes through selection of a predictor variable and a split value that minimize the variability in the response across the child nodes.

Tree models are built from training data for which the response values are known, and these models are subsequently used to score (classify or predict) response values for new data. For classification trees, the most frequent response level of the training observations in a leaf is used to classify observations in that leaf. For regression trees, the average response of the training observations in a leaf is used to predict the response for observations in that leaf. The splitting rules that define the leaves provide the information that is needed to score new data.

The process of building a decision tree begins with growing a large, full tree. Various measures, such as the Gini index, entropy, and residual sum of squares, are used to assess candidate splits for each node. To prevent overfitting, the full tree is pruned back to a smaller tree that balances the goals of fitting training data and predicting new data. Two approaches for finding the best tree are cost-complexity pruning (Breiman et al. 1984) and C4.5 pruning (Quinlan 1993).

Example: Predicting Claim Rates for Loans (continued)

The following statements use the HPSPLIT procedure to build a regression tree for **Rate**:

```
proc hpsplit data=Claims seed=15531
     plots=(wholetree zoomedtree(nodes=('0' '3') depth=2));
   class RefInd;
   model Rate = RefInd PayIncmRatio RefInctvRatio UnempRate Age Price;
```

```
    grow variance;
    prune costcomplexity;
    partition fraction(validate=0.3);
run;
```

With 10,000 observations, it is reasonable to use a PARTITION statement to reserve 30% of the data for validation, leaving the remaining 70% for training. The GROW statement specifies the variance (residual sum of squares) criterion for determining variable splits. The PRUNE statement requests the cost-complexity method of pruning. The procedure uses the validation set to determine the size of the optimal tree. If a validation set is not specified, the procedure uses k-fold cross validation for this purpose.

Figure 31 provides information about the methods that are used to grow and prune the tree.

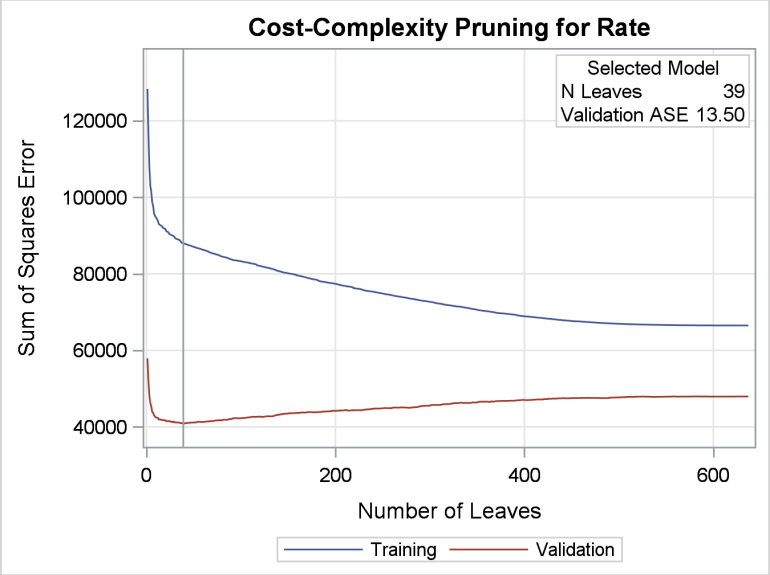
Figure 31 Model Information

The HPSPLIT Procedure

Model Information	
Split Criterion Used	Variance
Pruning Method	Cost-Complexity
Subtree Evaluation Criterion	Cost-Complexity
Number of Branches	2
Maximum Tree Depth Requested	10
Maximum Tree Depth Achieved	10
Tree Depth	10
Number of Leaves Before Pruning	637
Number of Leaves After Pruning	39

The cost-complexity pruning plot in Figure 32 displays the error sum of squares for the training and validation data as a function of the number of leaves. A tree size of 39 leaves minimizes this quantity.

Figure 32 Pruning Plot



The diagram in Figure 33, which is requested using the WHOLETREE option, provides an overview of the final tree, which has 39 leaves. The leaf color represents the predicted value of **Rate**, which is the average observed value of **Rate** for the training observations in that leaf.

Figure 33 Whole Tree Plot

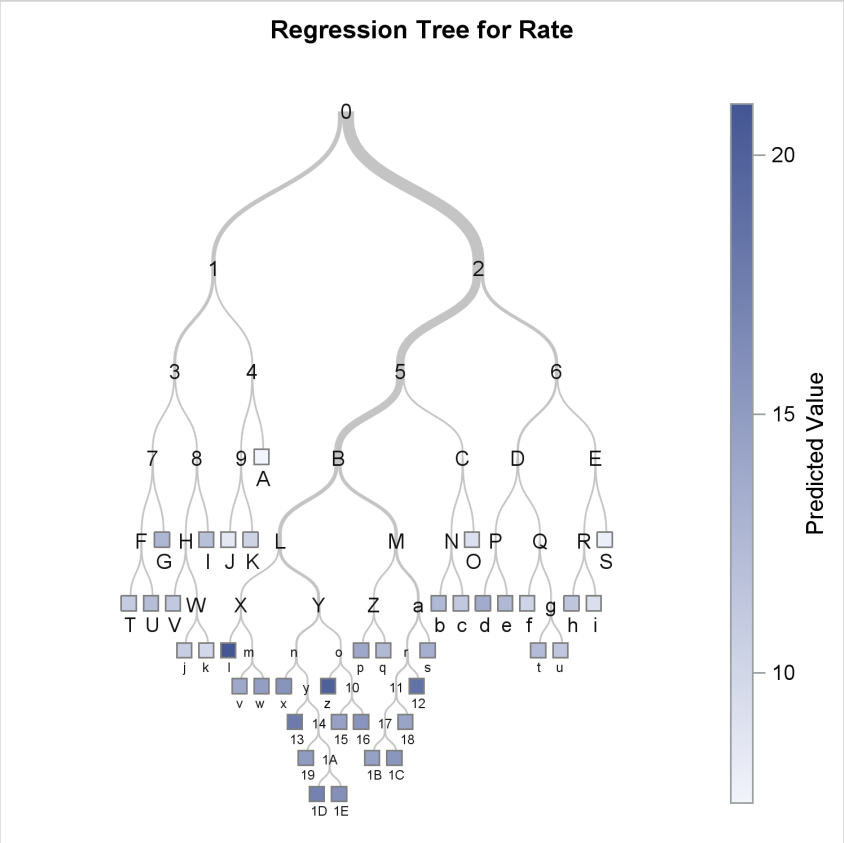
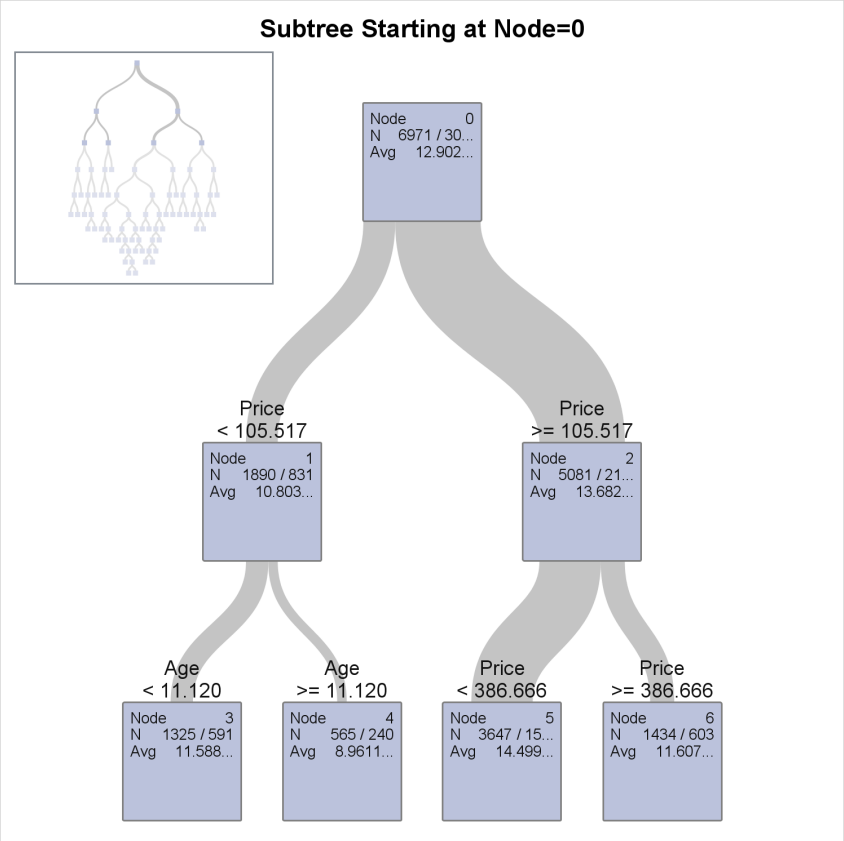


Figure 34 Zoomed Plot Starting at Node 0



The diagram in Figure 34, which is requested using the ZOOMEDTREE option, displays the root node (Node 0) and the next two levels of the final tree. Node 0 contains all of the 6,971 observations in the training data. The first split assigns the 1,890 observations where **Price** < 105.517 to Node 1, and the remaining 5,081 observations where **Price** $\geq$ 105.517 to Node 2. The next split assigns the observations in Node 1 where **Age** < 11.120 to Node 3. A second diagram, which is requested using the ZOOMEDTREE option and is not shown, displays Node 3 and the two levels that follow Node 3.

Figure 35 shows fit statistics for the final tree.

Figure 35 Fit Statistics

The HPSPLIT Procedure

Fit Statistics for Selected Tree

	N Leaves	ASE	RSS
Training	39	12.6203	87975.8
Validation	39	13.4979	40885.3

Figure 36 shows measures of variable importance. The variables **Price** and **Age** are the most useful predictors.

Figure 36 Variable Importance

Variable Importance

		Training		Validation			
Variable	**Variable Label**	**Relative**	**Importance**	**Relative**	**Importance**	**Relative Ratio**	**Count**
Price	Wtd Avg of House Price at Loan Origination	1.0000	157.4	1.0000	104.9	1.0000	14
Age	Age of Loan in Years	0.7728	121.6	0.7502	78.7345	0.9709	16
UnempRate	Wtd Avg of Unemployment Rates	0.0719	11.3167	0.1023	10.7354	1.4226	1
PayIncmRatio	Wtd Avg of Payment to Income Ratios	0.1582	24.8905	0.0892	9.3573	0.5638	6
RefInctvRatio	Wtd Avg of Refinance Incentive Ratios	0.0458	7.2074	0.0098	1.0288	0.2141	1

This example illustrates a limitation of regression tree models: they are adequate for fitting response surfaces that are constant over rectangular regions of the predictor space, but they lack the flexibility necessary to capture smooth relationships between the predictors and the response. In these situations, regression models with continuous effects will outperform tree models—and, in fact, for the claim rate prediction problem, the approaches discussed in the previous example provide better solutions. On the other hand, tree models offer the advantages of being easy to explain and handling missing values efficiently through the use of surrogate variables. For a comprehensive discussion of tree-based methods, see Hastie, Tibshirani, and Friedman (2009).

Summary: Benefits of Modern Approaches for Model Building

Table 8 provides a high-level comparison of the five approaches discussed in this paper. All these approaches share a common goal of delivering good predictive ability with future data, but they differ in the benefits that they offer and the assumptions that they require you to make.

All these approaches avoid overfitting the training data by giving you methods of choosing tuning parameters and computing model fit statistics that are based on information criteria and validation techniques. When you have sufficient data for partitioning, you should use validation data for choosing the tuning parameter and test data for assessing predictive ability.

The ability to score future data is an essential aspect of predictive modeling. All the procedures that are illustrated in this paper provide ways to score data with the final model, as summarized in Table 9.

In order to decide which modeling approaches are appropriate for your work, you should understand their underlying assumptions, characteristics, and relative benefits. These aspects are explained in the “Details” sections of the procedure chapters in the *SAS/STAT 14.1 User’s Guide*.

Table 8 New Tools for Regression Modeling in Recent Releases of SAS/STAT Software

Approach	Benefits	Model Type	Availability
Lasso methods for selecting regression effects	Sparse models for high-dimensional data; potentially more interpretable	Parametric	GLMSELECT, HPGENSELECT, QUANTSELECT
Effect selection for generalized linear models	Wide variety of response distributions	Parametric	HPGENSELECT, QUANTSELECT
Effect selection for quantile regression	Ability to model the entire conditional response distribution	Parametric	QUANTSELECT
Generalized additive models with penalization	Flexibility for capturing complex dependency relationships	Semiparametric	GAMPL
Classification and regression trees	Interpretability of small trees, handling of missing values	Nonparametric	HPSPLIT

Table 9 Functionality for Scoring

Procedure	Feature	Description
GLMSELECT	SCORE statement	Creates SAS data set that contains predicted values for new data
	CODE statement	Writes SAS DATA step code for computing predicted values
HPGENSELECT	CODE statement	Writes SAS DATA step code for computing predicted values
QUANTSELECT	CODE statement	Writes SAS DATA step code for computing predicted values
GAMPL	OUTPUT statement	Computes predicted values for observations with missing responses
HPSPLIT	CODE statement	Writes SAS DATA step code for computing predicted values

Keeping Up with New Releases of SAS/STAT

The model building approaches that are described in this paper are five of the many enhancements in recent releases of SAS/STAT software. The best place to find out about these enhancements is the chapter "What's New in SAS/STAT" in the online documentation at http://support.sas.com/documentation/onlinedoc/stat/. Also, be sure to visit the Statistics and Operations Research focus area at http://support.sas.com/statistics. There you can watch helpful videos, download overview papers, and subscribe to a quarterly e-newsletter.

REFERENCES

Breiman, L., Friedman, J., Olshen, R. A., and Stone, C. J. (1984). *Classification and Regression Trees.* Belmont, CA: Wadsworth.

Cohen, R., and Rodriguez, R. N. (2013). "High-Performance Statistical Modeling." In *Proceedings of the SAS Global Forum 2013 Conference*. Cary, NC: SAS Institute Inc. http://support.sas.com/resources/papers/proceedings13/401-2013.pdf.

Efron, B., Hastie, T. J., Johnstone, I. M., and Tibshirani, R. (2004). "Least Angle Regression." *Annals of Statistics* 32:407–499. With discussion.

Fan, J., and Lv, J. (2008). "Sure Independence Screening for Ultrahigh Dimensional Feature Space." *Journal of the Royal Statistical Society, Series B* 70:849–911.

Frigo, C., and Osterloo, K. (2016). "exSPLINE That: Explaining Geographic Variation in Insurance Pricing." In *Proceedings of the SAS Global Forum 2016 Conference*. Cary, NC: SAS Institute Inc. http://support.sas.com/resources/papers/proceedings16/8441-2016.pdf.

Gu, C., and Wahba, G. (1991). "Minimizing GCV/GML Scores with Multiple Smoothing Parameters via the Newton Method." *SIAM Journal on Scientific Computing* 12:383–398.

Hastie, T., Tibshirani, R., and Wainwright, M. (2015). *Statistical Learning with Sparsity: The Lasso and Generalizations*. Boca Raton, FL: CRC Press.

Hastie, T. J., Tibshirani, R. J., and Friedman, J. H. (2009). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. 2nd ed. New York: Springer-Verlag.

Johnston, G., and Rodriguez, R. N. (2015). "Introducing the HPGENSELECT Procedure: Model Selection for Generalized Linear Models and More." In *Proceedings of the SAS Global Forum 2015 Conference*. Cary, NC: SAS Institute Inc. http://support.sas.com/resources/papers/proceedings15/SAS1742-2015.pdf.

Koenker, R., and Bassett, G. W. (1978). "Regression Quantiles." *Econometrica* 46:33–50.

Liu, J., Zhao, Z., Wang, J., and Ye, J. (2014). "Safe Screening with Variational Inequalities and Its Application to Lasso." In *JMLR Workshop and Conference Proceedings, Vol. 32: Proceedings of the Thirty-First International Conference on Machine Learning, Second Cycle*. http://jmlr.org/proceedings/papers/v32/liuc14.pdf.

Quinlan, J. R. (1993). *C4.5: Programs for Machine Learning*. San Francisco: Morgan Kaufmann.

Rodriguez, R. N. (2014). "SAS/STAT 13.1: Round-Up." In *Proceedings of the SAS Global Forum 2014 Conference*. Cary, NC: SAS Institute Inc. http://support.sas.com/resources/papers/proceedings14/SAS181-2014.pdf.

Stokes, M. (2013). "Current Directions in SAS/STAT Software Development." In *Proceedings of the SAS Global Forum 2013 Conference*. Cary, NC: SAS Institute Inc. http://support.sas.com/resources/papers/proceedings13/432-2013.pdf.

Stokes, M., Chen, F., Yuan, Y., and Cai, W. (2012). "Look Out: After SAS/STAT 9.3 Comes SAS/STAT 12.1!" In *Proceedings of the SAS Global Forum 2012 Conference*. Cary, NC: SAS Institute Inc. http://support.sas.com/resources/papers/proceedings12/313-2012.pdf.

Stokes, M., and Statistical R&D Staff (2015). "SAS/STAT 14.1: Methods for Massive, Missing, or Multifaceted Data." In *Proceedings of the SAS Global Forum 2015 Conference*. Cary, NC: SAS Institute Inc. http://support.sas.com/resources/papers/proceedings15/SAS1940-2015.pdf.

Wood, S. (2003). "Thin Plate Regression Splines." *Journal of the Royal Statistical Society, Series B* 65:95–114.

Wood, S. (2006). *Generalized Additive Models*. Boca Raton, FL: Chapman & Hall/CRC.

Yuan, M., and Lin, L. (2006). "Model Selection and Estimation in Regression with Grouped Variables." *Journal of the Royal Statistical Society, Series B* 68:49–67.

Zou, H., and Hastie, T. (2005). "Regularization and Variable Selection via the Elastic Net." *Journal of the Royal Statistical Society, Series B* 67:301–320.

Acknowledgments

The following SAS/STAT developers contributed to this paper: Weijie Cai, Gordon Johnston, Jun Liu, Warren Kuhfeld, and Yonggang Yao. The author also thanks Ed Huddleston for editorial assistance.

Paper 259-2009

Applications of the GLMSELECT Procedure for Megamodel Selection

Robert A. Cohen, SAS Institute Inc., Cary, NC

ABSTRACT

When you can select regression models from tens of thousands of effects, what possibilities for modeling are open to you? This paper explores applications of the GLMSELECT procedure in SAS/STAT® software to such problems. The GLMSELECT procedure supports a variety of model selection methods for general linear models. Examples of megamodels arising in genomic data analysis and nonparametric modeling are discussed. In addressing these examples, built-in facilities of the procedure to handle validation and test data are highlighted in addition to techniques for extending the procedure's functionality to address model selection bias by using bootstrap-based model averaging.

INTRODUCTION

When you are faced with a predictive modeling problem that has many possible predictor effects—dozens, hundreds, or even thousands—a natural question is "What subset of the effects provides the best model for the data?" Statistical model selection seeks to answer this question, using a variety of definitions of the "best" model in addition to a variety of heuristic procedures for approximating the true but computationally infeasible solution. The GLMSELECT procedure implements statistical model selection in the framework of general linear models. Methods include not only extensions to general linear models of methods long familiar in the REG procedure (forward, backward, and stepwise) but also the newer LASSO and LAR methods of Tibshirani (1996) and Efron et al. (2004), respectively.

Although the model selection question seems reasonable, trying to answer it for real data can lead to problematic pitfalls, including the following:

- Only one model is selected, and even that is not guaranteed to be the "best." Other, more parsimonious or more intuitively reasonable models that might provide nearly as good or even better models might not be found by the particular heuristic method that is used.
- Model selection might be unduly affected by outliers.
- A "selection bias" exists because a parameter is more likely to be selected if it is above its expected value than if it is below its expected value.
- Standard methods of inference for the final model are invalid in the model selection context.

To some degree these pitfalls are intrinsic, and they have even led some experts to stridently denounce model selection. However, certain features of the GLMSELECT procedure—in particular its extensive capabilities for customizing the selection and its flexibility and power in specifying complex potential effects—can partially mitigate these problems.

Model averaging approaches provide a way to make more stable inferences based on a set of models. Methods for doing this are presented in Burnham and Anderson (2002), and Bayesian approaches are discussed in Raftery, Madigan, and Hoeting (1997). However, when confronted with thousands of regressors you still have the issue of how to produce a reasonable set of candidate models to average, and this is where model selection comes into play.

This paper explores two examples of megamodel selection. The first example shows how effect selection can be used in the context of nonparametric modeling, where a large number of regressors are needed to provide the flexibility to model the unknown form of the underlying function. The second example, motivated by microarray data, shows how you can combine large scale variable selection with bootstrap methods to identify important regressors and produce averaged models whose predictive performance is competitive with other state-of-the-art methods in this field.

FEATURES OF PROC GLMSELECT

The main features of the GLMSELECT procedure are as follows:

- **Model Specification**
  - offers different parameterizations for classification effects
  - supports any degree of interaction (crossed effects) and nested effects
  - supports hierarchy among effects
  - provides for internal partitioning of data into training, validation, and testing roles
  - provides an EFFECT statement to generate spline, polynomial, multimember, and collection effects (new in SAS/STAT 9.2)
- **Selection Control**
  - provides multiple methods for effect selection
  - enables selection from a very large number of effects (tens of thousands)
  - offers selection of individual levels of classification effects
  - provides effect selection based on a variety of selection criteria
  - provides stopping rules based on a variety of model evaluation criteria
  - provides leave-one-out and k-fold cross validation
- **Display and Output**
  - produces graphical representation of the selection process
  - produces an output data set that contains predicted values and residuals
  - produces an output data set that contains the design matrix (new in SAS/STAT 9.2)
  - produces macro variables that contain selected models
  - supports parallel processing of BY groups
  - supports multiple SCORE statements

You can find detailed discussions of these features in Cohen (2006).

THE EFFECT STATEMENT

The experimental EFFECT statement, new in SAS/STAT 9.2, enables you to construct special collections of columns for the **X** matrix in your model. These collections are referred to as *constructed effects* to distinguish them from the usual model effects formed from continuous or classification variables.

The following *effect-types* are available:

COLLECTION	is a collection effect that defines one or more variables as a single effect with multiple degrees of freedom. During the selection process, the variables in a collection enter or leave as a unit.
MULTIMEMBER | MM	is a multimember classification effect whose levels are determined by one or more variables that appear in a CLASS statement. A typical example is the effect that teachers have on the performance of students, where each student has been taught by several teachers.
POLYNOMIAL | POLY	is a multivariate polynomial effect in the specified numeric variables.
SPLINE	is a regression spline effect whose columns are univariate spline expansions of one or more continuous variables. A spline expansion replaces the original variable with an expanded or larger set of new variables.

The first example in this paper uses the EFFECT statement to define spline effects. For details and examples of other uses of the EFFECT statement, see Chapter 42, "The GLMSELECT Procedure" (*SAS/STAT User's Guide*).

NONPARAMETRIC MODELING BY USING SPLINE EFFECTS

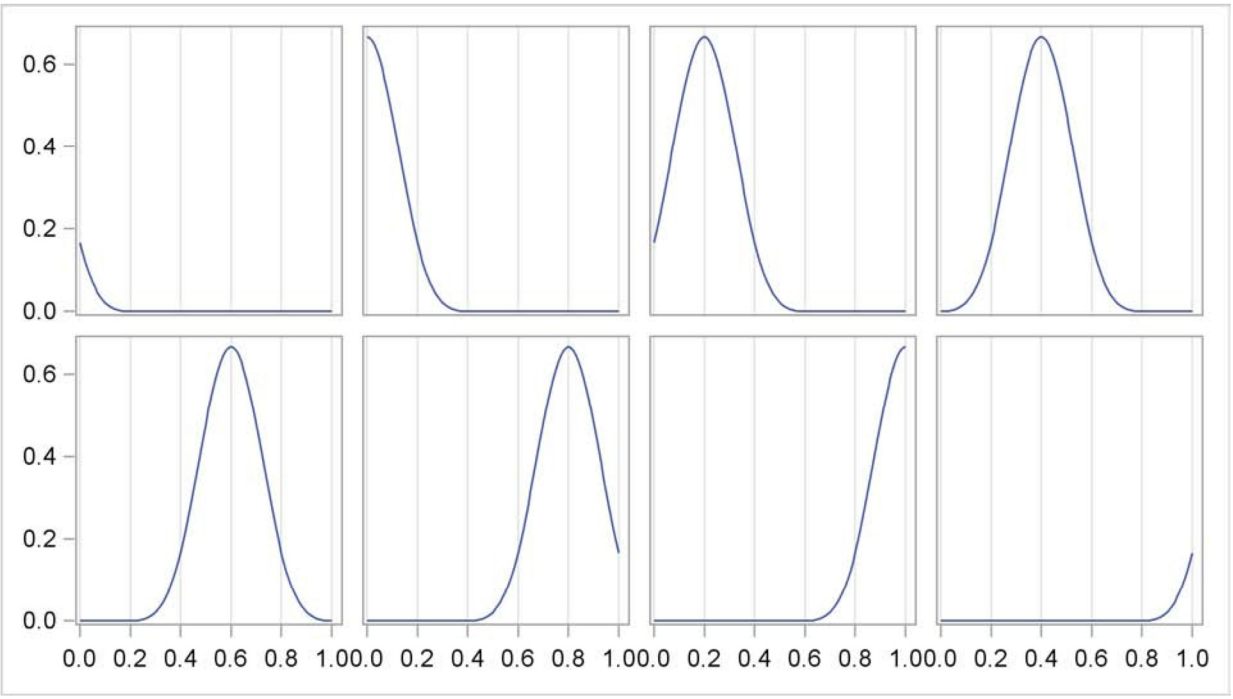
One approach to nonparametric fitting is to approximate the unknown underlying function as a linear combination of a set of basis functions. Once you specify the basis functions, then you can use least squares regression to obtain the coefficients of the linear combination. A problem with this approach is that for most data, you do not know a priori what set of basis functions to use. You need to supply a sufficiently rich set to enable the features in the data to be approximated. However, if you use too rich a set of functions, then this approach yields a fit that undersmooths the data and captures spurious features in the noise. Spline function bases provide a computationally convenient and flexible way to specify a rich set of basis functions, and variable selection can be used to obtain a parsimonious subset to prevent overfitting.

A spline function is a piecewise polynomial function where the individual polynomials have the same degree and connect smoothly at join points whose abscissa values, referred to as knots, are prespecified. By positioning an appropriate number of knots, you can use spline functions to parsimoniously fit curves to a wide variety of data. Although PROC GLMSELECT supports splines of any degree, this paper uses the cubic splines (the default) exclusively. Visually a cubic spline is a smooth curve, and it is the most commonly used spline when a smooth fit is desired.

Given a set of n distinct knots, any cubic spline can be represented as a linear combination of $n + 4$ basis functions. There are two widely used spline bases: namely, truncated power function bases and B-spline bases. Each cubic B-spline basis function has the property that its support (the set at which the function is nonzero) is an interval spanned by five adjacent knots. Hence you can associate local features in your data with particular B-spline basis functions. Truncated power bases do not share this property and are not used in this paper.

Figure 1 shows a cubic B-spline basis defined on $[0, 1]$ with four equally spaced knots at 0.2, 0.4, 0.6, and 0.8. Note that this basis consists of eight functions, each of which is nonzero over an interval that spans at most five knots.

Figure 1 Cubic B-Spline Basis with Four Equally Spaced Interior Knots



The following code simulates scatter plot data where the underlying true function is one period of the sine function $f(x) = \sin(2\pi x)$:

```
data Sine;
   do n=0 to 2048;
      x=n/2048;
      noise      = 0.5*rannor(12345);
      sine       = sin(2*constant('pi')*x);
      noisySine  = sine + noise;
      output;
   end;
run;
```

You can use an EFFECT statement to generate the columns in the design matrix that correspond to the basis shown in Figure 1. By selecting columns from this design matrix you can fit a smooth curve to the data as follows:

```
proc glmselect data=Sine;
   effect spl = spline(x/knotmethod=equal(4) split details);
   model  noisySine = spl;
   output out=sineOutCoarse p=predicted;
run;
```

The EFFECT statement in the preceding code creates a constructed effect named spl that consists of the eight cubic B-spline basis functions that correspond to the four equally spaced internal knots at 0.2, 0.4, 0.6, and 0.8 inside the range $[0, 1]$ of the variable x. Boundary knots outside the range of the data are also needed in defining this basis.

The DETAILS option in the EFFECT statement requests the knot information and the basis function information shown in Figure 2.

Figure 2 B-Spline Basis Information

The GLMSELECT Procedure

Knots for Spline Effect spl

Knot Number	Boundary	x
1	*	-0.4
2	*	-0.2
3	*	-0.0
4		0.2
5		0.4
6		0.6
7		0.8
8	*	1.0
9	*	1.2
10	*	1.4

The GLMSELECT Procedure

Basis Details for Spline Effect spl

Column	--Support-		Support Knots
1	-0.4	0.2	1-4
2	-0.4	0.4	1-5
3	-0.2	0.6	2-6
4	-0.0	0.8	3-7
5	0.2	1.0	4-8
6	0.4	1.2	5-9
7	0.6	1.4	6-10
8	0.8	1.4	7-10

The SPLIT option enables the eight columns in the design matrix that correspond to this basis to enter or leave the model individually. The individual basis functions in this split effect are named spl_1, spl_2,..., spl_8, and the corresponding labels are spl:1, spl:2,..., spl:8. Because no options are specified in the MODEL statement, the default stepwise selection method is used to select effects that are shown in the selected effects table in Figure 3.

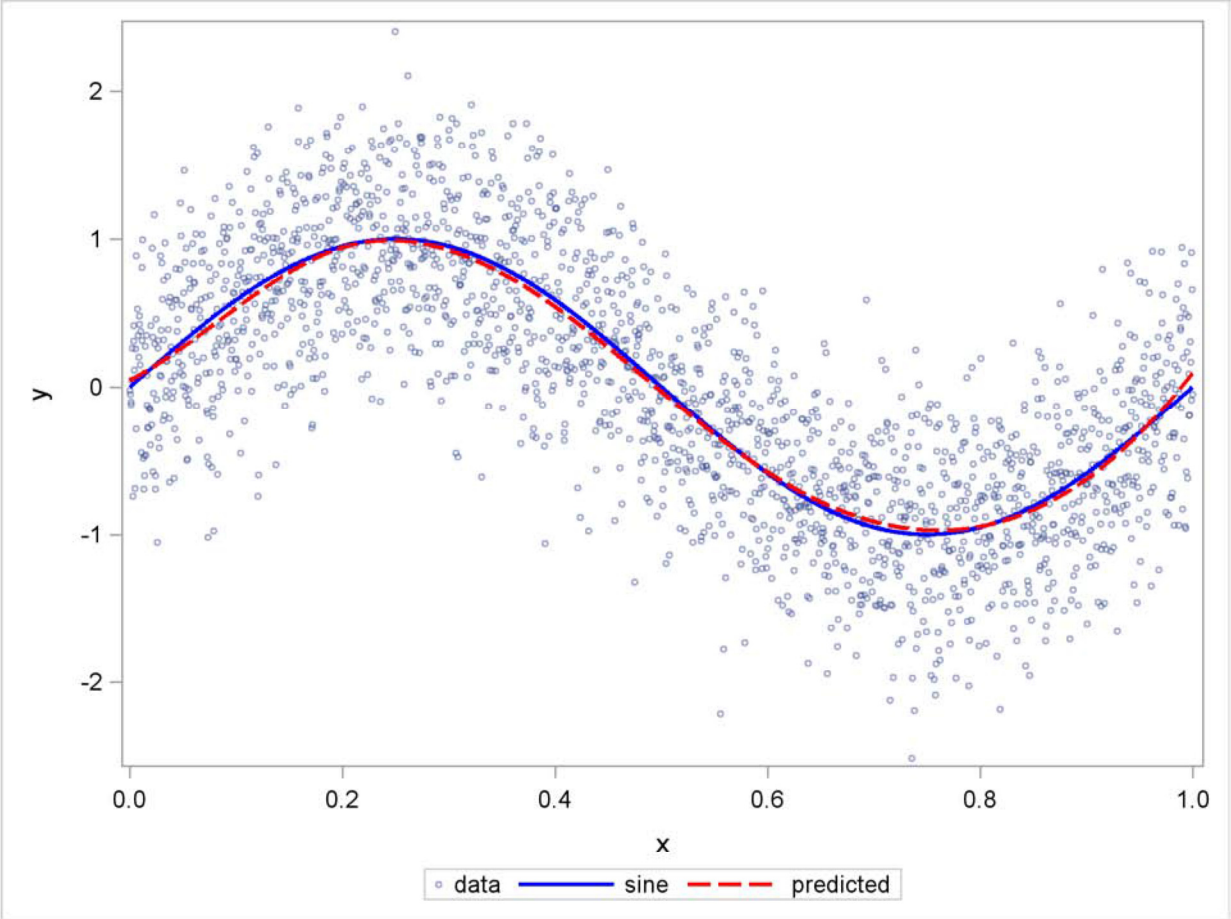
Figure 3 Selected Effects

```
Effects: Intercept spl:3 spl:4 spl:5 spl:6 spl:8
```

The OUTPUT statement captures the predicted values in an output data set. You can produce the fit plot in Figure 4 by using the SGPLOT procedure as follows:

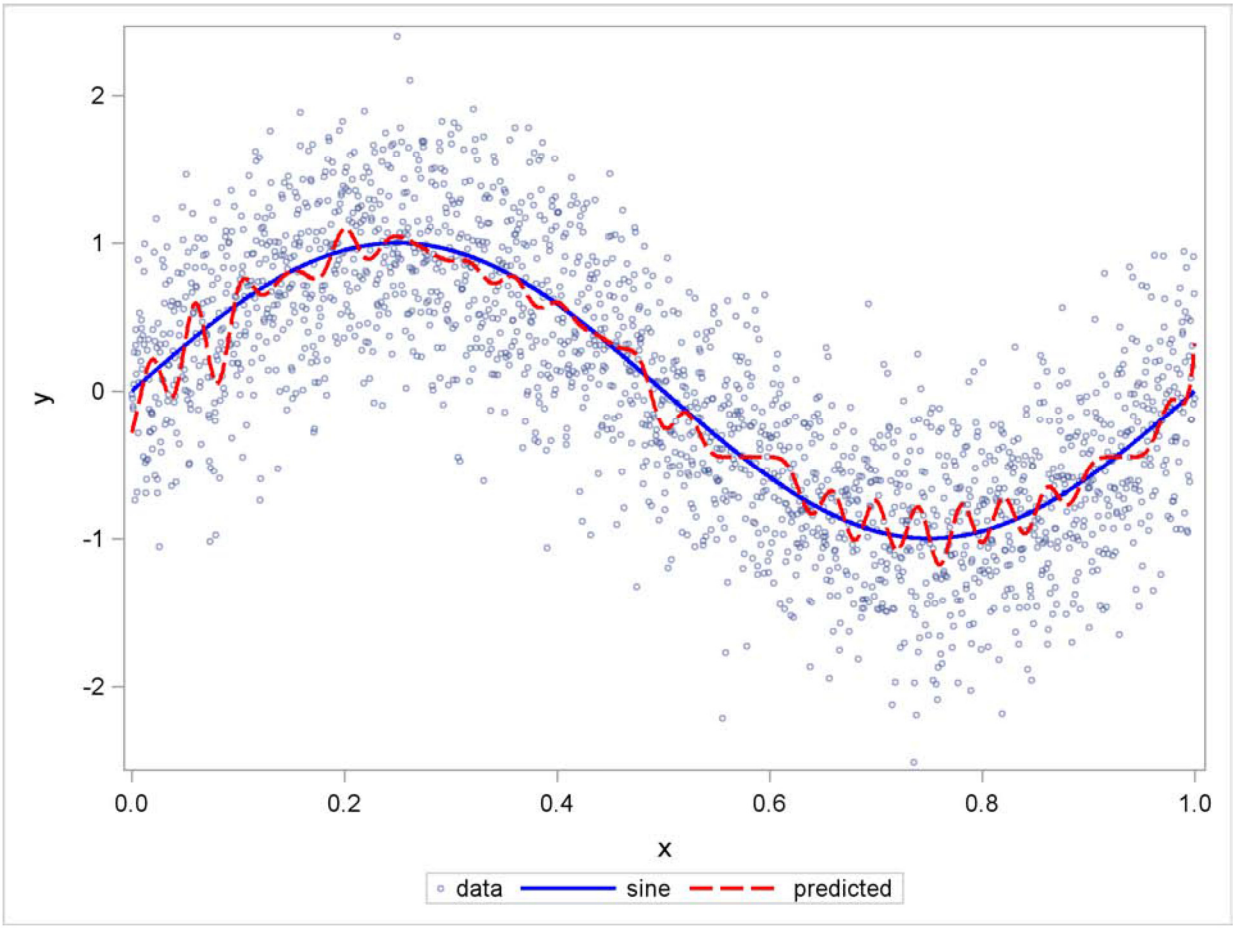
```
proc sgplot data=sineOutCoarse;
   yaxis label='y';
   scatter x=x y=noisySine/transparency=0.5
                           markerAttrs = (size=3)
                           legendLabel = 'data';
   series  x=x y=sine/lineattrs=(color=blue thickness=2);
   series  x=x y=predicted/lineattrs=(color=red thickness=2 pattern=mediumDash);
run;
```

Figure 4 Fit Plot with Four Internal Knots



You can see that selecting basis functions from the B-spline basis enables you to obtain a successful fit. Can you improve the fit by using a much finer set of knots? Figure 5 shows that the answer is no. In this case, 49 equally spaced knots are used. The distance between adjacent knots is $1/50$, and so the support width of each cubic B-spline basis is 0.08 $(4/50)$. Because each basis function is supported on such a small fraction of the range of the data, many of these basis functions are needed in the final fit; this results in the overfitting that you clearly see in Figure 5.

Figure 5 Fit Plot with 49 Internal Knots



Suppose you suspect that your data have some local features that you want to capture in a nonparametric fit based on spline approximations. If you know the locations of these local features, you can accommodate this by using unequally spaced knots with more knots concentrated around the locations of these local features. Cases where such a priori information is not available appear to be problematic. In order to capture local features whose locations are not known, you need to use many knots dispersed over the entire range of the data. But then how do you avoid the overfitting? One approach, known as the penalized B-spline method (Eilers and Marx 1996), adds a penalty to the least squares objective function that increases as the roughness of the fit increases. By appropriately choosing a smoothing parameter that adjusts the size of this roughness penalty, you can balance the goodness of the fit (bias) with the roughness (variance) of the fit. However, because only one parameter controls this bias variance tradeoff, you cannot effectively model data that have local features that occur at different scales. In such cases, effect selection from several sets of B-spline bases that correspond to different scales in the data proves to be an effective strategy.

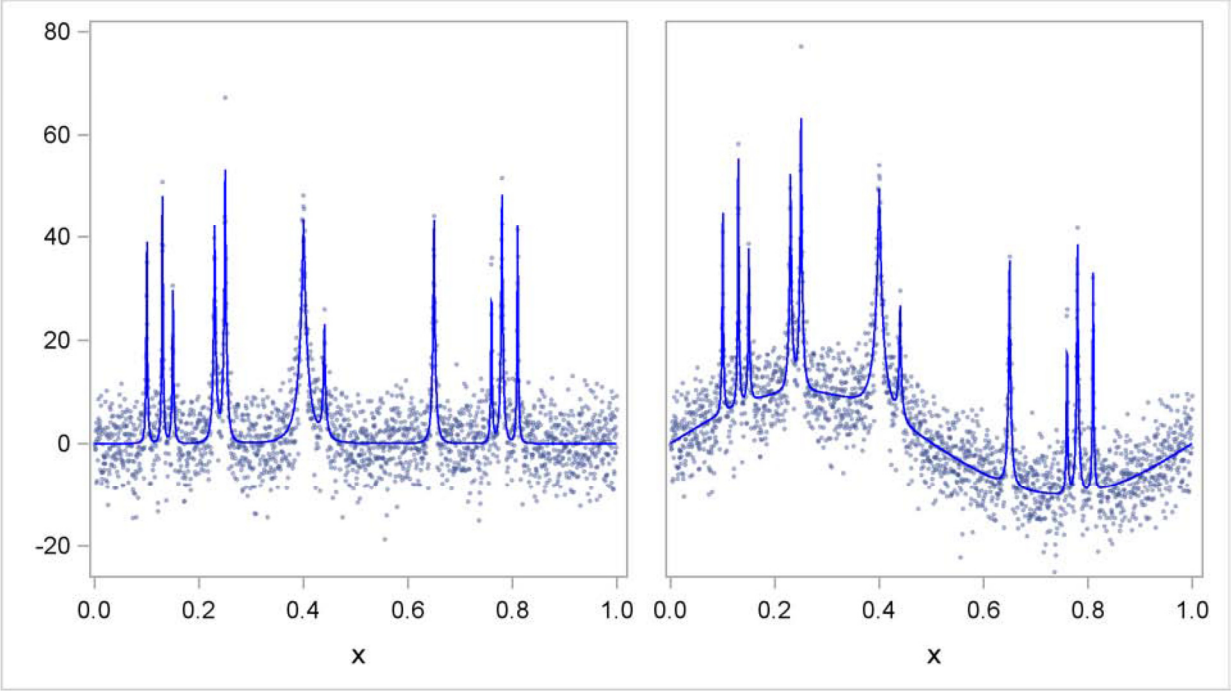
To illustrate this approach, data for this example are adapted from a set of benchmarks developed by Donoho and Johnstone (1994), which have become popular in the statistics literature. The particular benchmark used is the "Bumps" function to which random noise has been added to create the test data. The following DATA step, adapted from Sarle (2001), creates the data:

```
%let random=12345;

data DoJoBumps;
   keep x bumps noisyBumps bumpsSine noisyBumpsSine;

   do n=1 to 2048;
      x=n/2048;
      link compute;
      noisyBumps     = bumps+5*rannor(&random);
      baseSine       = 10*sin(2*constant('pi')*x);
      bumpsSine      = baseSine + bumps;
```

```
      noisyBumpsSine = baseSine + noisyBumps;
      output;
   end;
stop;

compute:
   array t(11) _temporary_ (.1 .13 .15 .23 .25 .4 .44 .65 .76 .78 .81);
   array b(11) _temporary_ (   4    5    3   4   5 4.2 2.1 4.3  3.1  5.1  4.2);
   array w(11) _temporary_ (.005 .005 .006 .01 .01 .03 .01 .01 .005 .008 .005);

   bumps=0;
   do i=1 to 11;
      bumps=bumps+b[i]*(1+abs((x-t[i])/w[i]))**-4;
   end;
   bumps=bumps*10.528514619;
return;
run;
```

The constants are chosen so that the noise-free bumps data have a standard deviation of 7. The standard deviation of the noise is 5, yielding a signal noisyBumps with a signal-to-noise ratio of 1.4 (7/5). For the purpose of highlighting signals with multiple scales, this DATA step also creates a signal bumpsSine by adding a sine curve with period 1 to the original bumps curve, yielding a modified bumps function with a sinusoidal baseline for the bumps. Figure 6 shows both the original and modified bumps functions superimposed on the data.

Figure 6 Bumps Function and Sinusoidal Baseline Variant



Suppose you want to fit a smooth curve to the data in the panel on the right in Figure 6. You have seen that in order to capture the sinusoidal baseline behavior, a B-spline basis with about four internal knots is appropriate. However, such a basis does not have the resolution necessary to capture the bumps. On the other hand, if you use a much finer set of knots to define the B-spline basis, then you can resolve the bumps but at the expense of overfitting the data in the regions between the bumps. A way out of this impasse is to provide B-spline bases at multiple scales and use model selection to obtain a parsimonious subset. The following code prototype shows how you can do this:

```
proc glmselect data=DoJoBumps;
   effect spl1 = spline(x/knotmethod=equal(1) split);
   effect spl2 = spline(x/knotmethod=equal(2) split);
   effect spl4 = spline(x/knotmethod=equal(4) split);
   effect spl8 = spline(x/knotmethod=equal(8) split);

   model  noisyBumpsSine = spl1 spl2 spl4 spl8;
run;
```

This method of specifying spline bases at multiple scales can be cumbersome if you want to use a large number of scales. You can use the KNOTMETHOD=MULTISCALE option in a single EFFECT statement to accomplish this more succinctly, as illustrated in the following code:

```
proc glmselect data=DoJoBumps;
   effect spl = spline(x/knotmethod=multiscale(endscale=8) details split);
   model  noisyBumpsSine = spl;
   output out=bumpsSineOut p=predicted;
run;
```

At scale i, the associated B-spline basis corresponds to 2^i equally spaced internal knots. For each scale, a separate spline effect is generated. The name of the constructed spline effect at scale i is formed by appending _Si to the effect name you specify in the EFFECT statement. The ENDSCALE=8 option requests that the finest scale use cubic B-splines defined on 2^8 equally spaced knots in the interval $[0, 1]$. Because the cubic B-splines are nonzero over five adjacent knots, at this finest scale the support of each B-spline basis function is an interval of length about 0.02 $(4/257)$, enabling the bumps in the underlying data to be resolved. The default value is ENDSCALE=7. At this scale you can still capture the bumps, but with less sharp resolution. For these data, using a value of ENDSCALE greater than 8 provides unneeded resolution, making it more likely that basis functions that fit spurious features in the noise are selected.

The DETAILS option in the EFFECT statement requests the display of spline knots and spline basis tables. These tables contain information about knots and basis functions at all scales. The results for scale 4 are shown in Figure 7 and Figure 8.

Figure 7 Spline Knots

Knots for Spline Effect spl

Knot Number	Scale	Scale Knot Number	Boundary	x
40	4	1	*	-0.1171
41	4	2	*	-0.0583
42	4	3	*	0.0005
43	4	4		0.0593
44	4	5		0.1181
45	4	6		0.1769
46	4	7		0.2357
47	4	8		0.2945
48	4	9		0.3533
49	4	10		0.4121
50	4	11		0.4708
51	4	12		0.5296
52	4	13		0.5884
53	4	14		0.6472
54	4	15		0.7060
55	4	16		0.7648
56	4	17		0.8236
57	4	18		0.8824
58	4	19		0.9412
59	4	20	*	1.0000
60	4	21	*	1.0588
61	4	22	*	1.1176

Figure 8 Spline Details

Basis Details for Spline Effect spl

Column	Scale	Scale Column	Support		Support Knots
32	4	1	-0.1171	0.0593	1–4
33	4	2	-0.1171	0.1181	1–5
34	4	3	-0.0583	0.1769	2–6
35	4	4	0.0005	0.2357	3–7
36	4	5	0.0593	0.2945	4–8
37	4	6	0.1181	0.3533	5–9
38	4	7	0.1769	0.4121	6–10
39	4	8	0.2357	0.4708	7–11
40	4	9	0.2945	0.5296	8–12
41	4	10	0.3533	0.5884	9–13
42	4	11	0.4121	0.6472	10–14
43	4	12	0.4708	0.7060	11–15
44	4	13	0.5296	0.7648	12–16
45	4	14	0.5884	0.8236	13–17
46	4	15	0.6472	0.8824	14–18
47	4	16	0.7060	0.9412	15–19
48	4	17	0.7648	1.0000	16–20
49	4	18	0.8236	1.0588	17–21
50	4	19	0.8824	1.1176	18–22
51	4	20	0.9412	1.1176	19–22

The “Dimensions” table in Figure 9 shows that at each step of the selection process, 548 effects are considered as candidates for entry or removal. Note that although the MODEL statement specifies a single constructed effect spl, the SPLIT suboption specifies that each of the parameters in this constructed effect be treated as an individual effect. You see that selecting individual basis functions from spline bases at multiple scales leads to large scale variable selection problems.

Figure 9 Dimensions

Dimensions

Number of Effects	548
Number of Parameters	548

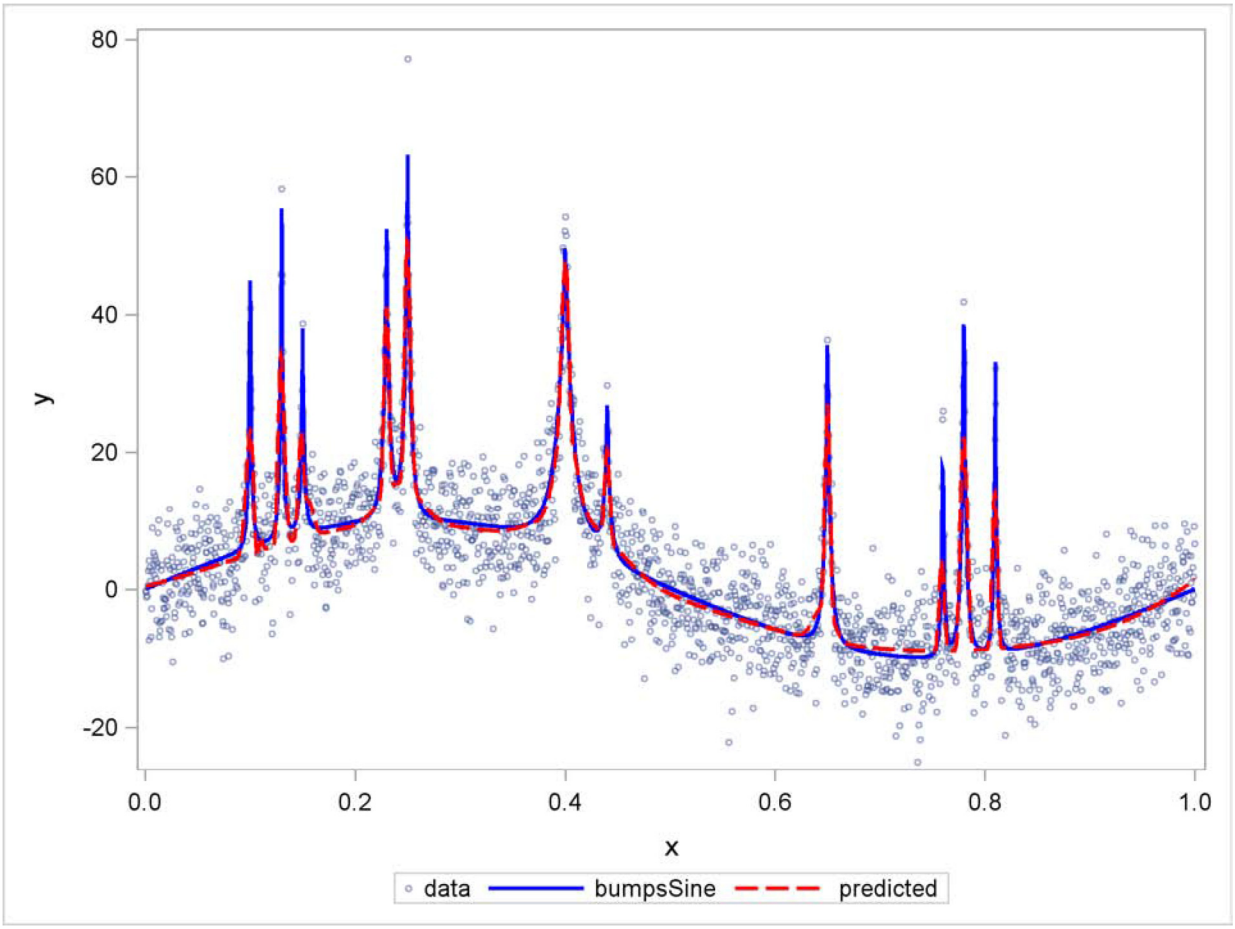
Figure 10 shows the parameter estimates for the selected model. The model, selected by the default stepwise method, contains a small number of B-spline basis functions from multiple scales. The fit plot in Figure 11 shows that this model accurately captures both the low-frequency sinusoidal baseline and also the bumps, without overfitting the regions between the bumps.

Figure 10 Parameter Estimates

Parameter Estimates

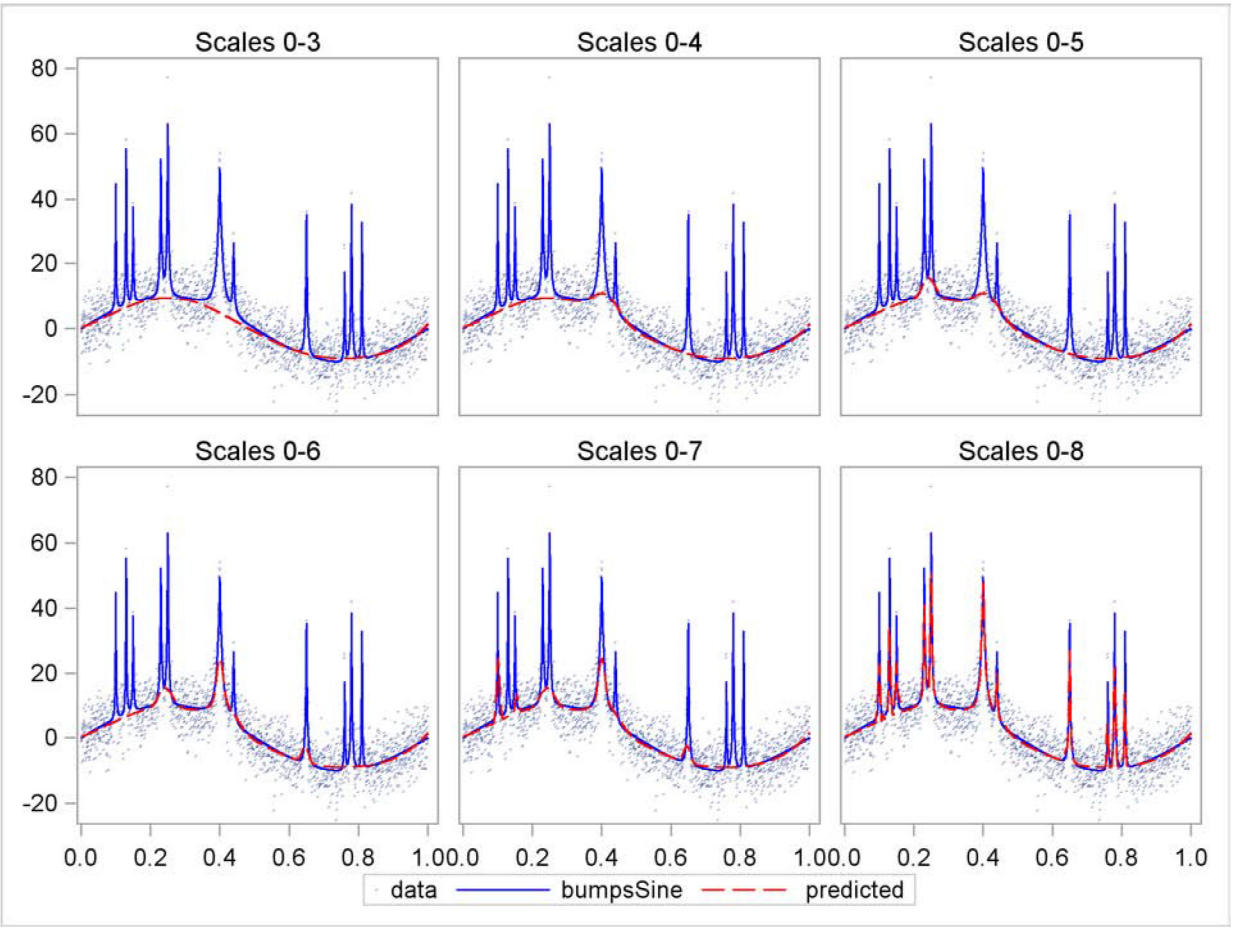
Parameter	DF	Estimate	Standard Error	t Value
Intercept	1	-2.111956	0.768772	-2.75
spl_S0:3	1	-18.424959	2.048443	-8.99
spl_S1:3	1	27.895811	1.864830	14.96
spl_S2:3	1	5.625511	1.656240	3.40
spl_S2:8	1	39.553382	5.819788	6.80
spl_S4:9	1	9.779992	1.430574	6.84
spl_S5:10	1	8.964383	1.523382	5.88
spl_S6:28	1	20.378594	2.810779	7.25
spl_S6:44	1	7.466644	1.868956	4.00
spl_S7:15	1	28.976992	2.560599	11.32
spl_S7:22	1	7.895816	2.160341	3.65
spl_S8:29	1	-18.081751	3.541369	-5.11
spl_S8:35	1	36.898716	3.164351	11.66
spl_S8:36	1	14.734931	3.168718	4.65
spl_S8:40	1	21.064344	2.932460	7.18
spl_S8:61	1	40.378057	3.067269	13.16
spl_S8:66	1	49.899718	3.370580	14.80
spl_S8:67	1	12.269691	3.267487	3.76
spl_S8:104	1	17.032223	3.641212	4.68
spl_S8:105	1	29.080374	3.796361	7.66
spl_S8:115	1	19.432295	2.931405	6.63
spl_S8:169	1	44.868001	3.529586	12.71
spl_S8:197	1	19.733357	2.778229	7.10
spl_S8:202	1	36.232431	3.153724	11.49
spl_S8:203	1	27.185588	3.153133	8.62
spl_S8:210	1	34.145250	2.772482	12.32

Figure 11 Fit by Selecting B-splines



You can see how the fit is built at multiple scales in the panel in Figure 12. Each plot in the panel shows the partial fit that you obtain if you use the estimated parameters from the selected model but just from the indicated scales. You see that the basis functions at scales 0–3 accurately model the slowly varying sinusoidal baseline. The basis functions at scales 4 through 7 capture the lower portion of the wide bumps. The tops of the bumps are captured by basis functions at scale 8.

Figure 12 Multiresolution Analysis



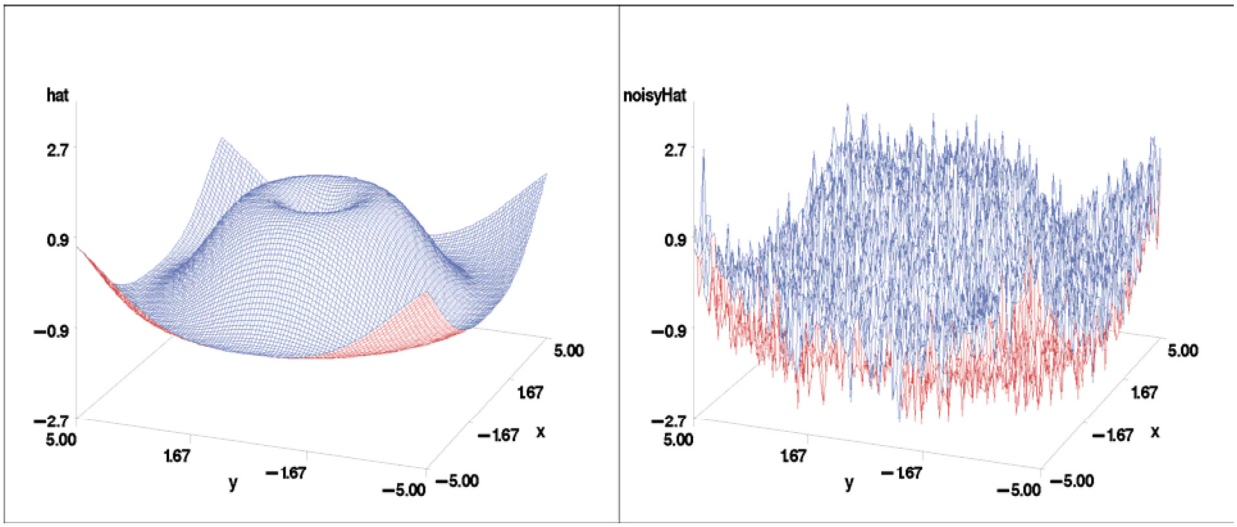
MULTIVARIATE NONPARAMETRIC FITTING

The preceding example shows how you can use variable selection with B-spline bases at multiple scales to fit functions of a single variable. The same ideas extend to multivariate functions by using products of one-dimensional B-spline bases. This section provides a two-dimensional example.

The following code generates the noisy data for the “hat” function that are displayed to the right of the true function in Figure 13:

```
data hat;
   do y=-5 to 5 by 0.1;
      do x=-5 to 5 by 0.1;
         hat      = sin(sqrt(x*x+y*y));
         noisyHat = hat + 0.5*rannor(1);
         output;
      end;
   end;
run;
```

Figure 13 Hat Function and Noisy Data



Suppose you have a basis, B_1, of m univariate functions defined on an interval $[a, b]$ and another basis, B_2, of m univariate functions defined on an interval $[c, d]$. Then you can define a basis, known as a tensor product basis, of bivariate functions defined on the rectangle $[a, b] \times [c, d]$ that consists of all mn pairwise products of one function from each basis. In particular, you can do this with the B-spline bases used in the preceding section, and you can use such a basis to fit a function to a response that depends on two regressors. If you want to fit such a response nonparametrically, you can use all pairwise products bases defined at multiple scales for each predictor together with variable selection to obtain a parsimonious subset. The following code show how you can do this to smooth the noisy hat data:

```
proc glmselect data=hat;
   effect spl_x = spline(x/knotmethod=multiscale(endscale=3) split);
   effect spl_y = spline(y/knotmethod=multiscale(endscale=3) split);
   model noisyHat = spl_x | spl_y;
   output out=outHat1 p=predHat;
run;
```

You use an effect statement that defines B-spline bases at multiple scales for each regressor. The ENDSCALE=3 option specifies that at the finest scale eight equally spaced internal knots are used. This provides sufficient resolution to model the hat function. The PROC GLM-type effect specification spl_x | spl_y is a concise way of specifying a model with both main effects spl_x and spl_y and their interaction spl_x * spl_y.

Since the split option is used on both EFFECT statements, each column in the design matrix can enter or leave the model individually. Furthermore, because the number of columns in the interaction is the product of the number of basis functions for each regressor, fitting nonparametric models by selecting from interactions of spline bases at multiple scales leads to large model selection problems. The dimension table in Figure 14 shows that this model yields a variable selection problem with 1,024 regressors.

Figure 14 Dimensions

The GLMSELECT Procedure

Dimensions

Number of Effects	1024
Number of Parameters	1024

The fit is displayed in the left panel of Figure 15. You see that the default stepwise selection successfully smooths the noisy data and recovers a smooth approximation of the underlying true hat function.

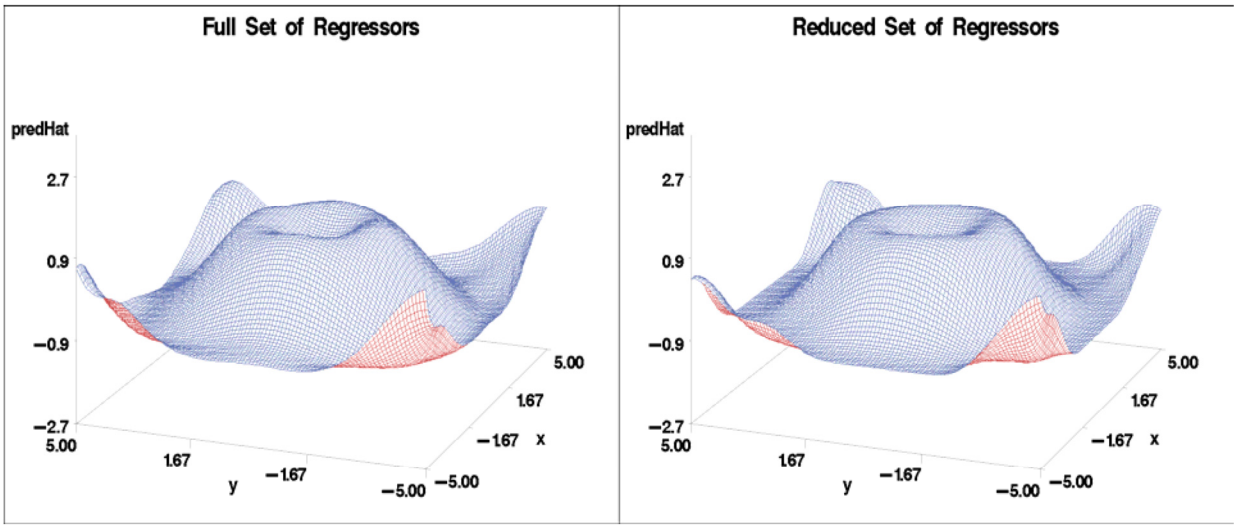
You can extend this approach to more than two dimensions. However, such a fully nonparametric approach in higher dimensions generates a huge variable selection problem. Usually, you can use prior knowledge to make an informed choice of a more appropriate set of basis functions than simply taking tensor products of univariate B-spline bases at

multiple scales. For example, you might know that local features in your data are radially symmetric. In this case, using only radially symmetric basis functions at the finer scales that capture local features is appropriate. In the context of tensor product spline bases, you obtain locally radially symmetric basis functions by using only interactions of splines that are defined with the same equal spacing of knots for each regressor. Another popular method is to use radial basis functions instead of tensor products of splines. You can apply the preceding reasoning in fitting the data in this example. The following code shows how you do this:

```
proc glmselect data=hat;
   effect spl_x  = spline(x/knotmethod=multiscale(endscale=2) split);
   effect spl_x8 = spline(x/knotmethod=equal(8) split);

   effect spl_y  = spline(x/knotmethod=multiscale(endscale=2) split);
   effect spl_y8 = spline(y/knotmethod=equal(8) split);

   model noisyHat = spl_x | spl_y  spl_x8 * spl_y8;

   output out=outHat2 p=predHat;
run;
```

Note that basis functions at the finest scale are defined separately as constructed effects spl_x8 and spl_y8. These two constructed effects interact with each other but not with basis functions at the coarser scales that are defined by constructed effects spl_x and spl_y. This produces at total of 544 regressors from which 47 are selected to yield the fit shown the right panel in Figure 15. Note that the second GLMSELECT step chooses from about half the number of regressors as the original GLMSELECT step in this example.

Figure 15 Fits by Selecting B-Splines



You see that both fitted surfaces in Figure 15 are successful in capturing the underlying hat function from the noisy data.

MODEL SELECTION FOR MICROARRAY DATA

Microarrays are arrangements of microscopic spots of short segments of genes on glass or silicon chip substrates, with the number of spots ranging from the hundreds to tens of thousands. A probe is formed by attaching fluorescent labels to the DNA of a sample being investigated and letting this material attach to the genetic material in the microarray. The intensity of the fluorescence at each spot in the microarray yields expression levels of how much of specific DNA fragments is present in the sample. A typical experiment consists of using samples from two classes of cells that correspond to the presence or absence of a property under investigation. Statistical challenges are to identify the genes that are predictive of this property and to build a reliable predictive model that can be used to classify new samples based on their gene expression profiles.

Usually the data for an experiment consist of a few hundred samples that form the training data used for building the model and an equal or greater number of samples that comprise the test data used for validating the model. Hence,

you are confronted with challenging statistical problems that have just hundreds of observations but tens of thousands of regressors.

The second phase of the "MicroArray Quality Control (MAQC) Project," a large collaborative effort between the FDA, industry, and academia, focuses on developing best practices for the development and validation of predictive signatures and classifiers for microarray data that the FDA aims to publish by the end of 2009. You can find details about this project and references to related publications and meetings at the Web site http://www.fda.gov/nctr/science/centers/toxicoinformatics/maqc/. As part of this project, Russ Wolfinger, director of scientific discovery and genomics at SAS, and many other groups have investigated a wide variety of different statistical approaches for building predictive classifiers that use the data sets from the MAQC-II initiative. Wolfinger and his team have shown that variable selection performed with the GLMSELECT procedure can produce predictive classifiers whose performance is competitive with other state-of-the-art techniques (Wolfinger, 2008).

The following example shows how you can use the GLMSELECT procedure to build such classifiers, and it uses bootstrap techniques to extend the analyses carried out by Wolfinger. The data sets used in the MAQC-II study are protected by confidential disclosure agreements, and so instead this paper uses simulated data that is designed to have properties and dimensionality consistent with the data in this study.

This example is organized into the following steps:

1. Create the simulated data.
2. Build a classifier with an optimal logistic model.
3. Build a classifier with stepwise selection.
4. Build a classifier with bootstrap-based model averaging.
5. Build a classifier with stepwise selection and validation.

Create the Simulated Data

The following DATA step creates the simulated training and test data sets, each with 300 observations and 35,000 explanatory variables. The binary response y, coded as 0 or 1, is constructed to depend in decreasing order of importance on just the first 10 regressors, which are collectively referred to as the "germane" regressors. The values of y are chosen to be consistent with the probabilities computed by applying an inverse logit transform to a linear predictor formed from the germane regressors.

```
%let nTest  =   300;
%let nTrain =   300;
%let nVars  = 35000;

data train test;
  array x{&nVars};
  do obsNum=1 to &nTest+&nTrain;

     /*--- randomly assign the regressors ---*/
     do j=1 to &nVars;
       x{j}=ranuni(1);
     end;

     /*--- Form a linear predictor that   ---*/
     /*--- depends on x1-x10              ---*/
     linp =  11*x1 - 10*x2 + 9*x3 - 8*x4 + 7*x5
             -6*x6 + 5*x7  - 4*x8 + 3*x9 - 2*x10;

     /*--- Use an inverse logit transform to assign    ---*/
     /*--- the probability that the response will be 1 ---*/
     TrueProb = 1/(1+exp(-linp));

     /*--- Assign the binary response to be ---*/
     /*--- consistent with the probability  ---*/
     if ranuni(1) < TrueProb then y=1;
                             else y=0;
```

```
      /*--- Output: nTrain observations to train data ---*/
      /*---         nTest  observations to test data  ---*/
      if obsNum<= &nTrain then output train;
                          else output test;
  end;
run;
```

The following code creates a macro classify that classifies observations in an input data set into classes 0 or 1 based on whether a probability is less than or greater than 0.5, respectively. Then it uses the FREQ procedure to count how many times this predicted classification agrees with the observed binary response.

```
%macro classify(source,var,probability);
   data &source;
      length &var $9;
      set &source;
      if &probability<= 0.5 then yPred = 0;
                            else yPred = 1;

      if y = yPred then &var = 'Correct';
                   else &var = 'Incorrect';
   run;

   proc freq data=&source;
      table &var / nocum;
   run;
%mend;
```

Figure 16 shows the results of applying this macro to the train and test data sets as follows:

```
%classify(train,Train,TrueProb);
%classify(test,Test,TrueProb);
```

You see that the response corresponds to its most likely value for about 95% of the observations in the training data and 92% of the observations in the test data. These percentages are bounds on the predictive accuracy that a good statistical model for these data should achieve.

Figure 16 Training and Test Set Classification for True Probability

The FREQ Procedure

Train	Frequency	Percent
Correct	284	94.67
Incorrect	16	5.33

Test	Frequency	Percent
Correct	277	92.33
Incorrect	23	7.67

Classification with an Optimal Logistic Model

Generalized linear models are often used to model data with a binary response. The special case of a linear logistic model assumes that the response follows a binomial distribution where the expected value μ_i of the response y_i is related to a linear combination of the explanatory variables by

$$g(\mu_i) = x_i'\beta$$

where g is the logit link function defined by $g(\mu) = \log\left(\frac{\mu}{1-\mu}\right)$. The response in this example is related to a linear combination of the germane regressors via an inverse logit transform, and so a linear logistic model that uses just the germane regressors is an optimal model.

In contrast to logistic models, the GLMSELECT procedure performs model selection under the usual linear model assumptions where the link function is the identity function and the error distribution is Gaussian. Using the GLMSELECT procedure for classification with a binary response violates these assumptions. Nevertheless, as the following analyses

show, careful use of linear variable selection can yield models whose predictive performance is competitive with the optimal logistic model. This approach of doing classification by using variable selection in very large linear models has also been used successfully in areas other than microarray analysis. For example, Foster and Stine (2004) use a modified version of linear stepwise selection to build a predictive model for bankruptcy from over 67,000 possible predictors and show that this yields a model whose predictions compare favorably with other recently developed data mining tools.

In order to obtain a reference point for comparing models produced by using PROC GLMSELECT with an optimal model for the simulated data in this example, the following code fits a logistic model that uses the unrealistic prior knowledge of the germane regressors:

```
proc logistic data=train;
   model y(event='1') = x1-x10;
   output          out=trainOut(keep=y BestEst) p=BestEst;
   score data=test out=testOut (rename=(P_1=BestEst) keep=y P_1);
run;
%classify(trainOut,Train,BestEst);
%classify(testOut,Test,BestEst);
```

The classification results in Figure 17 show that this optimal logistic model correctly classifies 95% of the training data and 90.33% of the test data, consistent with the expected accuracy bounds for this simulated data.

Figure 17 Training and Test Set Classification Errors for Optimal Logistic Model

The FREQ Procedure

Train	Frequency	Percent
Correct	285	95.00
Incorrect	15	5.00

Test	Frequency	Percent
Correct	271	90.33
Incorrect	29	9.67

For real microarray data, prior knowledge of what regressors to use is not available. In such cases, a simple approach that you might consider is to use a logistic model with the subset of the regressors that have the highest pairwise correlations with the response. There are several problems associated with this simple approach:

1. Regressors that are important in a multivariate analysis might have small pairwise correlations with the response.
2. Even after ranking regressors by their correlation with the response, you still need to determine how many of these regressors to use.
3. With the small number of observations and the very large number of regressors, this method often selects many regressors that have very little out-of-sample predictive utility but by chance have relatively large correlations with the response in the training data.

Multivariate variable selection methods such as stepwise selection partially address the first problem but are impacted by the multivariate analogues of the latter two problems. You still need to determine an appropriate stopping rule, and you have to deal with multiple comparison issues at each step of the selection process. A further difficulty is that traditional implementations of variable selection techniques for both linear and logistic regression start by assembling the full $\mathbf{X}'\mathbf{X}$ matrix, where $\mathbf{X}$ is the design matrix. With 35,000 regressors, the computational cost of forming and storing this matrix is very large, making such techniques impractical on most current-generation computers.

Classification with Linear Stepwise Selection

The GLMSELECT procedure has features that enable you to mitigate all of the preceding problems. You can use validation and data resampling to address issues of stopping rules and selection bias. For wide data, the GLMSELECT procedure builds the relevant portions of the $\mathbf{X}'\mathbf{X}$ matrix as the selection process proceeds, at the expense of requiring a pass through the data at each step. When the number of regressors that are selected is a small fraction of the

total number of regressors (as occurs for microarray data), this approach yields substantial memory and computational savings, making such analyses tractable on even modest hardware.

The following code provides a starting point for the analysis:

```
ods graphics on;
proc glmselect data=train plots=(criteria ase) testdata=test seed=1;
   model y = x1-x&nVars / selection=stepwise(choose=cv maxsteps=30)
                          cvMethod=random(5);
   output          out=trainOut(keep=y predGls)  p=predGls;
   score data=test out=testOut (keep=y predGls)  p=predGls;
run;
ods graphics off;
%classify(trainOut,Train,predGls);
%classify(testOut,Test,predGls);
```

The default stepwise selection method with variable selection and stopping based on the Schwarz Bayesian information criterion (SBC) is used. If the SBC values decrease monotonically at each of the first 30 steps, the MAXSTEPS=30 option stops the stepwise selection process at step 30. The CHOOSE=CV option, in concert with the CVMETHOD=RANDOM(5) option, specifies that among the models at each step, the selected model is the one that yields the lowest predicted residual sum of squares obtained by using fivefold cross validation. This statistic is computed by randomly splitting the data into five approximately equal-sized parts. One of these parts is held out for validation, and the model is fit on the remaining four parts. This fitted model is used to compute the predicted residual sum of squares on the omitted part, and this process is repeated for each of five parts. The sum of the five predicted residual sum of squares so obtained is the CVPRESS statistic.

When you enable ODS Graphics and request the PLOTS=CRITERIA option in the PROC GLMSELECT statement, you obtain the panel of fit criteria shown in Figure 18. You can see that the CVPRESS criterion and all the other fit criteria shown are still decreasing at step 30, which thus becomes the selected step.

Figure 18 Panel of Fit Criteria

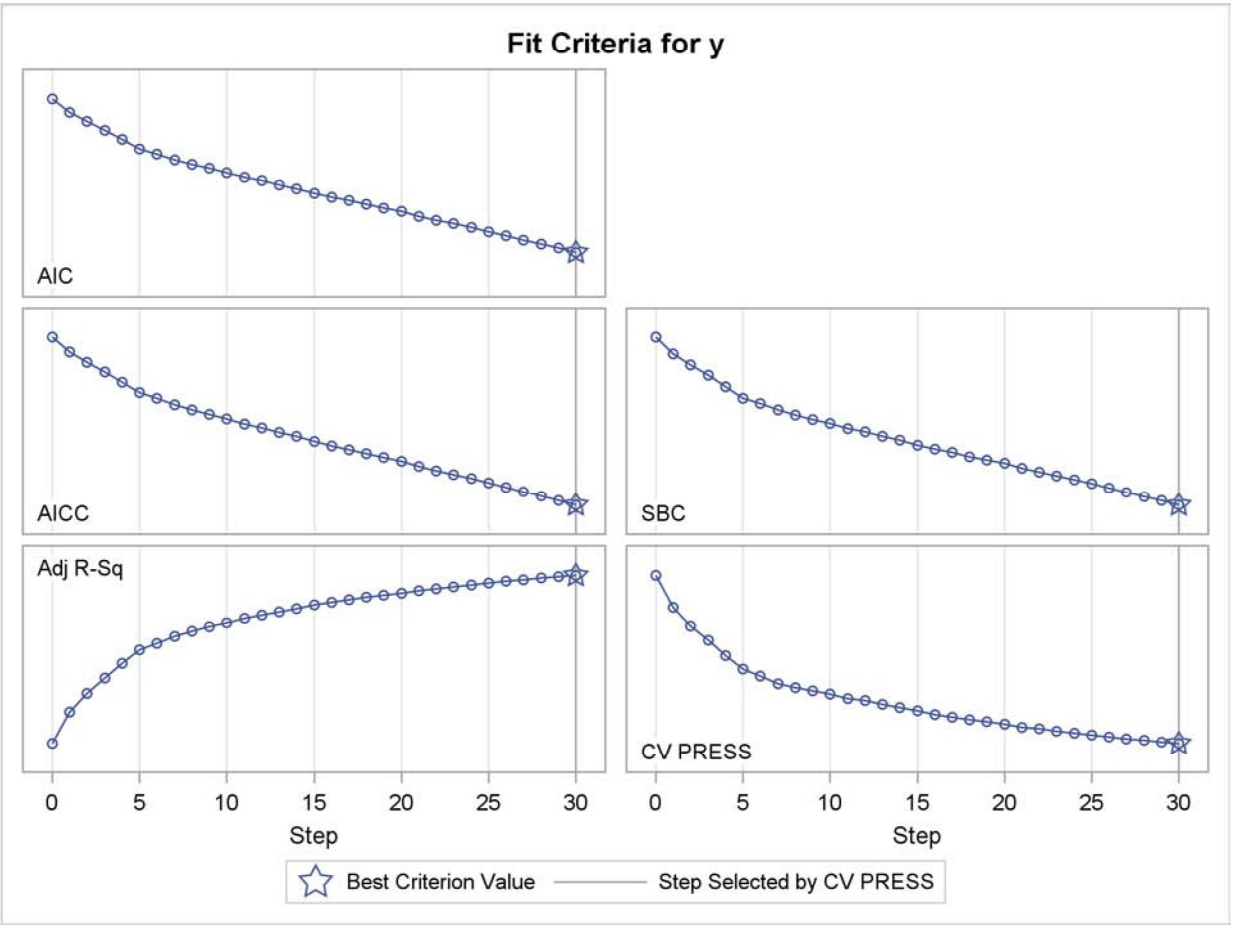


Figure 19 shows the effects in the selected model. You see that although all but two of the germane regressors are selected, most of the regressors selected are not systematically related to the response. This indicates that the selected model overfits the training data and so generalizes poorly to new data.

Figure 19 Selected Effects

```
Effects: Intercept x1 x2 x3 x4 x5 x6 x7 x9 x2291 x4062 x4356 x4745 x4996 x6028
         x6096 x9026 x11002 x11126 x11735 x13702 x14969 x16976 x17543 x18477
         x24775 x25042 x25982 x32233 x32762 x33126
```

You can see the effects of this overfitting in the classification results in Figure 20; these results show that 100% of the observations in the training data but just 76% of the observations in the test data are correctly classified. Note that the predicted responses produced by PROC GLMSELECT are not probabilities. Classification is done by using 0.5 (the midpoint of the response coding values) as the critical value and assigning observations to classes 0 or 1 depending on which side of the critical value the predicted response falls.

Figure 20 Training and Test Set Classification Errors for GLMSELECT Model

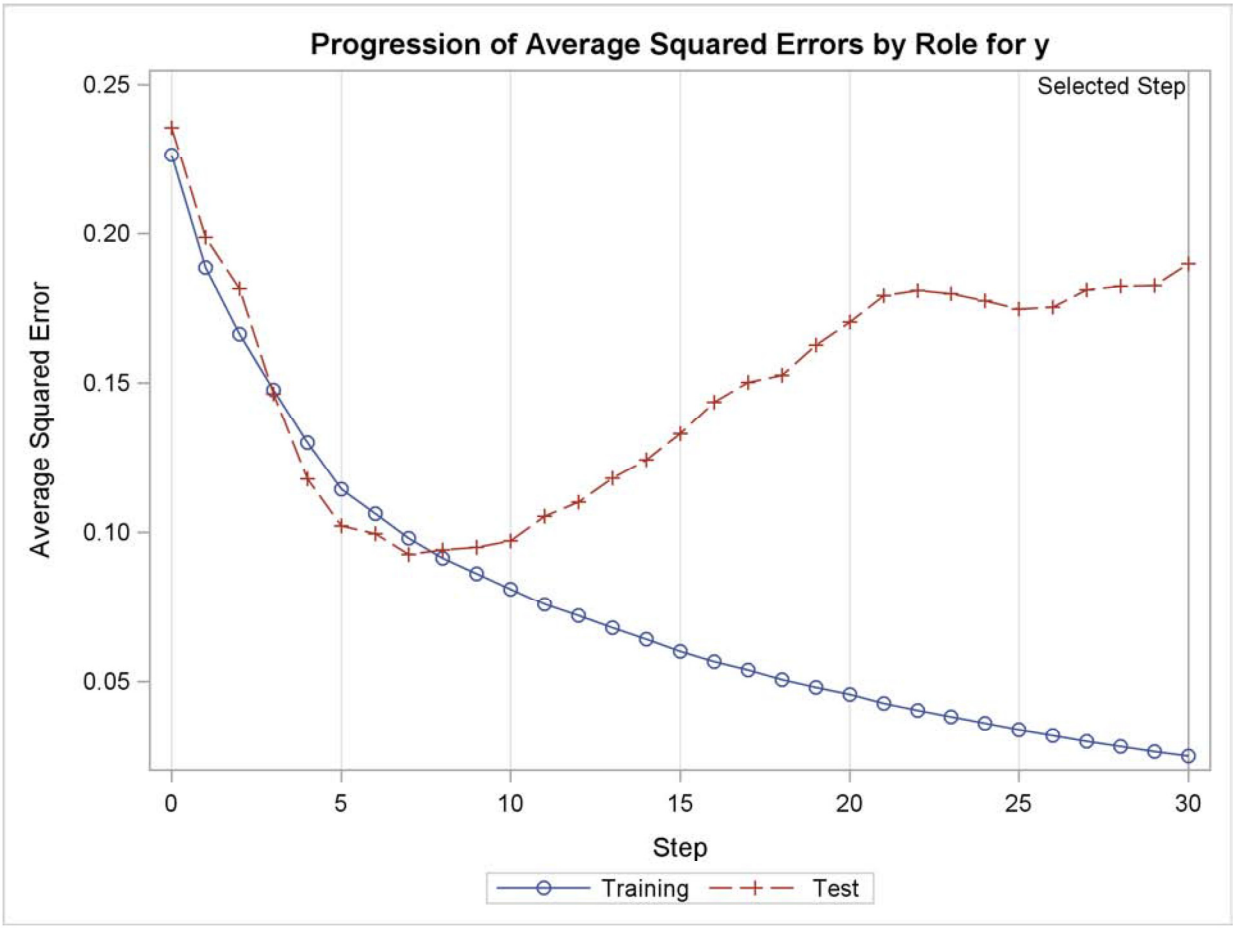
The FREQ Procedure

Train	Frequency	Percent
Correct	300	100.00

Test	Frequency	Percent
Correct	228	76.00
Incorrect	72	24.00

You can also clearly see the impact of the overfitting in the average squared error plot in Figure 21. This plot shows how the average squared error on the training and test data changes at the steps of the selection process. You see that although this error decreases monotonically on the training data, it starts growing after step 7 on the test data. Note that the test data is being used to evaluate how the models at each step of the selection process generalize to new data. You cannot use this data as part of the selection process. In studies such as the MAQC-II project, researchers are provided with only the training data, and the models they produce are evaluated by their performance on the unseen test data.

Figure 21 Average Squared Error Progression



Classification with Bootstrap-Based Model Averaging

Suppose you repeated the preceding model selection for multiple sets of training data that represent the same underlying biology. Each set of training data would yield different sets of selected regressors that each overfit their respective training data. However, the germane regressors would appear frequently in the selected models, but few of the non-predictive regressors would appear multiple times. Hence the average prediction from all these models would give greater weight to the germane regressors and thus would produce more accurate classification on new data. For real microarray experiments you do not have multiple sets of training data, but you can simulate this situation by repeatedly sampling your training data. In the bootstrap method, you generate a sample with the same number of observations as the training data by randomly drawing with replacement from the training data. Bootstrap predictions are obtained by fitting models to these bootstrap samples and averaging the predictions that you obtain.

The following code shows how you can use the SURVEYSELECT procedure to produce the bootstrap samples:

```
%let nSamples = 100;

proc surveyselect data=train out=trainBoot
    outall
    reps     = &nSamples
    sampsize = &nTrain
    method   = urs
    seed     = 1;
run;
```

The output data set trainBoot contains all the bootstrap samples of the input data, indexed by a variable named Replicate. The OUTALL option specifies that all the observations in the input data set appear in each sample. A variable named NumberHits holds the number of times that each observation is selected within a sample. The METHOD=URS option

specifies that the sampling is done with replacement.

By using Replicate as a BY variable and NumberHits as a FREQ variable, the following code selects models for all the bootstrap samples in a single invocation of the GLMSELECT procedure. The macro bootstrapAverage forms the average predictions across the BY groups and uses these average predictions for classification.

```
proc glmselect data=trainBoot;
   freq NumberHits;
   by   Replicate;

   model y=x1-x&nVars/maxsteps=20;
   output          out=trainOut(keep=y obsNum predGls)  p=predGls;
   score data=test out=testOut (keep=y obsNum predGls)  p=predGls;
   ods output parameterEstimates = pEst;
run;

%macro bootstrapAverage(source);
   proc sort data=&source.Out;
     by  obsNum;
   run;

   proc means data=&source.Out noprint;
      by obsNum;
      var predGls;
      output out=predOut(keep=pMean) mean=pMean;
   run;

   data results;
      length  &source $9;
      set &source;
      set predOut;
      if pMean<= 0.5 then yPred = 0;
                     else yPred = 1;
      if y = yPred then &source = 'Correct';
                   else &source = 'Incorrect';
   run;

   proc freq data=results;
       table &source / nocum;
   run;
%mend;

%bootstrapAverage(Train);
%bootstrapAverage(Test);
```

Figure 22 shows the classification accuracy you achieve using the average predictions from the bootstrap models. You see that although the average of the bootstrap predictions still overfits the training data, the average predictions yield classification accuracy on the test data that rivals the accuracy obtained with the optimal logistic model, which uses a priori knowledge of the germane regressors.

Figure 22 Training and Test Set Classification Errors for Bootstrap GLMSELECT Model

The FREQ Procedure

Train	Frequency	Percent
Correct	300	100.00

Test	Frequency	Percent
Correct	267	89.00
Incorrect	33	11.00

You can also form an average model whose parameter values are the averages of the parameter values in the N bootstrap models. If a parameter is not selected for a given sample, then it assigned the value 0 for that sample and it is included when forming the average parameter estimates. With this definition of the average model, the predictions

you obtain by using the average parameter estimates to score a data set are the same as the predictions you obtain by scoring this data set by using each of the bootstrap models and then averaging the predictions. To show this, denote by $\beta^{(i)}$ the parameter estimates for the bootstrap sample i where $\beta_j^{(i)} = 0$ if parameter j is not the selected model for sample i. Then the predicted values $\hat{y}^{(i)}$ for bootstrap model i are given by

$$\hat{y}^{(i)} = \mathbf{X}\beta^{(i)}$$

where **X** is the design matrix of the data to be scored. Forming averages gives

$$\hat{y}^{\text{ave}} = \frac{\sum_{i=1}^{N} \hat{y}^{(i)}}{N} = \frac{\sum_{i=1}^{N} \mathbf{X}\beta^{(i)}}{N} = \mathbf{X}\frac{\sum_{i=1}^{N} \beta^{(i)}}{N} = \mathbf{X}\beta^{\text{ave}}$$

where for parameter j, $\beta_j^{\text{ave}} = \frac{1}{N}\sum_{i=1}^{N} \beta_j^{(i)}$

You can see that if a parameter estimate is nonzero in just a few of the bootstrap models, then averaging the estimates for this parameter shrinks this estimate towards zero. It is this shrinkage that ameliorates the bias that a parameter is more likely to be selected if it is above its expected value rather than below it. This reduction in bias results in the improved predictions on new data that you obtain with the average model.

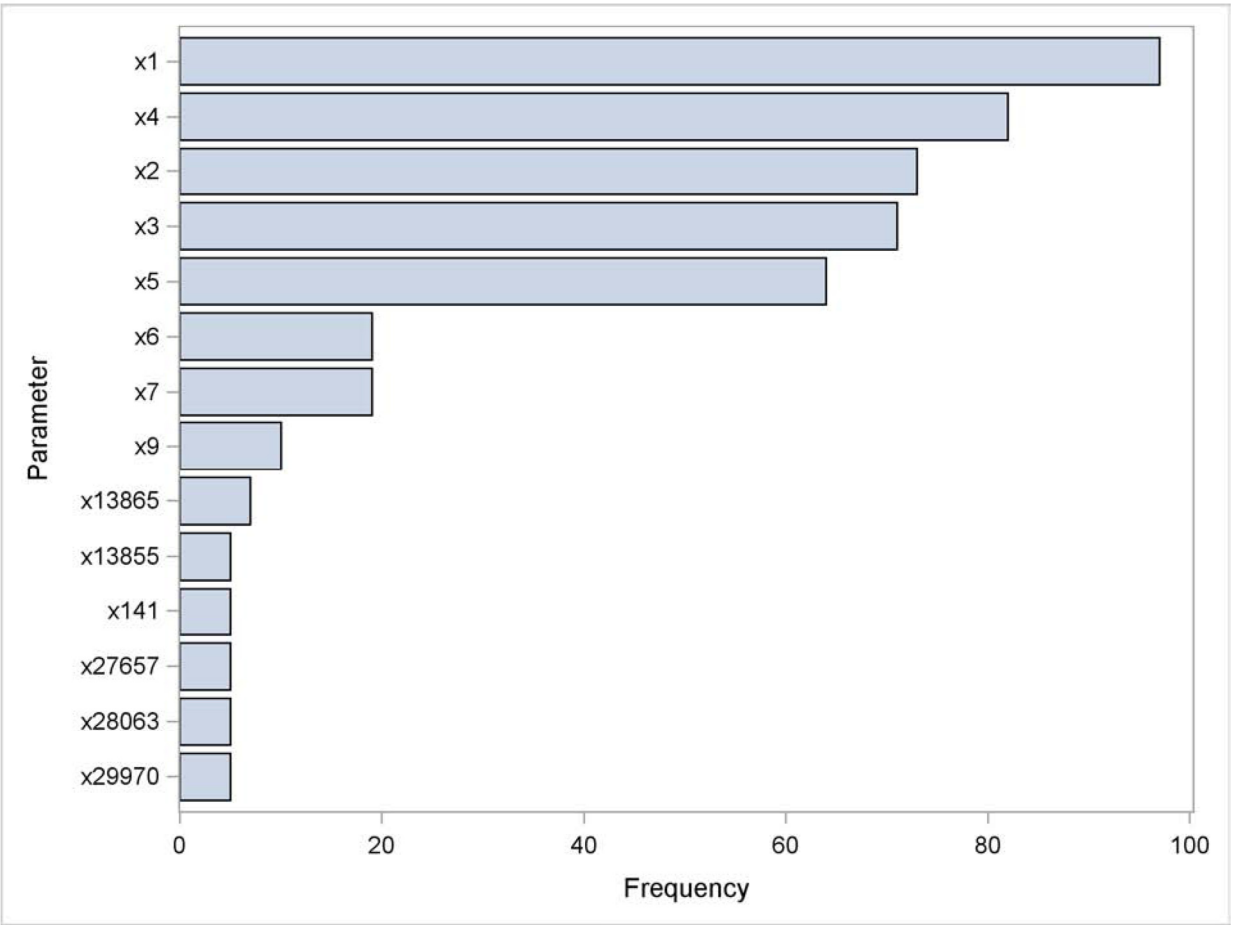
However, the average model is not parsimonious in that it can have nonzero estimates for all parameters that are selected in any bootstrap sample. In the example of this section, 1,344 of the 35,000 regressors are nonzero in the average model. However, 1,165 of these 1,344 regressors are selected exactly once and just 14 regressors are selected in at least 5% of the bootstrap models. One way to try to achieve a more parsimonious model is to average over the subset of the bootstrap models that are selected multiple times. This strategy fails completely in this example because no two bootstrap models contain identical sets of regressors. Another approach to obtaining a more parsimonious model is to ignore parameters that are selected in less than some prespecified percentage of the bootstrap models. This strategy is not reliable because although a parameter might appear in just a small percentage of models, it might be an essential parameter in those models. A simple scenario where this can occur is when two regressors are highly correlated and any appropriate model must contain one of these regressors. If just a few bootstrap models contain one of these two correlated regressors, then ignoring this parameter when forming the average model is inappropriate.

You can get a parsimonious model by using the number of times that regressors are selected as an indication of importance and then refit a new model that uses just the regressors you deem to be most important. This approach is not without its own risks. One possible problem is that you might have several regressors that for purposes of prediction can be used as surrogates for one another. In this case it possible that none of these regressors individually appears in a large enough percentage of the bootstrap models to be deemed important, even though every model contains at least one of these regressors. Despite such potential problems, this strategy is often successful.

The following code uses the output data set that contains the parameter estimates from the bootstrap models to identify the most frequently selected regressors, which are displayed in the bar chart in Figure 23:

```
proc sort data=pEst;by parameter;run;

proc freq data=pEst order=freq noprint;
  tables parameter/nocum out=freqOut(drop=percent);
run;

data freqOut;
  set freqOut;
  rename count=nTimesSelected;
  PercentSelected=100*count/&nSamples;
run;

proc sgplot data=freqout(where=(nTimesSelected>=5 and
                                Parameter ^= "Intercept"));
  yaxis discreteOrder = data;
  hbar Parameter/freq=nTimesSelected;
run;
```

Figure 23 Regressors Selected At Least Five Times



You can now use this reduced set of regressors for further exploratory data analysis that would be intractable using all the original regressors. For example, you might try to see whether interactions among these regressors are important, and you can investigate whether nonlinear transformations of these regressors might be appropriate. As with the original variable selection, you need to be extremely careful that by performing these additional steps you are not simply overfitting the training data.

Classification by Using Stepwise Selection with Validation

Another way to get a parsimonious model directly with PROC GLMSELECT is to mimic the use of a test data set to stop the model selection before overfitting the training data starts to occur. Although you cannot use the test data during the selection process, provided that you have a sufficient number of training observations, you can split the input data into smaller training and validation subsets and use the validation data as a surrogate for the test data. You can use a PARTITION statement to do this as follows:

```
ods graphics on;

proc glmselect data=train plots=ase testdata=test seed=1;
   partition fraction(validate=0.3333);

   model y = x1-x&nVars / selection=stepwise(choose=validate maxsteps=30);
   output          out=trainOut(keep=y predGls)  p=predGls;
   score data=test out=testOut (keep=y predGls)  p=predGls;
run;

ods graphics off;

%classify(trainOut,Train,predGls);
%classify(testOut,Test,predGls);
```

The PARTITION statement specifies that about 100 (1/3 of 300) randomly selected observations from the input data set are reserved as validation data and that the remaining 200 observations are used as the training data. The CHOOSE=VALIDATE option specifies that the selected model is the model at the step that yields the lowest average error sum of squares on the validation data.

Figure 24 shows the selected effects and Figure 25 shows classification errors on the training and test data sets. You can see that this model underfits the training data but still yields good predictions on the test data.

Figure 24 Selected Effects

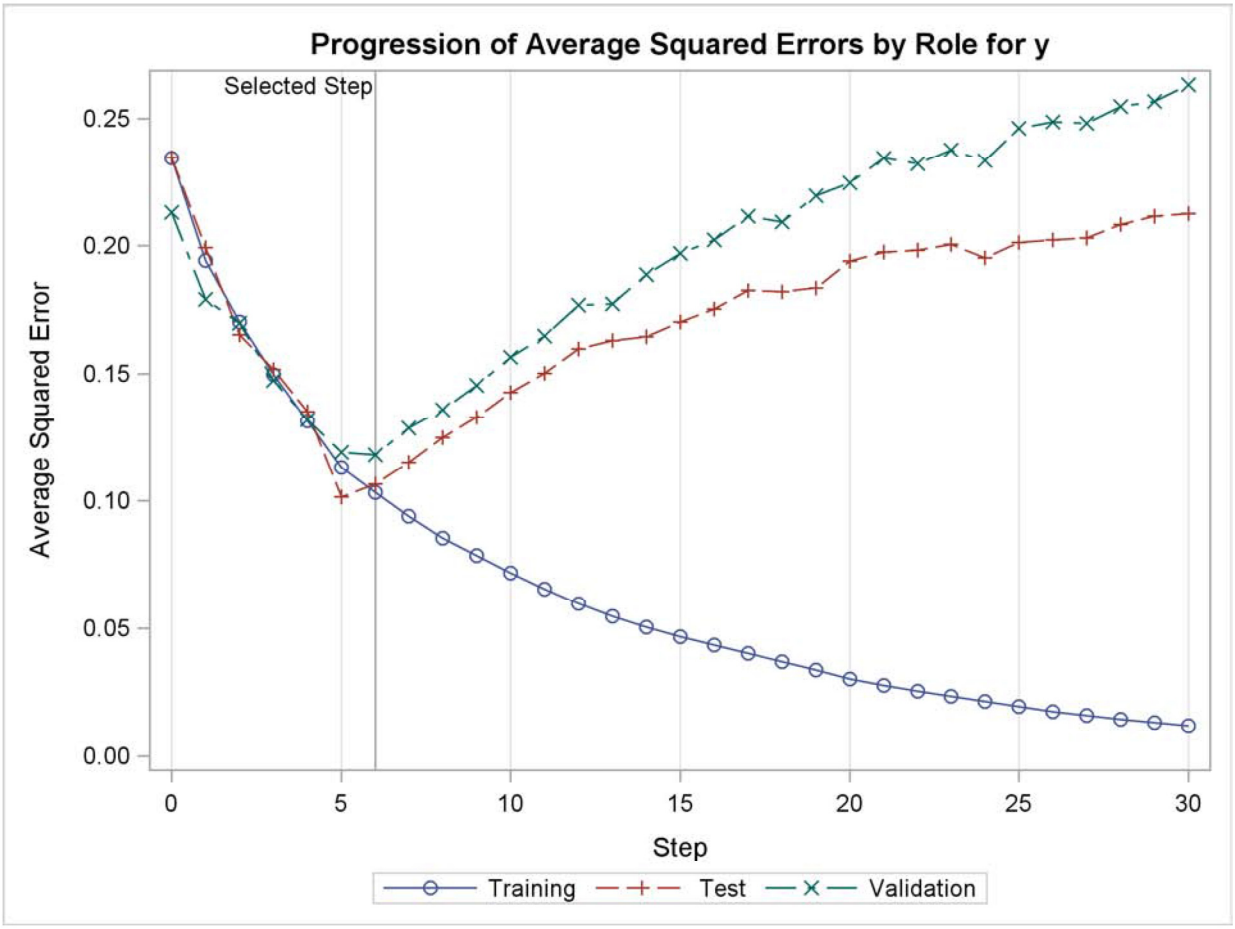
```
The GLMSELECT Procedure
     Selected Model

Effects: Intercept x1 x2 x3 x4 x5 x9
```

Figure 25 Training and Test Set Classification Errors for GLMSELECT Model with Validation

```
The FREQ Procedure

Train         Frequency     Percent
-----------------------------------
Correct             265       88.33
Incorrect            35       11.67

Test          Frequency     Percent
-----------------------------------
Correct             270       90.00
Incorrect            30       10.00
```

Figure 26 compares the progression of the average squared errors on the training, validation, and test data. You see that the average squared error on the validation data behaves similarly to the average squared error on the test data and prevents the overfitting of the training data.

Figure 26 Average Squared Error Progression



You might also consider using a bootstrap together with a randomly chosen validation data set with each bootstrap sample. However, as the preceding results have shown, you can expect most such bootstrap models to underfit the training data and omit some important regressors. Because the average predictions of the bootstrap models correspond to an average model where the estimates of infrequently selected parameters are shrunk, the bootstrap predictions will amplify the effect of underfitting the training data. One approach to address this is to form an average model where nonselected parameters are not counted when forming the average estimates. See Burnham and Anderson (2002) for details about this strategy.

FUTURE WORK

You have seen how applying bootstrap analysis can be used to address the problem of model selection bias that arises when a single model is produced by using variable selection. In order to facilitate such bootstrap analyses, future releases of the GLMSELECT procedure will include a BOOTSTRAP statement that will enable you to perform bootstrap analysis and obtain appropriately averaged models as a built-in feature. For larger problems, bootstrap analyses are computationally demanding in that the same model selection process needs to be repeated hundreds or even thousands of times. Because the model selection on each bootstrap sample is done independently, multiple samples can be processed in parallel. To exploit this high degree of parallelism, PROC GLMSELECT will support distributed grid processing of the bootstrap analyses.

CONCLUSIONS

This paper demonstrates how you can use the GLMSELECT procedure to perform model selection where the number of regressors is large. Microarray experiments provide one important class of such problems. Other examples include nonparametric fitting by selecting from a large set of suitable basis functions. The paper also highlights the use of the new experimental EFFECT statement to generate spline basis functions, and it demonstrates how you can perform bootstrap analyses to produce models that address problems that arise with single model selection.

REFERENCES

Burnham, K. P. and Anderson, D. R. (2002), *Model Selection and Multimodel Inference,* Second Edition, New York: Springer-Verlag.

Cohen, R, (2006), "Introducing the GLMSELECT Procedure for Model Selection," *Proceedings of the Thirty-First Annual SAS Users Group International Conference.*

Donoho, D. L. and Johnstone, I. M. (1994), "Ideal Spatial Adaptation via Wavelet Shrinkage," *Biometrika*, 81, 425–455.

Efron, B., Hastie, T., Johnstone, I., and Tibshirani, R. (2004), "Least Angle Regression (with Discussion)," *Annals of Statistics*, 32, 407–499.

Eilers, P. H. C. and Marx, B. D. (1996), "Flexible Smoothing with B-Splines and Penalties (with Discussion)," *Statistical Science*, 11, 89–121.

Foster, D. P. and Stine, R. A. (2004), "Variable Selection in Data Mining: Building a Predictive Model for Bankruptcy," *Journal of the American Statistical Association*, 99, 303–313.

Harrell, F. E. (2001), *Regression Modeling Strategies,* New York: Springer-Verlag.

Hastie, T., Tibshirani, R., and Friedman, J. (2001), *The Elements of Statistical Learning,* New York: Springer-Verlag.

Miller, A. (2002), *Subset Selection in Regression,* Second Edition, New York: Chapman & Hall/CRC.

Raftery, A. E., Madigan, D., and Hoeting, J. A. (1997), "Bayesian Model Averaging for Linear Regression Models," *Journal of the American Statistical Association*, 92, 179–191.

Sarle, W. S. (2001) "Donoho-Johnstone Benchmarks: Neural Net Results," ftp://ftp.sas.com/pub/neural/dojo/dojo.html: last accessed January 6, 2009.

Tibshirani, R. (1996), "Regression Shrinkage and Selection via the Lasso," *Journal of the Royal Statistical Society Series B*, 58, 267–288.

Wolfinger, R. (2008), *Private Communication*.

Paper SAS1742-2015

Introducing the HPGENSELECT Procedure: Model Selection for Generalized Linear Models and More

Gordon Johnston and Robert N. Rodriguez, SAS Institute Inc.

Abstract

Generalized linear models are highly useful statistical tools in a broad array of business applications and scientific fields. How can you select a good model when numerous models that have different regression effects are possible? The HPGENSELECT procedure, which was introduced in SAS/STAT® 12.3, provides forward, backward, and stepwise model selection for generalized linear models. In SAS/STAT 14.1, the HPGENSELECT procedure also provides the LASSO method for model selection. You can specify common distributions in the family of generalized linear models, such as the Poisson, binomial, and multinomial distributions. You can also specify the Tweedie distribution, which is important in ratemaking by the insurance industry and in scientific applications.

You can run the HPGENSELECT procedure in single-machine mode on the server where SAS/STAT is installed. With a separate license for SAS® High-Performance Statistics, you can also run the procedure in distributed mode on a cluster of machines that distribute the data and the computations.

This paper shows you how to use the HPGENSELECT procedure both for model selection and for fitting a single model. The paper also explains the differences between the HPGENSELECT procedure and the GENMOD procedure.

Introduction

Generalized linear models are highly versatile statistical models that have a huge range of applications. For example, these models are used in the insurance industry to set rates, in the airline industry to reduce the frequency of flight delays, and in health care to find relationships between cancer incidence and possible causes.

What makes these models so versatile? Generalized linear models accommodate response variables that follow many different distributions, including the normal, binomial, Poisson, gamma, and Tweedie distribution. Like other linear models, generalized linear models use a linear predictor. They also involve a link function, which transforms the mean of the response variable to the scale of the linear predictor.

The HPGENSELECT procedure was introduced in SAS/STAT 12.3 in July 2013. Like the GENMOD procedure, the HPGENSELECT procedure uses maximum likelihood to fit generalized linear models. In addition, PROC HPGENSELECT provides variable selection (including forward, backward, stepwise, and LASSO selection methods) for building models, and it supports standard distributions and link functions. It also provides specialized models for zero-inflated count data, ordinal data, and unordered multinomial data.

PROC HPGENSELECT is a high-performance analytical procedure, which means that you can run it in two ways:

- You can run the procedure in single-machine mode on the server where SAS/STAT is installed, just as you can with other SAS/STAT procedures. No additional license is required.
- You can run the procedure in distributed mode on a cluster of machines that distribute the data and the computations. Because each node in the cluster does a slice of the work, PROC HPGENSELECT exploits the computing power of the cluster to fit large models to massive amounts of data. To run in distributed mode, you need to license SAS High-Performance Statistics.

Comparing the HPGENSELECT and GENMOD Procedures

Like the GENMOD procedure, the HPGENSELECT procedure uses maximum likelihood to fit generalized linear models. Whereas the GENMOD procedure offers a rich set of methods for statistical inference such as Bayesian analysis and postfit analysis, the HPGENSELECT procedure is designed for predictive modeling and other large-data tasks. In addition, PROC HPGENSELECT enables you to do variable selection for generalized linear models, which is new in SAS/STAT. You can run PROC HPGENSELECT in single-machine mode and exploit all the cores on your computer. And as the size of your problems grows, you can take full advantage of all the cores and large memory in distributed computing environments.

Building Generalized Linear Models with the HPGENSELECT Procedure

In order to fit a generalized linear model, you specify a response distribution that is appropriate for your data, a set of independent variables (covariates), and a link function that transforms the linear predictor to the scale of the response. Covariates can be either continuous variables or classification variables, or they can be effects that involve two or more variables.

Table 1 shows the response distributions that PROC HPGENSELECT provides.

Table 1 Response Probability Distributions from PROC HPGENSELECT

Distribution	Default Link Function	Appropriate Response Data Type
Binary	Logit	Binary
Binomial	Logit	Binomial events/trials
Gamma	Inverse	Continuous, positive
Inverse Gaussian	Inverse square	Continuous, positive
Multinomial with generalized logit link function		Nominal categorical
Multinomial	Logit	Ordered categorical
Negative binomial	Log	Count
Gaussian	Identity	Continuous
Poisson	Log	Count
Tweedie	Log	Continuous or mixed discrete and continuous
Zero-inflated negative binomial	Log/logit	Count with zero-inflation probability
Zero-inflated Poisson	Log/logit	Count with zero-inflation probability

Examples

The following examples illustrate key features of the HPGENSELECT procedure.

Fitting a Poisson Model to Auto Insurance Data

This example uses an automobile insurance data set called OntarioAuto, which has about 500,000 observations. The data set contains a response variable, **NumberOfClaims**, which represents the number of claims that an individual policyholder submits in a certain time period. The log transform of its mean depends on the continuous regressors **PolicyAge**, **DriverAge**, and **LicenseAge** and on four classification regressors, **MultiVehicle**, **Gender**, **RatingGroup**, and **TransactType**. The logarithm of an exposure variable, **logExposure**, is used as an offset variable to normalize the number of claims to the same time period. The following statements use the HPGENSELECT procedure, running in single-machine mode, to fit a Poisson regression model that has all the variables:

```
libname Data 'C:\Data';
proc HPGenselect data=Data.OntarioAuto;
   class Gender RatingGroup MultiVehicle TransactType;
   model NumberOfClaims=MultiVehicle Gender RatingGroup
                        TransactType PolicyAge DriverAge
                        LicenseAge / dist=Poisson
                                     link=Log CL
                                     offset=logExposure;
   performance details;
   code file = 'AutoScore.txt';
run;
```

The LIBNAME statement specifies data that in this example happen to be saved locally on the computer on which SAS® is running. The CLASS statement identifies the classification variables in the model, and the MODEL statement specifies the response variable, the regression variables, and options such as the distribution, the link function, and the offset variable. The CL option requests that confidence limits for all model parameters be displayed.

The PERFORMANCE statement requests that procedure execution times be displayed. The CODE statement produces a text file named AutoScore.txt that can be used for scoring. This file contains fitted model information that can be included in a DATA step for scoring, as shown on page 4.

The procedure output in Figure 1 provides the settings that are used in this analysis. The "Performance Information" table shows that PROC HPGENSELECT executed in single-machine mode on four concurrent threads, which is the number of CPUs on the machine. The "Model Information" table shows model information, such as the distribution and link function that were used. The "Number of Observations" table shows the number of observations that were read and the number that were used in the analysis. More observations were read than were used in the analysis because some observations had missing values for either the response or regression variables.

Figure 1 Model Settings

Performance Information	
Execution Mode	Single-Machine
Number of Threads	4

Model Information	
Data Source	DATA.ONTARIOAUTO
Response Variable	NumberOfClaims
Offset Variable	logexposure
Class Parameterization	GLM
Distribution	Poisson
Link Function	Log
Optimization Technique	Newton-Raphson with Ridging

Number of Observations Read	567962
Number of Observations Used	386729

Figure 2 shows the levels of the classification variables that were listed in the CLASS statement, important fit statistics such as Akaike's information criterion (AIC), and the resulting parameter estimates, confidence limits, and standard errors.

Figure 2 Model Fit Results

Class Level Information		
Class	**Levels**	**Values**
Gender	2	F M
RatingGroup	12	02 05 08 11 14 17 20 23 26 29 30 31
MultiVehicle	2	Multi Single
TransactType	3	MOD NEW REN

Fit Statistics	
-2 Log Likelihood	29822
AIC (smaller is better)	29860
AICC (smaller is better)	29860
BIC (smaller is better)	30066
Pearson Chi-Square	517848
Pearson Chi-Square/DF	1.3391

Figure 2 *continued*

Parameter Estimates							
Parameter	DF	Estimate	Standard Error	95% Confidence Limits		Chi-Square	Pr > ChiSq
Intercept	1	-3.672780	0.196874	-4.05865	-3.28691	348.0274	<.0001
MultiVehicle Multi	1	-0.289906	0.041559	-0.37136	-0.20845	48.6613	<.0001
MultiVehicle Single	0	0	.	.	.	.	.
Gender F	1	0.051268	0.040167	-0.02746	0.12999	1.6291	0.2018
Gender M	0	0	.	.	.	.	.
RatingGroup 02	1	-0.770014	0.295287	-1.34877	-0.19126	6.8000	0.0091
RatingGroup 05	1	-0.157194	0.197180	-0.54366	0.22927	0.6355	0.4253
RatingGroup 08	1	-0.045116	0.193078	-0.42354	0.33331	0.0546	0.8152
RatingGroup 11	1	0.077805	0.189152	-0.29293	0.44854	0.1692	0.6808
RatingGroup 14	1	0.120489	0.185472	-0.24303	0.48401	0.4220	0.5159
RatingGroup 17	1	0.116955	0.184574	-0.24480	0.47871	0.4015	0.5263
RatingGroup 20	1	0.258596	0.184863	-0.10373	0.62092	1.9568	0.1619
RatingGroup 23	1	0.228393	0.184471	-0.13316	0.58995	1.5329	0.2157
RatingGroup 26	1	0.294483	0.186638	-0.07132	0.66029	2.4896	0.1146
RatingGroup 29	1	0.213461	0.191124	-0.16113	0.58806	1.2474	0.2640
RatingGroup 30	1	-0.014585	0.289647	-0.58228	0.55311	0.0025	0.9598
RatingGroup 31	0	0	.	.	.	.	.
TransactType MOD	1	0.262652	0.043490	0.17741	0.34789	36.4745	<.0001
TransactType NEW	1	0.139413	0.082705	-0.02269	0.30151	2.8415	0.0919
TransactType REN	0	0	.	.	.	.	.
PolicyAge	1	-0.005330	0.003386	-0.01197	0.00131	2.4770	0.1155
DriverAge	1	-0.000102	0.001619	-0.00327	0.00307	0.0040	0.9496
LicenseAge	1	-0.013132	0.002377	-0.01779	-0.00847	30.5271	<.0001

The timing table in Figure 3 shows that the procedure took a little more than two seconds to run.

Figure 3 Timing

Procedure Task Timing		
Task	Seconds	Percent
Reading and Levelizing Data	0.41	18.46%
Full model fit	1.81	81.54%

The following DATA step statements score the first 100 observations of the original data set by using the fitted model information in AutoScore.txt. The variable **P_NumberOfClaims** represents predicted values in the scored data set ScoreData.

```
data ScoreData;
   keep P_NumberOfClaims NumberOfClaims MultiVehicle Gender
        RatingGroup  TransactType PolicyAge DriverAge
        LicenseAge Exposure;
   set Data.OntarioAuto(obs=100);
   %inc 'AutoScore.txt';
run;
```

Figure 4 shows the first 10 observations in the scored data set. You can score any data set by using this method; the only requirement is that all the regression variables and the offset variable that are in the original model be present.

Figure 4 Scoring Data

Obs	NumberOfClaims	LicenseAge	PolicyAge	DriverAge	Exposure	TransactType	MultiVehicle
1	0	28	9.0	44	0.10685	MOD	Multi
2	0	23	10.0	63	0.10137	REN	Multi
3	0	24	7.0	45	0.00000	MOD	Multi
4	0	41	8.0	58	0.04110	MOD	Multi
5	0	29	20.5	47	0.00000	MOD	Multi
6	0	21	11.0	74	0.32329	MOD	Multi
7	0	19	7.0	76	0.00000	MOD	Multi
8	0	6	6.0	22	0.18904	MOD	Multi
9	0	49	6.0	69	0.01918	MOD	Multi
10	0	19	5.0	67	0.90959	MOD	Multi

Obs	RatingGroup	Gender	P_NumberOfClaims
1	17	M	0.001951
2	26	M	0.001802
3	17	F	.
4	17	M	0.000635
5	31	M	.
6	17	F	0.006718
7	02	M	.
8	17	M	0.004692
9	14	F	0.000284
10	29	M	0.020977

Fitting a Tweedie Model to Auto Insurance Data

Now, suppose you want to fit a model for the cost of claims instead of the number of claims. The OntarioAuto data set contains the variable **DollarClaims**, which represents the cost of an individual policyholder's claims over a period of time. Many observations have a value of 0 for **DollarClaims** because there were no claims for those observations. However, for observations that have nonzero cost, a continuous distribution is appropriate. The Tweedie distribution is sometimes used for this type of data because it can model continuous data that have a discrete component at 0.

The following statements use the HPGENSELECT procedure, running in single-machine mode, to fit a Tweedie regression model for **DollarClaims** by using the same regressors as in the previous example:

```
libname Data 'C:\Data';
proc HPGenselect data=Data.OntarioAuto;
   class Gender RatingGroup MultiVehicle TransactType;
   model DollarClaims=MultiVehicle Gender RatingGroup
                      TransactType PolicyAge DriverAge
                      LicenseAge / dist=Tweedie
                                   link=Log CL
                                   offset=logExposure;
   performance details;
run;
```

The “Model Information” table in Figure 5 shows the Tweedie model settings.

Figure 5 Model Information

Model Information	
Data Source	DATA.ONTARIOAUTO
Response Variable	DollarClaims
Offset Variable	logexposure
Class Parameterization	GLM
Distribution	Tweedie
Link Function	Log
Optimization Technique	Quasi-Newton

Figure 6 shows the resulting Tweedie model fit statistics and parameter estimates.

Figure 6 Fit Statistics

Fit Statistics	
-2 Log Likelihood	77912
AIC (smaller is better)	77954
AICC (smaller is better)	77954
BIC (smaller is better)	78183
Pearson Chi-Square	1.4562E9
Pearson Chi-Square/DF	3765.71

Parameter Estimates

Parameter	DF	Estimate	Standard Error	95% Confidence Limits		Chi-Square	Pr > ChiSq
Intercept	1	5.231829	0.272081	4.69856	5.76510	369.7533	<.0001
MultiVehicle Multi	1	-0.351287	0.063940	-0.47661	-0.22597	30.1840	<.0001
MultiVehicle Single	0	0	.	.	.	.	.
Gender F	1	0.069925	0.060776	-0.04919	0.18904	1.3237	0.2499
Gender M	0	0	.	.	.	.	.
RatingGroup 02	1	-2.128568	0.405827	-2.92397	-1.33316	27.5102	<.0001
RatingGroup 05	1	-1.161764	0.273965	-1.69873	-0.62480	17.9823	<.0001
RatingGroup 08	1	-1.079393	0.269130	-1.60688	-0.55191	16.0855	<.0001
RatingGroup 11	1	-0.654231	0.260553	-1.16491	-0.14356	6.3048	0.0120
RatingGroup 14	1	-0.261659	0.251862	-0.75530	0.23198	1.0793	0.2989
RatingGroup 17	1	-0.122954	0.249669	-0.61230	0.36639	0.2425	0.6224
RatingGroup 20	1	-0.039529	0.251577	-0.53261	0.45355	0.0247	0.8751
RatingGroup 23	1	0.215756	0.249169	-0.27261	0.70412	0.7498	0.3865
RatingGroup 26	1	0.175484	0.253458	-0.32129	0.67225	0.4794	0.4887
RatingGroup 29	1	0.422019	0.256827	-0.08135	0.92539	2.7001	0.1003
RatingGroup 30	1	-0.512928	0.417520	-1.33125	0.30540	1.5092	0.2193
RatingGroup 31	0	0	.	.	.	.	.
TransactType MOD	1	0.336215	0.064126	0.21053	0.46190	27.4891	<.0001
TransactType NEW	1	0.061979	0.134865	-0.20235	0.32631	0.2112	0.6458
TransactType REN	0	0	.	.	.	.	.
PolicyAge	1	-0.009993	0.004904	-0.01960	-0.00038175	4.1527	0.0416
DriverAge	1	0.001526	0.002467	-0.00331	0.00636	0.3823	0.5364
LicenseAge	1	-0.009299	0.003383	-0.01593	-0.00267	7.5556	0.0060
Dispersion	1	1579.296618	25.892051	1529.35580	1630.86824	.	.
Power	1	1.562776	0.005967	1.55113	1.57451	.	.

The parameters **Intercept** through **LicenseAge** are regression parameters, and **Dispersion** and **Power** are Tweedie dispersion and power parameters, respectively.

The timing table in Figure 7 shows that PROC HPGENSELECT took slightly more than 1.5 minutes to run. This is considerably more time than the Poisson model took, because the Tweedie likelihood takes more resources to compute than the Poisson likelihood.

Figure 7 Timing

Procedure Task Timing

Task	Seconds	Percent
Reading and Levelizing Data	0.36	0.38%
Full model fit	94.57	99.62%

Model Selection with a Zero-Inflated Model

The examples in this section use a simulated data set named GLMData, which has 10 million observations. These data contain a response variable named **yZIP**, which is constructed to have a zero-inflated Poisson (ZIP) distribution. The response variable **yZIP** depends on a number of regression variables that are listed in Table 2. In addition to including these regression variables, the data set contains a number of noise variables that are unrelated to **yZIP**.

Table 2 Regressors for ZIP Model

Regressor Name	Type	Number of Levels	Role
xIn1–xIn20	Continuous		Regressor for Poisson mean
xSubtle	Continuous		Regressor for Poisson mean
xTiny	Continuous		Regressor for Poisson mean
xOut1–xOut80	Continuous		Noise
cIn1–cIn5	Classification	2–5	Regressor for Poisson mean and zero-inflation probability
cOut1–cOut5	Classification	2–5	Noise

The Poisson mean part of the ZIP model depends on the variables **xIn1–xIn20** and **cIn1–cIn5** through a logarithmic link. It also depends on the variables **xTiny** and **xSubtle**, but the dependence is considerably weaker. The zero-inflation probability depends on the classification variables **cIn1–cIn5** through a logit link function. The variables **xOut1–xOut80** and **cOut1–cOut5** are noise variables that are included in the model selection process but do not influence the response. A model selection procedure should screen out these variables as being unimportant to the model.

The following statements fit a zero-inflated model that uses **yZIP** as the response and all the variables in Table 2 as regressors. The HPGENSELECT procedure runs in single-machine mode in this example and uses only the first 50,000 observations from the data set GLMData to perform stepwise model selection. As in the preceding example, the data are saved locally on the computer on which SAS is running.

```
libname Data 'C:\Data';
proc hpgenselect data=Data.GLMData(obs=50000);
   class c:;
   model yZIP = x: c: / dist=ZIP;
   zeromodel c:;
   selection method=stepwise(choose=sbc);
   performance details;
run;
```

The MODEL statement specifies the model for the Poisson mean part of the model, and the ZEROMODEL statement specifies the model for the zero-inflation probability. The symbols **x:** and **c:** are shorthand for all variables that begin with **x** and **c**, respectively. The SELECTION statement requests that the stepwise selection method be used and that the final model be chosen on the basis of the best Schwarz Bayesian criterion (SBC).

The "Performance Information" table in Figure 8 shows that PROC HPGENSELECT ran in single-machine mode on four concurrent threads. The "Model Information" table shows model settings for the zero-inflated model. The "Number of Observations" table shows that 50,000 observations were used in the analysis.

Figure 8 Performance Information

Performance Information	
Execution Mode	Single-Machine
Number of Threads	4

Model Information	
Data Source	DATA.GLMDATA
Response Variable	yZIP
Class Parameterization	GLM
Distribution	Zero-Inflated Poisson
Link Function	Log
Zero Model Link Function	Logit
Optimization Technique	Newton-Raphson with Ridging

Number of Observations Read	50000
Number of Observations Used	50000

The “Selection Summary” table in Figure 9 shows that the model in step 30 was selected on the basis of the minimum SBC. None of the noise variables were selected. However, **xSubtle** and **xTiny** were not included in the model. Would performing model selection with more data provide a better model by including these variables?

Figure 9 Selection Summary

Selection Summary

Step	Effect Entered	Number Effects In	SBC	p Value
0	Intercept	1		
	Intercept_Zero	2	146817.850	.
1	cIn5_Zero	3	131452.827	<.0001
2	cIn4_Zero	4	119946.851	<.0001
3	cIn3_Zero	5	114440.541	<.0001
4	xIn20	6	110924.857	<.0001
5	xIn19	7	107695.553	<.0001
6	xIn18	8	104609.925	<.0001
7	xIn17	9	101844.442	<.0001
8	xIn16	10	99349.952	<.0001
9	xIn15	11	96947.036	<.0001
10	cIn2_Zero	12	94827.139	<.0001
11	xIn14	13	92923.729	<.0001
12	xIn13	14	91186.990	<.0001
13	xIn12	15	89697.891	<.0001
14	xIn11	16	88390.109	<.0001
15	xIn10	17	87236.680	<.0001
16	cIn4	18	86337.206	<.0001
17	cIn5	19	84618.512	<.0001
18	cIn3	20	83550.746	<.0001
19	xIn9	21	82695.565	<.0001
20	xIn8	22	81912.081	<.0001
21	xIn7	23	81311.094	<.0001
22	xIn6	24	80875.454	<.0001
23	cIn2	25	80561.009	<.0001
24	cIn1_Zero	26	80335.362	<.0001
25	xIn5	27	80119.301	<.0001
26	xIn4	28	79921.678	<.0001
27	xIn3	29	79810.099	<.0001
28	xIn2	30	79786.064	<.0001
29	cIn1	31	79774.564	<.0001
30	xIn1	32	79770.630*	0.0001
31	xOut53	33	79771.580	0.0017
32	xOut14	34	79775.166	0.0072
33	xOut51	35	79780.299	0.0171
34	xOut56	36	79785.855	0.0218
35	xOut33	37	79792.191	0.0342
36	cOut2	38	79807.117	0.0346
37	xOut5	39	79813.927	0.0452
38	cOut5	40	79847.492	0.0459

*** Optimal Value of Criterion**

Selected Effects:	Intercept xIn1 xIn2 xIn3 xIn4 xIn5 xIn6 xIn7 xIn8 xIn9 xIn10 xIn11 xIn12 xIn13 xIn14 xIn15 xIn16 xIn17 xIn18 xIn19 xIn20 cIn1 cIn2 cIn3 cIn4 cIn5 Intercept_Zero cIn1_Zero cIn2_Zero cIn3_Zero cIn4_Zero cIn5_Zero

The timing table in Figure 10 shows that the procedure took about 70 seconds to run in single-machine mode.

Figure 10 Timing

Procedure Task Timing		
Task	**Seconds**	**Percent**
Reading and Levelizing Data	0.41	0.60%
Candidate evaluation	27.88	40.92%
Candidate model fit	38.38	56.32%
Final model fit	1.47	2.16%

Performing the analysis by using the full data set of 10 million observations would take several hours in single-machine mode. You can use distributed mode to do the same analysis in far less time. The entire data set of 10 million observations was loaded on a Hadoop server. The following statements read the data from the Hadoop server and perform the computations in distributed mode on a different server that contains 10 server nodes:

```
option set=SAS_HADOOP_JAR_PATH='C:\hadoop\cloudera';
option set=GRIDHOST='bigmath.unx.sas.com';
option set=GRIDINSTALLLOC='/opt/TKGrid';
option set=GRIDMODE='asym';

libname gridlib HADOOP
   server="hpa.sas.com"
   user=XXXXXX
   HDFS_TEMPDIR="temp"
   HDFS_PERMDIR="perm"
   HDFS_METADIR="meta"
   config="demo.xml"
   DBCREATE_TABLE_EXTERNAL=NO;

proc hpgenselect data=gridlib.GLMData;
   class c:;
   model yZIP = x: c: / dist=ZIP;
   zeromodel c:;
   selection method=stepwise(choose=sbc);
   performance details nodes=10;
run;
```

The “Performance Information” table in Figure 11 shows that the analysis was performed in distributed mode by using 10 computing nodes, each with 32 threads. The table also shows that PROC HPGENSELECT ran in asymmetric mode, where the computations are performed in a distributed computing environment that is separate from the database where the data are stored. The “Model Information” table shows the same model as in the previous analysis. The “Number of Observations” table shows that all 10 million observations were used.

Figure 11 Performance Information

Performance Information	
Host Node	bigmath.unx.sas.com
Execution Mode	Distributed
Grid Mode	Asymmetric
Number of Compute Nodes	10
Number of Threads per Node	32

Model Information	
Data Source	GRIDLIB.GLMDATA
Response Variable	yZIP
Class Parameterization	GLM
Distribution	Zero-Inflated Poisson
Link Function	Log
Zero Model Link Function	Logit
Optimization Technique	Newton-Raphson with Ridging

Number of Observations Read	10000000
Number of Observations Used	10000000

The “Selection Summary” table in Figure 12 shows that this analysis included all the variables that are included in the model in the previous analysis. In addition, the variable **xSubtle** is included, reflecting the larger amount of data.

Figure 12 Selection Summary

Selection Summary

Step	Effect Entered	Number Effects In	SBC	p Value
0	Intercept	1		
	Intercept_Zero	2	29524449.3	.
1	cIn5_Zero	3	26540103.3	<.0001
2	cIn4_Zero	4	24211661.5	<.0001
3	cIn3_Zero	5	23058742.7	<.0001
4	xIn20	6	22360427.5	<.0001
5	xIn19	7	21704302.2	<.0001
6	xIn18	8	21092219.0	<.0001
7	xIn17	9	20525192.6	<.0001
8	xIn16	10	20015823.1	<.0001
9	xIn15	11	19555603.2	<.0001
10	cIn2_Zero	12	19106403.6	<.0001
11	xIn14	13	18693647.9	<.0001
12	xIn13	14	18333245.6	<.0001
13	xIn12	15	18021684.7	<.0001
14	xIn11	16	17759850.3	<.0001
15	xIn10	17	17541040.8	<.0001
16	cIn5	18	17347544.1	<.0001
17	cIn4	19	16995861.2	<.0001
18	cIn3	20	16766910.6	<.0001
19	xIn9	21	16580877.7	<.0001
20	xIn8	22	16433007.4	<.0001
21	xIn7	23	16318265.8	<.0001
22	xIn6	24	16234037.8	<.0001
23	cIn2	25	16164935.0	<.0001
24	xIn5	26	16106958.3	<.0001
25	cIn1_Zero	27	16053016.7	<.0001
26	xIn4	28	16015246.4	<.0001
27	xIn3	29	15994179.6	<.0001
28	xIn2	30	15984996.4	<.0001
29	cIn1	31	15978262.2	<.0001
30	xIn1	32	15975845.3	<.0001
31	xSubtle	33	15975836.8*	<.0001
32	xTiny	34	15975843.1	0.0017
33	xOut52	35	15975852.6	0.0104
34	xOut22	36	15975862.8	0.0149
35	xOut28	37	15975873.2	0.0164
36	xOut18	38	15975884.7	0.0328
37	xOut73	39	15975896.3	0.0336
38	cOut1	40	15975908.3	0.0420
39	xOut2	41	15975920.4	0.0450

*** Optimal Value of Criterion**

Selected Effects:	Intercept xIn1 xIn2 xIn3 xIn4 xIn5 xIn6 xIn7 xIn8 xIn9 xIn10 xIn11 xIn12 xIn13 xIn14 xIn15 xIn16 xIn17 xIn18 xIn19 xIn20 xSubtle cIn1 cIn2 cIn3 cIn4 cIn5 Intercept_Zero cIn1_Zero cIn2_Zero cIn3_Zero cIn4_Zero cIn5_Zero

Parameter estimates for the selected model are shown in Figure 13 and Figure 14.

Figure 13 Parameter Estimates

Parameter Estimates

Parameter	DF	Estimate	Standard Error	Chi-Square	Pr > ChiSq
Intercept	1	-0.319361	0.003691	7485.3170	<.0001
xln1	1	-0.050986	0.001034	2433.1003	<.0001
xln2	1	0.099071	0.001033	9199.5521	<.0001
xln3	1	-0.150147	0.001034	21087.3993	<.0001
xln4	1	0.201001	0.001035	37748.7598	<.0001
xln5	1	-0.249349	0.001035	58090.6836	<.0001
xln6	1	0.300777	0.001035	84458.6925	<.0001
xln7	1	-0.351769	0.001036	115273.105	<.0001
xln8	1	0.400321	0.001037	149022.440	<.0001
xln9	1	-0.449548	0.001038	187488.110	<.0001
.					
.					
.					
cln5 2	1	0.747241	0.002696	76825.5556	<.0001
cln5 3	1	0.497155	0.002724	33308.6157	<.0001
cln5 4	1	0.246343	0.002859	7424.3835	<.0001
cln5 5	0	0	.	.	.

Figure 14 Zero-Inflation Parameter Estimates

Zero-Inflation Parameter Estimates

Parameter	DF	Estimate	Standard Error	Chi-Square	Pr > ChiSq
Intercept_Zero	1	6.527324	0.012450	274868.944	<.0001
cln1_Zero 1	1	-1.003686	0.004723	45164.8001	<.0001
cln1_Zero 2	0	0	.	.	.
cln2_Zero 1	1	4.011342	0.007866	260070.923	<.0001
cln2_Zero 2	1	1.998211	0.006136	106040.112	<.0001
cln2_Zero 3	0	0	.	.	.
cln3_Zero 1	1	-9.024949	0.014108	409226.351	<.0001
cln3_Zero 2	1	-6.014037	0.010579	323163.031	<.0001
cln3_Zero 3	1	-3.017093	0.007793	149886.519	<.0001
cln3_Zero 4	0	0	.	.	.
.					
.					
.					
cln5_Zero 3	1	-10.037194	0.016290	379661.851	<.0001
cln5_Zero 4	1	-5.022051	0.011111	204286.226	<.0001
cln5_Zero 5	0	0	.	.	.

The timing table in Figure 15 shows that PROC HPGENSELECT took slightly more than five minutes to run, most of the time spent in model evaluation and fitting.

Figure 15 Timing

Procedure Task Timing

Task	Seconds	Percent
Distributing Data	2.03	0.65%
Reading and Levelizing Data	112.12	35.87%
Candidate evaluation	71.30	22.81%
Candidate model fit	123.81	39.60%
Final model fit	3.35	1.07%

Model Selection by the LASSO Method

This example shows how you can use the HPGENSELECT procedure in single-machine mode to perform model selection by using the LASSO method. For more information about the LASSO method see, for example, Hastie, Tibshirani, and Friedman (2009).

The following statements use the first 50,000 observations from the data set GLMData to perform model selection by the LASSO method. Here, the Poisson response variable **yPoisson** depends on the regression variables in Table 2. The SELECTION statement specifies that the LASSO method be used and that the final model be selected on the basis of the minimum SBC criterion.

```
libname Data 'C:\Data';
proc hpgenselect data=Data.GLMData(obs=50000);
   class c:;
   model yPoisson = x: c: / dist=Poisson;
   selection method=lasso(choose=sbc) details=all;
run;
```

PROC HPGENSELECT uses a group LASSO method so that all parameters that are associated with levels of CLASS effects are included or excluded together.

Model selection is performed by varying a regularization parameter, which controls the amount of shrinkage in the regression coefficients. Those coefficients that are shrunk to zero are deemed to be out of the model, and coefficients that are not zero are in the model. The “Selection Details” table in Figure 16 shows the sequence of steps in the model selection process. Each successive step corresponds to a smaller regularization parameter (named **Lambda** in Figure 16), which corresponds to less regression coefficient shrinkage. Unlike the stepwise method, the LASSO method includes zero or more effects in each step.

Figure 16 LASSO Method Selection Summary

Selection Details						
Step	Description	Effects In Model	Lambda	AIC	AICC	BIC
0	Initial Model	1	1	215879.209	215879.209	215888.029
1	xIn17 entered	5	0.8	209124.963	209124.964	209169.062
	xIn18 entered	5	0.8	209124.963	209124.964	209169.062
	xIn19 entered	5	0.8	209124.963	209124.964	209169.062
	xIn20 entered	5	0.8	209124.963	209124.964	209169.062
2	xIn13 entered	10	0.64	196159.045	196159.053	196273.703
	xIn14 entered	10	0.64	196159.045	196159.053	196273.703
	xIn15 entered	10	0.64	196159.045	196159.053	196273.703
	xIn16 entered	10	0.64	196159.045	196159.053	196273.703
	cIn5 entered	10	0.64	196159.045	196159.053	196273.703
3	xIn11 entered	13	0.512	184409.836	184409.851	184577.412
	xIn12 entered	13	0.512	184409.836	184409.851	184577.412
	cIn4 entered	13	0.512	184409.836	184409.851	184577.412
4	xIn8 entered	17	0.4096	172994.506	172994.532	173215.001
	xIn9 entered	17	0.4096	172994.506	172994.532	173215.001
	xIn10 entered	17	0.4096	172994.506	172994.532	173215.001
	cIn3 entered	17	0.4096	172994.506	172994.532	173215.001
5	xIn7 entered	18	0.3277	163951.437	163951.465	164180.751
6	xIn6 entered	20	0.2621	157281.319	157281.354	157537.092
	cIn2 entered	20	0.2621	157281.319	157281.354	157537.092
7	xIn4 entered	22	0.2097	152456.156	152456.196	152729.569
	xIn5 entered	22	0.2097	152456.156	152456.196	152729.569
8		22	0.1678	148941.187	148941.227	149214.600
9	xIn3 entered	24	0.1342	146514.918	146514.963	146805.970
	cIn1 entered	24	0.1342	146514.918	146514.963	146805.970
10		24	0.1074	144859.219	144859.264	145150.272
11	xIn2 entered	25	0.0859	143758.891	143758.939	144058.764
12		25	0.0687	143012.513	143012.561	143312.386
13		25	0.055	142522.998	142523.045	142822.870
14	xIn1 entered	26	0.044	142204.818	142204.869	142513.510
15		26	0.0352	141992.171	141992.221	142300.863
16	xOut53 entered	29	0.0281	141855.391	141855.450	142190.542
	xOut56 entered	29	0.0281	141855.391	141855.450	142190.542
	xOut60 entered	29	0.0281	141855.391	141855.450	142190.542
17	xOut14 entered	33	0.0225	141768.812	141768.892	142156.883
	xOut72 entered	33	0.0225	141768.812	141768.892	142156.883
	xOut77 entered	33	0.0225	141768.812	141768.892	142156.883
	cOut3 entered	33	0.0225	141768.812	141768.892	142156.883
18	xOut5 entered	36	0.018	141704.761	141704.851	142119.291*
	xOut20 entered	36	0.018	141704.761	141704.851	142119.291*
	xOut44 entered	36	0.018	141704.761	141704.851	142119.291*
19	xOut6 entered	45	0.0144	141680.110	141680.252	142200.477
	xOut10 entered	45	0.0144	141680.110	141680.252	142200.477
	xOut16 entered	45	0.0144	141680.110	141680.252	142200.477
	xOut33 entered	45	0.0144	141680.110	141680.252	142200.477
	xOut51 entered	45	0.0144	141680.110	141680.252	142200.477
	xOut52 entered	45	0.0144	141680.110	141680.252	142200.477

Figure 16 *continued*

Selection Details						
Step	Description	Effects In Model	Lambda	AIC	AICC	BIC
	xOut78 entered	45	0.0144	141680.110	141680.252	142200.477
	cOut1 entered	45	0.0144	141680.110	141680.252	142200.477
	cOut4 entered	45	0.0144	141680.110	141680.252	142200.477
20	xOut3 entered	59	0.0115	141674.422	141674.656	142344.725
	xOut7 entered	59	0.0115	141674.422	141674.656	142344.725
	xOut8 entered	59	0.0115	141674.422	141674.656	142344.725
	xOut11 entered	59	0.0115	141674.422	141674.656	142344.725
	xOut35 entered	59	0.0115	141674.422	141674.656	142344.725
	xOut42 entered	59	0.0115	141674.422	141674.656	142344.725
	xOut54 entered	59	0.0115	141674.422	141674.656	142344.725
	xOut57 entered	59	0.0115	141674.422	141674.656	142344.725
	xOut64 entered	59	0.0115	141674.422	141674.656	142344.725
	xOut66 entered	59	0.0115	141674.422	141674.656	142344.725
	xOut70 entered	59	0.0115	141674.422	141674.656	142344.725
	xOut73 entered	59	0.0115	141674.422	141674.656	142344.725
	xOut74 entered	59	0.0115	141674.422	141674.656	142344.725
	cOut5 entered	59	0.0115	141674.422	141674.656	142344.725

*** Optimal Value of Criterion**

The model in step 18 was selected on the basis of the minimum SBC, and the effects that are selected are shown in Figure 17. The variables **xIn1–xIn20** and **cIn1–cIn5** were included in the model, and a few of the noise variables were also selected.

Figure 17 Effects Selected by the LASSO Method

Selected Effects:	Intercept xIn1 xIn2 xIn3 xIn4 xIn5 xIn6 xIn7 xIn8 xIn9 xIn10 xIn11 xIn12 xIn13 xIn14 xIn15 xIn16 xIn17 xIn18 xIn19 xIn20 xOut5 xOut14 xOut20 xOut44 xOut53 xOut56 xOut60 xOut72 xOut77 cIn1 cIn2 cIn3 cIn4 cIn5 cOut3

Parameter estimates for the selected model are shown in Figure 18.

Figure 18 Parameter Estimates

Parameter Estimates		
Parameter	**DF**	**Estimate**
Intercept	1	-0.306403
xIn1	1	-0.028968
xIn2	1	0.071474
xIn3	1	-0.132928
xIn4	1	0.194972
xIn5	1	-0.207019
xIn6	1	0.289436
xIn7	1	-0.338620
xIn8	1	0.393349
xIn9	1	-0.434013
.		
.		
.		
cIn3 3	1	0.124542
cIn3 4	0	0
cIn4 1	1	-0.767686
cIn4 2	1	-0.573186
cIn4 3	1	-0.389338
cIn4 4	1	-0.194644
cIn4 5	0	0
cIn5 1	1	0.945248
cIn5 2	1	0.707911
cIn5 3	1	0.445848
cIn5 4	1	0.207569
cIn5 5	0	0
cOut3 1	1	0.002422
cOut3 2	1	-0.001612
cOut3 3	1	-0.002164
cOut3 4	0	0

Distributed Mode

For more information about distributed mode, see Cohen and Rodriguez (2013) and *SAS/STAT 13.1 User's Guide: High-Performance Procedures*, at `http://support.sas.com/documentation/onlinedoc/stat/`. For more information about the LIBNAME statement, see *SAS/ACCESS 9.4 for Relational Databases: Reference, Third Edition*, at `http://support.sas.com/documentation/onlinedoc/access/`.

Summary of Benefits

The HPGENSELECT procedure, added in SAS/STAT 12.3, provides the following:

- model selection and model fitting for standard generalized linear model distributions and link functions
- a growing number of model selection techniques, now including the LASSO method
- zero-inflated models, ordinal and nominal multinomial models, and the Tweedie model
- predictive modeling for large data problems in a distributed computing environment
- the ability to use all available CPUs in single-machine mode on the server where SAS/STAT is installed

References

Cohen, R., and Rodriguez, R. N. (2013). "High-Performance Statistical Modeling." In *Proceedings of the SAS Global Forum 2013 Conference*. Cary, NC: SAS Institute Inc. http://support.sas.com/resources/papers/proceedings13/401-2013.pdf.

Hastie, T. J., Tibshirani, R. J., and Friedman, J. H. (2009). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. 2nd ed. New York: Springer-Verlag.

Paper SAS525-2017

Five Things You Should Know about Quantile Regression

Robert N. Rodriguez and Yonggang Yao, SAS Institute Inc.

Abstract

The increasing complexity of data in research and business analytics requires versatile, robust, and scalable methods of building explanatory and predictive statistical models. Quantile regression meets these requirements by fitting conditional quantiles of the response with a general linear model that assumes no parametric form for the conditional distribution of the response; it gives you information that you would not obtain directly from standard regression methods. Quantile regression yields valuable insights in applications such as risk management, where answers to important questions lie in modeling the tails of the conditional distribution. Furthermore, quantile regression is capable of modeling the entire conditional distribution; this is essential for applications such as ranking the performance of students on standardized exams. This expository paper explains the concepts and benefits of quantile regression, and it introduces you to the appropriate procedures in SAS/STAT® software.

Introduction

Students taking their first course in statistics learn to compute quantiles—more commonly referred to as percentiles—as descriptive statistics. But despite the widespread use of quantiles for data summarization, relatively few statisticians and analysts are acquainted with quantile regression as a method of statistical modeling, despite the availability of powerful computational tools that make this approach practical and advantageous for large data.

Quantile regression brings the familiar concept of a quantile into the framework of general linear models,

$$y_i = \beta_0 + \beta_1 x_{i1} + \cdots + \beta_p x_{ip} + \epsilon_i \ , \quad i = 1, \ldots, n$$

where the response y_i for the ith observation is continuous, and the predictors $x_{i1}, \ldots, x_{ip}$ represent main effects that consist of continuous or classification variables and their interactions or constructed effects. Quantile regression, which was introduced by Koenker and Bassett (1978), fits specified percentiles of the response, such as the 90th percentile, and can potentially describe the entire conditional distribution of the response.

This paper provides an introduction to quantile regression for statistical modeling; it focuses on the benefits of modeling the conditional distribution of the response as well as the procedures for quantile regression that are available in SAS/STAT software. The paper is organized into six sections:

- Basic Concepts of Quantile Regression
- Fitting Quantile Regression Models
- Building Quantile Regression Models
- Applying Quantile Regression to Financial Risk Management
- Applying Quantile Process Regression to Ranking Exam Performance
- Summary

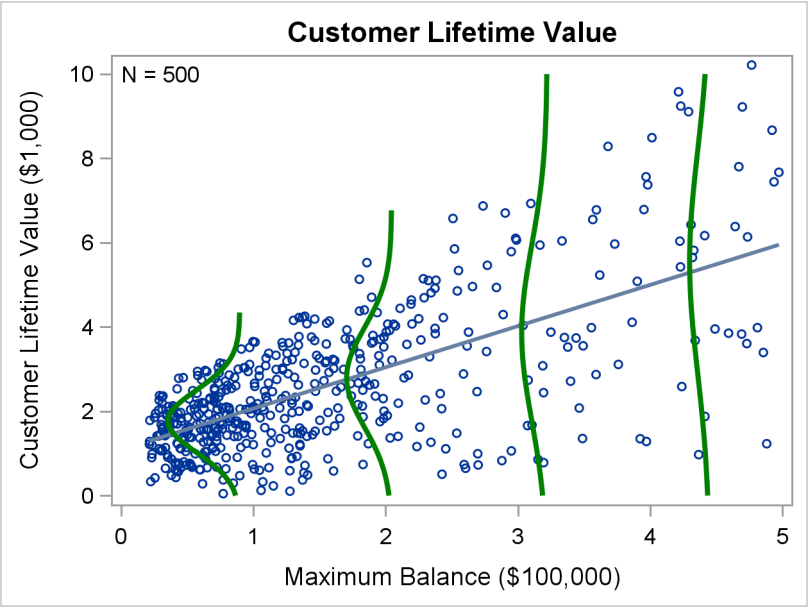
The first five sections present examples that illustrate the concepts and benefits of quantile regression along with procedure syntax and output. The summary distills these examples into five key points that will help you add quantile regression to your statistical toolkit.

Basic Concepts of Quantile Regression

Although quantile regression is most often used to model specific conditional quantiles of the response, its full potential lies in modeling the entire conditional distribution. By comparison, standard least squares regression models only the conditional mean of the response and is computationally less expensive. Quantile regression does not assume a particular parametric distribution for the response, nor does it assume a constant variance for the response, unlike least squares regression.

Figure 1 presents an example of regression data for which both the mean and the variance of the response increase as the predictor increases. In these data, which represent 500 bank customers, the response is the customer lifetime value (CLV) and the predictor is the maximum balance of the customer's account. The line represents a simple linear regression fit.

Figure 1 Variance of Customer Lifetime Value Increases with Maximum Balance

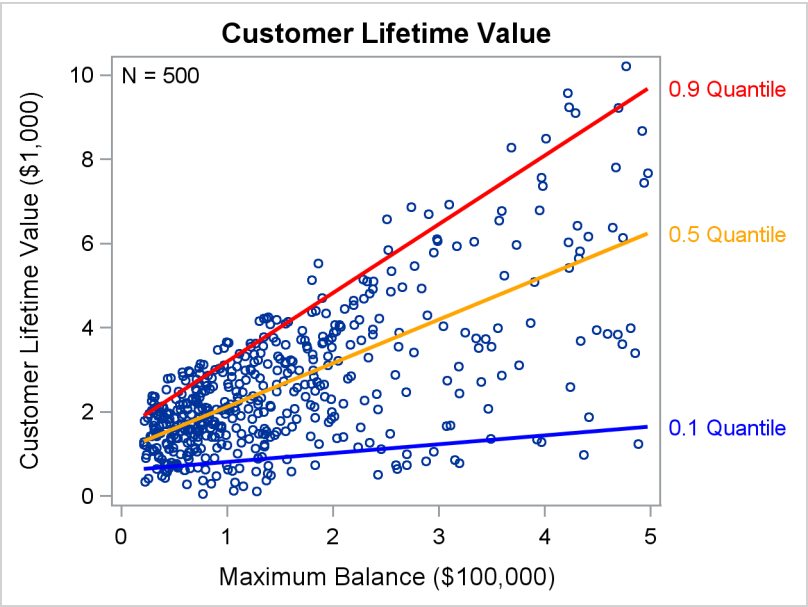


Least squares regression for a response Y and a predictor X models the conditional mean $E[Y|X]$, but it does not capture the conditional variance $\text{Var}[Y|X]$, much less the conditional distribution of Y given X.

The green curves in Figure 1 represent the conditional densities of CLV for four specific values of maximum balance. A set of densities for a comprehensive grid of values of maximum balance would provide a complete picture of the conditional distribution of CLV given maximum balance. Note that the densities shown here are normal only for the purpose of illustration.

Figure 2 shows fitted linear regression models for the quantile levels 0.10, 0.50, and 0.90, or equivalently, the 10th, 50th, and 90th percentiles.

Figure 2 Regression Models for Quantile Levels 0.10, 0.50, and 0.90

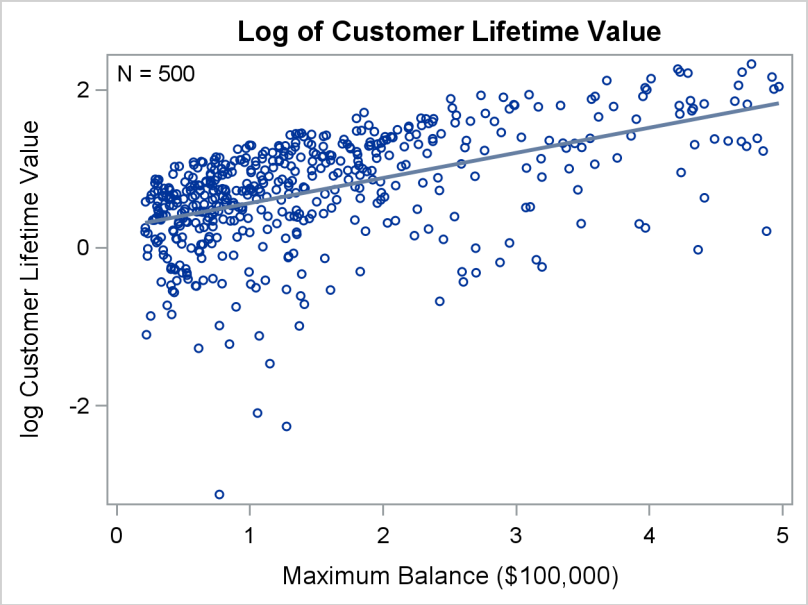


The quantile level is the probability (or the proportion of the population) that is associated with a quantile. The quantile level is often denoted by the Greek letter τ, and the corresponding conditional quantile of Y given X is often written as $Q_\tau(Y|X)$. The quantile level τ is the probability $\Pr[Y \leq Q_\tau(Y|X)|X]$, and it is the value of Y below which the proportion of the conditional response population is τ.

By fitting a series of regression models for a grid of values of τ in the interval (0,1), you can describe the entire conditional distribution of the response. The optimal grid choice depends on the data, and the more data you have, the more detail you can capture in the conditional distribution.

Quantile regression gives you a principled alternative to the usual practice of stabilizing the variance of heteroscedastic data with a monotone transformation $h(Y)$ before fitting a standard regression model. Depending on the data, it is often not possible to find a simple transformation that satisfies the assumption of constant variance. This is evident in Figure 3, where the variance of log(CLV) increases for maximum balances near \$100,000, and the conditional distributions are asymmetric.

Figure 3 Log Transformation of CLV



Even when a transformation does satisfy the assumptions for standard regression, the inverse transformation does not predict the mean of the response when applied to the predicted mean of the transformed response:

$$E(Y|X) \neq h^{-1}(E(h(Y)|X))$$

In contrast, the inverse transformation can be applied to the predicted quantiles of the transformed response:

$$Q_\tau(Y|X) = h^{-1}(Q_\tau(h(Y)|X))$$

Table 1 summarizes some important differences between standard regression and quantile regression.

Table 1 Comparison of Linear Regression and Quantile Regression

Linear Regression	Quantile Regression
Predicts the conditional mean $E(Y\|X)$	Predicts conditional quantiles $Q_\tau(Y\|X)$
Applies when n is small	Needs sufficient data
Often assumes normality	Is distribution agnostic
Does not preserve $E(Y\|X)$ under transformation	Preserves $Q_\tau(Y\|X)$ under transformation
Is sensitive to outliers	Is robust to response outliers
Is computationally inexpensive	Is computationally intensive

Koenker (2005) and Hao and Naiman (2007) provide excellent introductions to the theory and applications of quantile regression.

Fitting Quantile Regression Models

The standard regression model for the average response is

$$E(y_i) = \beta_0 + \beta_1 x_{i1} + \cdots + \beta_p x_{ip} \ , \quad i = 1, \ldots, n$$

and the β_j's are estimated by solving the least squares minimization problem

$$\min_{\beta_0,\ldots,\beta_p} \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j \right)^2$$

In contrast, the regression model for quantile level τ of the response is

$$Q_\tau(y_i) = \beta_0(\tau) + \beta_1(\tau) x_{i1} + \cdots + \beta_p(\tau) x_{ip} \ , \quad i = 1, \ldots, n$$

and the $\beta_j(\tau)$'s are estimated by solving the minimization problem

$$\min_{\beta_0(\tau),\ldots,\beta_p(\tau)} \sum_{i=1}^{n} \rho_\tau \left(y_i - \beta_0(\tau) - \sum_{j=1}^{p} x_{ij}\beta_j(\tau) \right)$$

where $\rho_\tau(r) = \tau \max(r, 0) + (1 - \tau) \max(-r, 0)$. The function $\rho_\tau(r)$ is referred to as the check loss, because its shape resembles a check mark.

For each quantile level τ, the solution to the minimization problem yields a distinct set of regression coefficients. Note that $\tau = 0.5$ corresponds to median regression and $2\rho_{0.5}(r)$ is the absolute value function.

Example: Modeling the 10th, 50th, and 90th Percentiles of Customer Lifetime Value

Returning to the customer lifetime value example, suppose that the goal is to target customers with low, medium, and high value after adjusting for 15 covariates (**X1**, ..., **X15**), which include the maximum balance, average overdraft, and total credit card amount used. Assume that low, medium, and high correspond to the 10th, 50th, and 90th percentiles of customer lifetime value, or equivalently, the 0.10, 0.50, and 0.90 quantiles.

The QUANTREG procedure in SAS/STAT software fits quantile regression models and performs statistical inference. The following statements use the QUANTREG procedure to model the three quantiles:

```
proc quantreg data=CLV ci=sparsity ;
   model CLV = x1-x15 / quantiles=0.10 0.50 0.90;
run;
```

You use the QUANTILES= option to specify the level for each quantile.

Figure 4 shows the "Model Information" table that the QUANTREG procedure produces.

Figure 4 Model Information

The QUANTREG Procedure

Model Information	
Data Set	WORK.CLV
Dependent Variable	CLV
Number of Independent Variables	15
Number of Observations	500
Optimization Algorithm	Simplex
Method for Confidence Limits	Sparsity

Number of Observations Read	500
Number of Observations Used	500

Figure 5 and Figure 6 show the parameter estimates for the 0.10 and 0.90 quantiles of CLV.

Figure 5 Parameter Estimates for Quantile Level 0.10

Parameter Estimates

Parameter	DF	Estimate	Standard Error	95% Confidence Limits		t Value	Pr > |t|
Intercept	1	9.9046	0.0477	9.8109	9.9982	207.71	<.0001
X1	1	0.8503	0.0428	0.7662	0.9343	19.87	<.0001
X2	1	0.9471	0.0367	0.8750	1.0193	25.81	<.0001
X3	1	0.9763	0.0397	0.8984	1.0543	24.62	<.0001
X4	1	0.9256	0.0413	0.8445	1.0067	22.43	<.0001
X5	1	0.6670	0.0428	0.5828	0.7511	15.58	<.0001
X6	1	0.2905	0.0443	0.2034	0.3776	6.55	<.0001
X7	1	0.2981	0.0393	0.2208	0.3754	7.58	<.0001
X8	1	0.2094	0.0413	0.1283	0.2905	5.07	<.0001
X9	1	-0.0633	0.0423	-0.1464	0.0199	-1.49	0.1356
X10	1	0.0129	0.0400	-0.0658	0.0916	0.32	0.7473
X11	1	0.1084	0.0421	0.0257	0.1912	2.57	0.0103
X12	1	-0.0249	0.0392	-0.1019	0.0520	-0.64	0.5248
X13	1	-0.0505	0.0410	-0.1311	0.0300	-1.23	0.2182
X14	1	0.2009	0.0548	0.0932	0.3086	3.66	0.0003
X15	1	0.1623	0.0433	0.0773	0.2473	3.75	0.0002

Figure 6 Parameter Estimates for Quantile Level 0.90

Parameter Estimates

Parameter	DF	Estimate	Standard Error	95% Confidence Limits		t Value	Pr > |t|
Intercept	1	10.1007	0.1386	9.8283	10.3730	72.87	<.0001
X1	1	0.0191	0.1485	-0.2726	0.3109	0.13	0.8975
X2	1	0.9539	0.1294	0.6996	1.2081	7.37	<.0001
X3	1	0.0721	0.1328	-0.1889	0.3332	0.54	0.5874
X4	1	1.1171	0.1243	0.8728	1.3613	8.99	<.0001
X5	1	-0.0317	0.1501	-0.3266	0.2631	-0.21	0.8326
X6	1	0.1096	0.1581	-0.2010	0.4202	0.69	0.4885
X7	1	0.2428	0.1436	-0.0394	0.5250	1.69	0.0915
X8	1	-0.0743	0.1364	-0.3424	0.1938	-0.54	0.5864
X9	1	0.0918	0.1401	-0.1835	0.3670	0.66	0.5127
X10	1	-0.2426	0.1481	-0.5336	0.0483	-1.64	0.1019
X11	1	0.9099	0.1414	0.6321	1.1878	6.44	<.0001
X12	1	0.7759	0.1353	0.5099	1.0418	5.73	<.0001
X13	1	0.5380	0.1392	0.2645	0.8115	3.87	0.0001
X14	1	0.6897	0.1475	0.3999	0.9796	4.68	<.0001
X15	1	1.0145	0.1516	0.7165	1.3124	6.69	<.0001

Note that the results in Figure 5 and Figure 6 are different. For example, the estimate for **X1** is significant in the model for the 0.10 quantile, but it is not significant in the model for the 0.90 quantile. In general, quantile regression produces a distinct set of parameter estimates and predictions for each quantile level.

The QUANTREG procedure provides extensive features for statistical inference, which are not illustrated here. These include the following:

- simplex, interior point, and smooth algorithms for model fitting
- sparsity and bootstrap resampling methods for confidence limits
- Wald, likelihood ratio, and rank-score tests

You can also use PROC QUANTREG to carry out quantile process regression, which fits models for an entire grid of values of τ in the interval (0,1). The following statements illustrate quantile process regression by specifying a grid that is spaced uniformly in increments of 0.02:

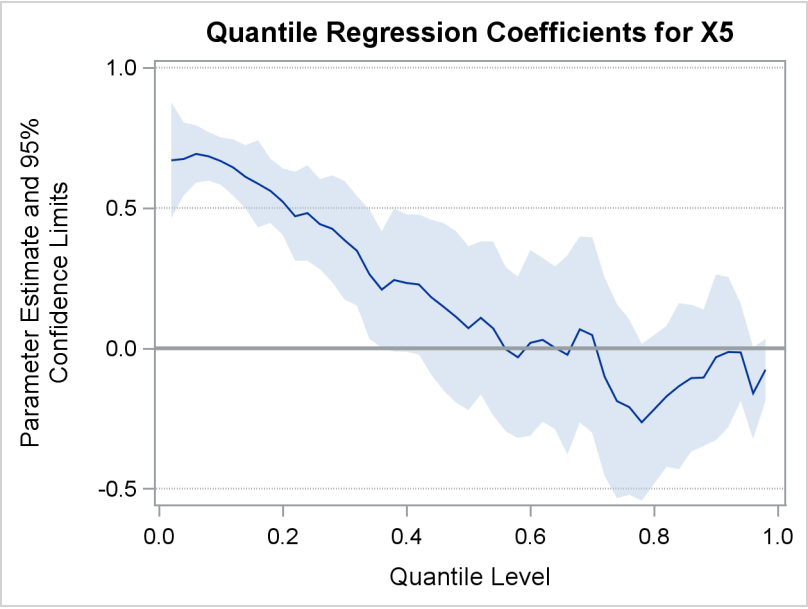
```
ods output ParameterEstimates=Estimates;
proc quantreg data=CLV ci=sparsity ;
   model CLV = x1-x15 / quantiles=0.02 to 0.98 by 0.02;
run;
```

The next statements use the parameter estimates and confidence limits that PROC QUANTREG produces to create a quantile process plot for **X5**:

```
%MACRO ProcessPlot(Parm=);
data ParmEst; set Estimates;
   if Parameter EQ "&Parm";
run;

title "Quantile Regression Coefficients for &Parm";
proc sgplot data=ParmEst noautolegend;
   band   x=quantile lower=LowerCL  upper=UpperCL / transparency=0.5;
   series x=quantile y=estimate ;
   refline 0 / axis=y lineattrs=(thickness=2px);
   yaxis label='Parameter Estimate and 95% Confidence Limits'
         grid gridattrs=(thickness=1px color=gray pattern=dot);
   xaxis label='Quantile Level';
run;
%MEND ProcessPlot;

%ProcessPlot(Parm=X5)
```

The quantile process plot, shown in Figure 7, displays the parameter estimates and 95% confidence limits as a function of quantile level. The plot reveals that **X5** positively affects the lower tail of the distribution of **CLV**, because the lower confidence limits are greater than 0 for quantile levels less than 0.37.

Figure 7 Quantile Process Plot for X5

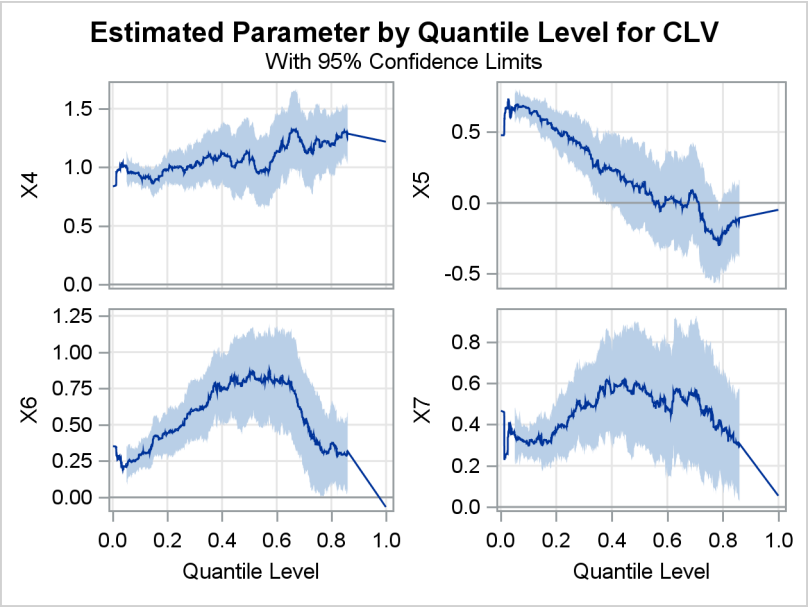


A drawback of specifying an explicit grid for quantile process regression is that the grid resolution might not be optimal for the data. As an alternative, you can search for the optimal grid, which depends on the data, by specifying the QUANTILE=PROCESS option in the MODEL statement. The optimal grid is usually not evenly spaced. The following statements illustrate the option:

```
proc quantreg data=CLV ci=sparsity ;
   model CLV = x1-x15 / quantile=process plot=quantplot;
run;
```

The PLOT=QUANTPLOT option requests paneled displays of quantile process plots for the intercept term and all the predictors. Figure 8 shows the second of the four displays that are produced, which includes the plot for **X5**.

Figure 8 Quantile Process Plots (Panel 2)



The plot for **X5** in Figure 7 is a linearly interpolated low-resolution counterpart of the optimal plot for **X5** in Figure 8. However, computing this low-resolution counterpart is much more efficient than computing the optimal one.

Paneled quantile process plots help you to readily identify which predictors are associated with different parts of the response distribution.

Building Quantile Regression Models

One of the most frequently asked questions in the framework of standard regression is this: "I have hundreds of variables—even thousands. Which should I include in my model?" The same question arises in the framework of quantile regression.

For standard regression, the flagship SAS/STAT procedure for model building is the GLMSELECT procedure. This procedure selects effects in general linear models of the form

$$y_i = \beta_0 + \beta_1 x_{i1} + \cdots + \beta_p x_{ip} + \epsilon_i \ , \quad i = 1, \ldots, n$$

where the response y_i is continuous. The predictors $x_{i1}, \ldots, x_{ip}$ represent main effects that consist of continuous or classification variables and their interactions or constructed effects.

The QUANTSELECT procedure performs effect selection for quantile regression. Like the GLMSELECT procedure, it is designed primarily for effect selection, and it does not include regression diagnostics or hypothesis testing, which are available in the QUANTREG procedure.

If you have too many predictors, the model can overfit the training data, leading to poor prediction when you apply the model to future data. To deal with this problem, the QUANTSELECT procedure supports a variety of model selection methods, including the lasso method; these are summarized in Table 2.

Table 2 Effect Selection Methods in the QUANTSELECT Procedure

Method	Description
Forward selection	Starts with no effects and adds effects
Backward elimination	Starts with all effects and deletes effects
Stepwise selection	Starts with no effects; effects are added and can be deleted
Lasso	Adds and deletes effects based on a constrained version of check loss where the ℓ_1 norm of the βs is penalized
Adaptive lasso	Constrains sum of absolute weighted βs; some βs set to 0

The QUANTSELECT procedure offers extensive capabilities for customizing model selection by using a wide variety of selection and stopping criteria, including significance-level-based criteria and information criteria. The procedure also enables you to use validation-based criteria by partitioning the data into subsets for training, validation, and testing.

The following example illustrates the use of the QUANTSELECT procedure.

Example: Predicting the Close Rates of Retail Stores

The close rate of a retail store is the percentage of shoppers who enter the store and make a purchase. Understanding what factors predict close rate is critical to the profitability and growth of large retail companies, and a regression model is constructed to study this question.

The close rates of 500 stores are saved in a data set named **Stores**. Each observation provides information about a store. The variables available for the model are the response **Close_Rate** and the following candidate predictors:

- **X1**, . . . , **X20**, which measure 20 general characteristics of stores, such as floor size and number of employees
- **P1**, . . . , **P6**, which measure six promotional activities, such as advertising and sales
- **L1**, . . . , **L6**, which measure special layouts of items in six departments

In practice, close rate data can involve hundreds of candidate predictors. A small set is used here for illustration.

By building a standard regression model, you can answer questions such as the following:

How can I predict the close rate of a new store?

Which variables explain the average close rate of a store?

By building a quantile regression model, you can answer a different set of questions:

How can I predict a high close rate, such as the 90th percentile of the close rate distribution?

Which variables explain a low close rate, such as the 10th percentile of the close rate distribution?

Are there variables that differentiate between low and high close rates?

The following statements use the QUANTSELECT procedure to build quantile regression models for levels 0.1, 0.5, and 0.9:

```
proc quantselect data=Stores plots=Coefficients seed=15531;
   model Close_Rate = X1-X20 L1-L6 P1-P6 / quantile = 0.1 0.5 0.9
                                           selection=lasso(sh=3);
   partition fraction(validate=0.3);
run;
```

The SELECTION= option specifies the lasso method with a stop horizon of 3. The PARTITION statement reserves 30% of the data for validation, leaving the remaining 70% for training.

Figure 9 summarizes the effect selection process for quantile level 0.1. The lasso method generates a sequence of candidate models, and the process chooses the model that minimizes the average check loss (ACL) computed from the validation data. The process stops at Step 14.

Figure 9 Selection Summary for Quantile Level 0.1

The QUANTSELECT Procedure
Quantile Level = 0.1

Selection Summary

Step	Effect Entered	Effect Removed	Number Effects In	Validation ACL
0	Intercept		1	0.1578
1	X2		2	0.1667
2	X4		3	0.1566
3	P3		4	0.1380
4	P1		5	0.1326
5	P2		6	0.1119
6	P4		7	0.1104
7	X20		8	0.1113
8	X3		9	0.1111
9	P5		10	0.1096
10		P5	9	0.1111
11	P5		10	0.1096
12		X3	9	0.1083*
13	L1		10	0.1105
14	X3		11	0.1117

The coefficient progression plot in Figure 10 visualizes the selection process. The variables **X2** and **X4** are the first to enter the model.

Figure 10 Coefficient Progression for Quantile Level 0.1

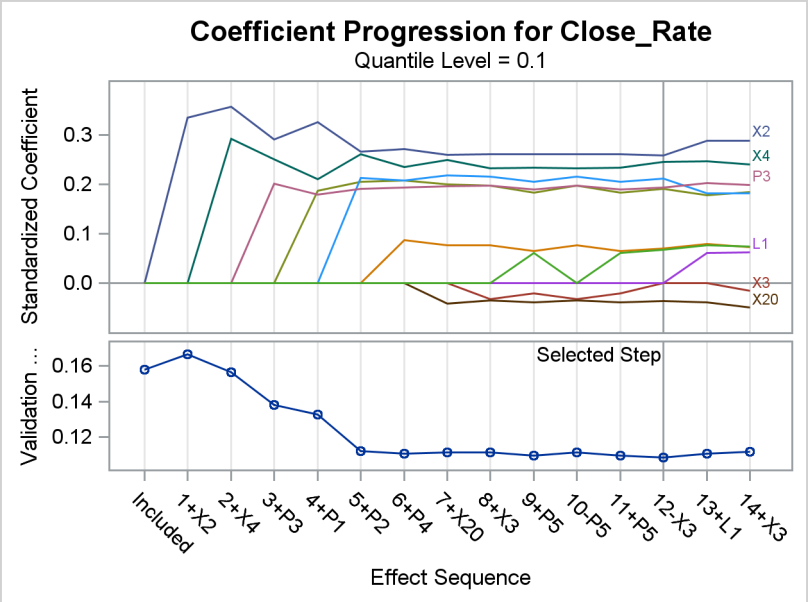


Figure 11 shows the fit statistics and parameter estimates for the final model for quantile level 0.1. The QUANTSELECT procedure produces parallel but distinct sets of results for quantile levels 0.5 and 0.9.

Figure 11 Fit Statistics and Parameter Estimates for Model Selected for Quantile Level 0.1

The QUANTSELECT Procedure
Quantile Level = 0.1

Fit Statistics	
Objective Function	36.17929
R1	0.38327
Adj R1	0.36909
AIC	-1616.52369
AICC	-1616.00496
SBC	-1581.62407
ACL (Train)	0.10134
ACL (Validate)	0.10826

Parameter Estimates

Parameter	DF	Estimate	Standardized Estimate
Intercept	1	60.097618	0
X2	1	0.953402	0.258498
X4	1	0.933705	0.245902
X20	1	-0.140895	-0.035981
P1	1	0.724145	0.190798
P2	1	0.783880	0.211752
P3	1	0.696274	0.193163
P4	1	0.260641	0.069442
P5	1	0.242147	0.067135

Figure 12 and Figure 13 show the parameter estimates for the final models for quantile levels 0.5 and 0.9.

Figure 12 Parameter Estimates for Model Selected for Quantile Level 0.5

Parameter Estimates

Parameter	DF	Estimate	Standardized Estimate
Intercept	1	60.950579	0
X2	1	1.508595	0.409029
X4	1	0.710687	0.187168
P3	1	0.361047	0.100163
P4	1	0.669943	0.178491
P5	1	0.544278	0.150902

Figure 13 Parameter Estimates for Model Selected for Quantile Level 0.9

Parameter Estimates

Parameter	DF	Estimate	Standardized Estimate
Intercept	1	61.079231	0
X2	1	0.982776	0.266463
X4	1	1.118507	0.294572
L2	1	1.027725	0.297930
L3	1	0.859988	0.240257
L5	1	0.672210	0.186588
P5	1	0.192967	0.053500

A sparse model that contains only six variables (**X2**, **X4**, **L2**, **L3**, **L5**, and **P5**) is selected as the best model for predicting the 90th percentile. The layout variables **L2**, **L3**, and **L5** are in this model, but not in the models for the 10th and 50th percentiles. The variables **X2** and **X4** are common to all three models. These results give you insights about store performance that you would not obtain directly from standard regression methods.

Applying Quantile Regression to Financial Risk Management

Although quantile regression can model the entire conditional distribution of the response, it often leads to deep insights and valuable solutions in situations where the most useful information lies in the tails. This is demonstrated by the application of quantile regression to the estimation of value at risk (VaR).

Financial institutions and their regulators use VaR as the standard measure of market risk. The quantity VaR measures market risk by how much a portfolio can lose within a given time period, with a specified confidence level $(1-\tau)$, where τ is often set to 0.01 or 0.05. More precisely, the value at risk at time t (denoted by VaR_t) is the conditional quantile of future portfolio values that satisfies the equation

$$\Pr[y_t < -\text{VaR}_t] = \tau\ , \quad 0 < \tau < 1$$

where $\{y_t\}$ is the series of asset returns and Ω_τ, the information available at time t, includes covariates and values of past asset returns.

Commonly used methods of estimating VaR include copula models, ARCH models, and GARCH models (GARCH stands for generalized autoregressive conditional homoscedasticity). SAS/ETS® software provides a number of procedures for fitting these models; see the *SAS/ETS 14.2 User's Guide*.

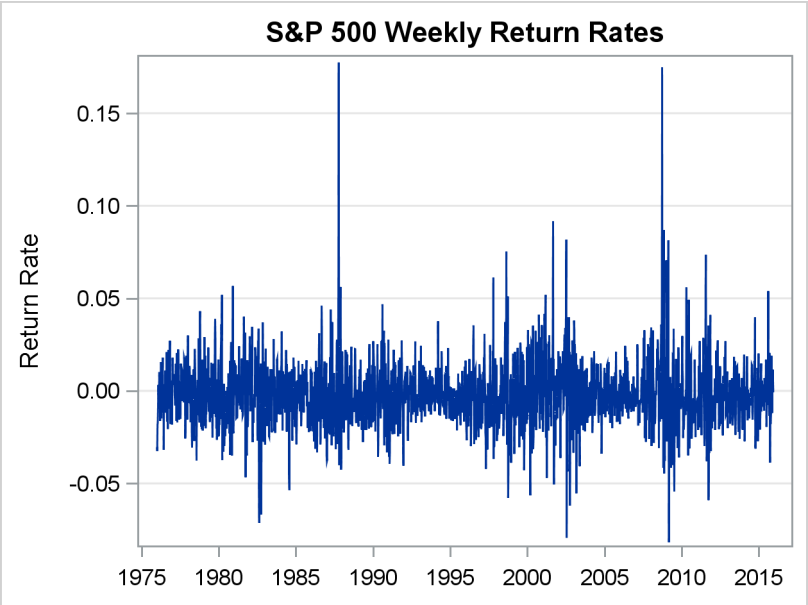
ARCH and GARCH models assume that financial returns are normally distributed. However, as pointed out by Xiao, Guo, and Lam (2015, p. 1144), the distributions of financial time series and market returns often display skewness and heavy tails. Extreme values of returns can bias estimates of VaR that are produced using ARCH and GARCH models.

Autoregressive quantile regression provides a robust alternative for estimating VaR that does not assume normality (Koenker and Zhao 1996; Koenker and Xiao 2006; Xiao and Koenker 2009). This is illustrated by the next example, which is patterned after the analysis of equity market indexes by Xiao, Guo, and Lam (2015, pp. 1159–1166).

Example: Computing Value at Risk for S&P 500 Return Rates

Figure 14 displays weekly return rates of the S&P 500 Composite Index.

Figure 14 Weekly Return Rates of the S&P 500 Index

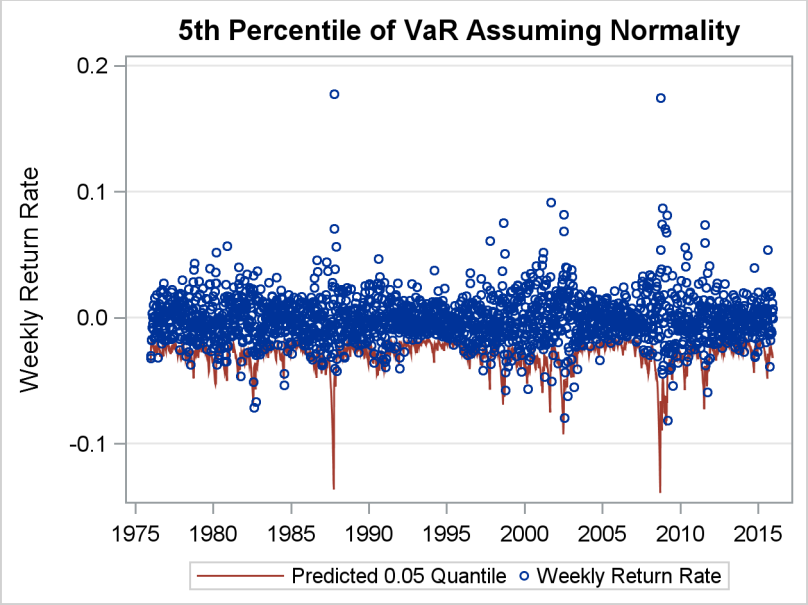


The following statements compute predicted 0.05 quantiles for the weekly return rate by fitting a standard GARCH(1,1) model, which assumes that the rate is normally distributed:

```
%let VaRQtlLevel=0.05; /* 95% confidence */

proc varmax data=SP500;
   model ReturnRate;
   garch form=ccc subform=garch p=1 q=1;
   output out=g11 lead=1;
   id date interval=week;
run;

data g11;
   set g11;
   qt=for1 + std1*quantile('normal',&VaRQtlLevel);
run;

title "%sysevalf(&VarQtlLevel*100)th Percentile of VaR Assuming Normality";
proc sgplot data=g11;
   series  y=qt         x=date / lineattrs=graphdata2(thickness=1);
   scatter y=ReturnRate x=date / markerattrs=(size=5);
   yaxis grid;
   xaxis display=(nolabel) type=linear %tick offsetmax=0.05 ;
   label ReturnRate = "Weekly Return Rate"
         qt         = "Predicted &VarQtlLevel Quantile";
run;
```

The results are plotted in Figure 15. The proportion of observed return rates that are less than the predicted quantiles (highlighted in red) is less than 0.05, because the model assumes that the rate distribution is symmetric when it is actually skewed in the high direction. Therefore, the predicted 0.05 quantile based on this model overestimates the risk.

Figure 15 Analysis Based on GARCH and Normal Quantile Regression Models



The robustness of quantile regression makes it an attractive alternative for modeling the heavy-tailed behavior of portfolio returns. Xiao, Guo, and Lam (2015, p. 1161) discuss an approach that uses an AR(1)–ARCH(7) quantile regression model for the return rate at time t.

The following statements implement a similar approach in two steps, the first of which fits an AR(1)–ARCH(7) model by using the VARMAX procedure in SAS/ETS software:

```
proc varmax data=SP500;
   model ReturnRate / p=1;
   garch form=ccc subform=garch q=6;
   output out=a1a7 lead=1;
   id date interval=week;
 run;
```

The MODEL statement specifies an AR(1) (autoregressive order one) model for the mean,

$$r_t = \alpha_0 + \alpha_1 r_{t-1} + u_t$$

where $u_t = \sigma_t \epsilon_t$. The GARCH statement specifies the ARCH(7) component:

$$\sigma_t = \gamma_0 + \gamma_1 |u_{t-1}| + \cdots + \gamma_6 |u_{t-6}|$$

No parametric distribution is assumed for ϵ_t. The VARMAX procedure creates an output data set named **A1A7** that saves the standard error of prediction in the variable **STD1**.

The second step fits a quantile regression model for level τ of VaR_t, which conditions on lagged values of the standard error that was estimated by PROC VARMAX:

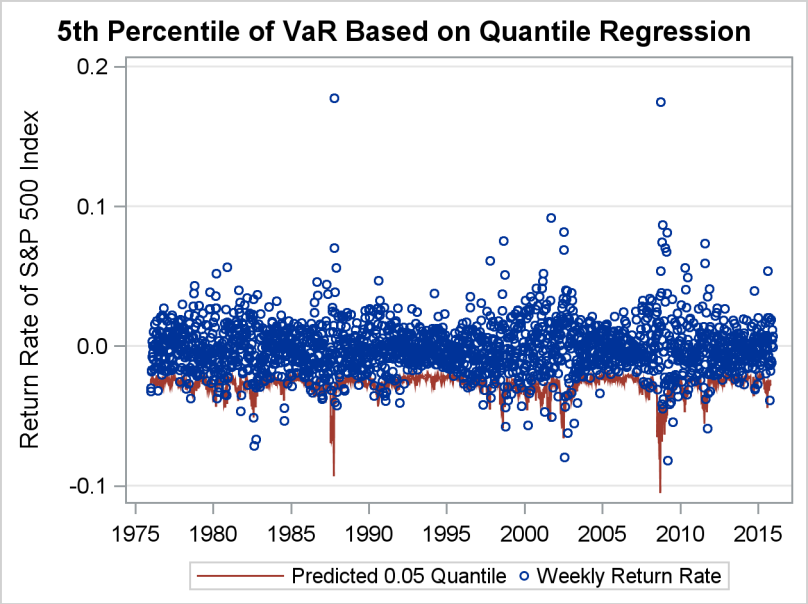
```
data a1a7;
   set a1a7;

   /* Lagged predictors for quantile regression */
   STD2=lag1(std1);
   STD3=lag2(std1);
   STD4=lag3(std1);
   STD5=lag4(std1);
   STD6=lag5(std1);
   STD7=lag6(std1);
run;

proc quantreg data=a1a7 ci=none;
   model ReturnRate = std1-std7 / quantile=&VaRQtlLevel;
   output out=qr p=p;
   id date;
   label ReturnRate = "Return Rate";
run;

title "%sysevalf(&VarQtlLevel*100)th Percentile of VaR Based on Quantile Regression";
proc sgplot data=qr;
   series  y=p x=date / lineattrs=graphdata2(thickness=1);
   scatter y=ReturnRate x=date / markerattrs=(size=5);
   yaxis label="Return Rate of S&P 500 Index" grid;
   xaxis display=(nolabel) type=linear %tick offsetmax=0.05 ;
   label p ="Predicted &VarQtlLevel Quantile";
   label ReturnRate="Weekly Return Rate";
run;
```

The form of the model is

$$Q_\tau(\text{VaR}_t) = \gamma_0(\tau) + \gamma_1(\tau)|u_{t-1}| + \cdots + \gamma_6(\tau)|u_{t-6}|$$

The QUANTREG procedure computes the predicted 0.05 quantiles of the return rates on the AR(1)–ARCH(7) variance predictions. This guarantees that precisely 5% of the observed return rates lie below the predicted 0.05 quantiles of VaR_t, which are plotted in Figure 16.

Figure 16 Analysis Based on Quantile Regression AR(1)–ARCH(7) Model



Applying Quantile Process Regression to Ranking Exam Performance

In the applications of quantile regression that have been discussed so far in this paper, the goal has been to predict conditional quantiles for specified quantile levels. However, in many applications—such as ranking the performance of students on exams—the goal is to predict conditional quantile *levels* for specified observations. You can use quantile process regression for this purpose because it predicts the entire conditional distribution of the response, and quantile levels are simply probabilities that can be computed from this distribution.

Consider a student named Mary who scored 1948 points on a college entrance exam. You cannot rank her performance unless you know the distribution of scores for all students who took the exam. Mary, her parents, and her teachers are primarily interested in her quantile level, which is 0.9. This informs them that she performed better than 90% of the students who took the exam.

Mathematically, if Y denotes the score for a randomly selected student who took the exam, and if $F(y)$ denotes the cumulative distribution function (CDF) of Y, then the CDF determines the quantile level for any observed value of Y. In particular, Mary's quantile level is $F(1948) = \Pr[Y \leq 1948] = 0.9$.

In practice, the quantile levels of a response variable Y must often be adjusted for the effects of covariates $X_1, \ldots, X_p$. This requires that the quantile levels be computed from the conditional distribution $F(y \mid X_1 = x_1, \ldots, X_p = x_p)$.

To see why such an adjustment makes a difference, consider a second student named Michael, who took the exam and scored 1617 points. Michael's quantile level is $F(1617) = 0.5$, so you might conclude that Mary performed better than Michael. However, if you learn that Mary is 17 and Michael is 12, then the question becomes, How did Mary and Michael perform relative to the other students in their age groups? The answer is given by their respective conditional quantile levels, which are $F(1948 \mid \text{Age} = 17)$ and $F(1617 \mid \text{Age} = 12)$.

With sufficient data, quantile process regression gives you a flexible method of obtaining adjusted quantile levels that does not require you to assume a parametric form for the conditional distribution of the response. The following example illustrates the computations.

Example: Ranking Exam Scores

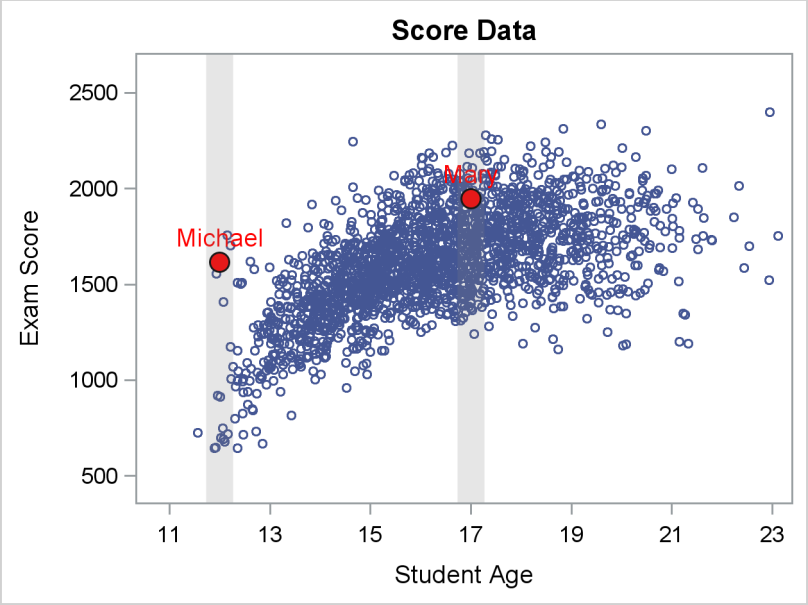
A SAS data set named **Score** contains three variables, **Name**, **Age**, and **Score**, which provide the names, ages, and scores of the 2,000 students who took the exam, including Mary and Michael. Figure 17 lists the first five observations.

Figure 17 Partial Listing of **Score**

Obs	Name	Age	Score
1	Michael	12.0	1617
2	Mary	17.0	1948
3	Yonggang	15.3	1661
4	Bob	15.3	1517
5	Youngjin	13.1	1305

The scatter plot in Figure 18 highlights the observations for Mary and Michael. Note that the distribution for 12-year-olds is different from the distribution for 17-year-olds. For a fair comparison, the quantile levels for Mary and Michael should be adjusted for the effect of age.

Figure 18 Exam Score versus Age

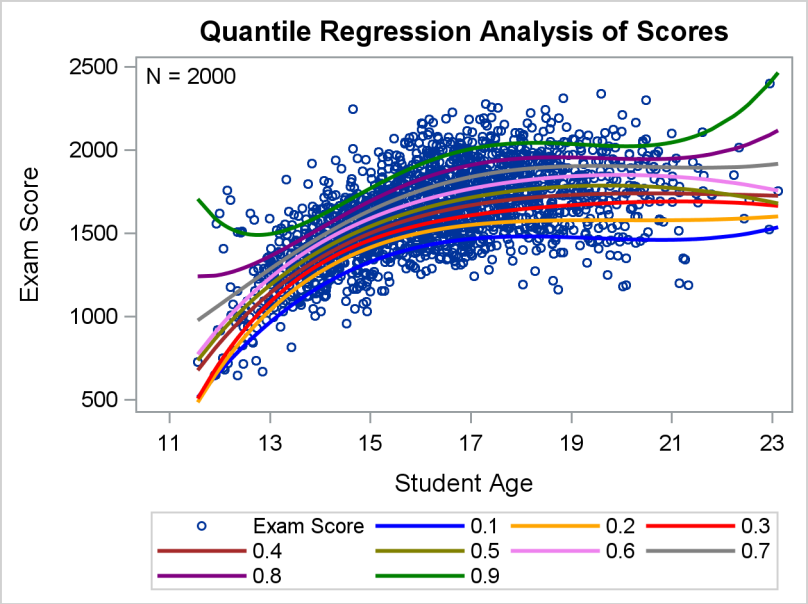


The first step in making this comparison is to fit a model that adequately describes the conditional score distribution. To account for the nonlinearity in the data, the following statements fit a quantile regression model that has four predictors, three of which are derived from **Age**. To examine the fit, it suffices to specify nine equally spaced quantile levels in the MODEL statement for PROC QUANTREG.

```
data Score;
   set Score;
   Age2   = Age*Age;
   Age3   = Age2*Age;
   AgeInv = 1/Age;
   label Score = "Exam Score"
         Age   =  "Student Age";
run;

proc quantreg data=Score;
   model Score = Age Age2 Age3 AgeInv / quantile = 0.10 to 0.90 by 0.1;
   output out=ModelFit p=Predicted;
   label Score = "Exam Score"
         Age   = "Student Age";
run;
```

The fit plot in Figure 19 shows that the model adequately captures the nonlinearity.

Figure 19 Conditional Quantile Regression Models for Exam Scores

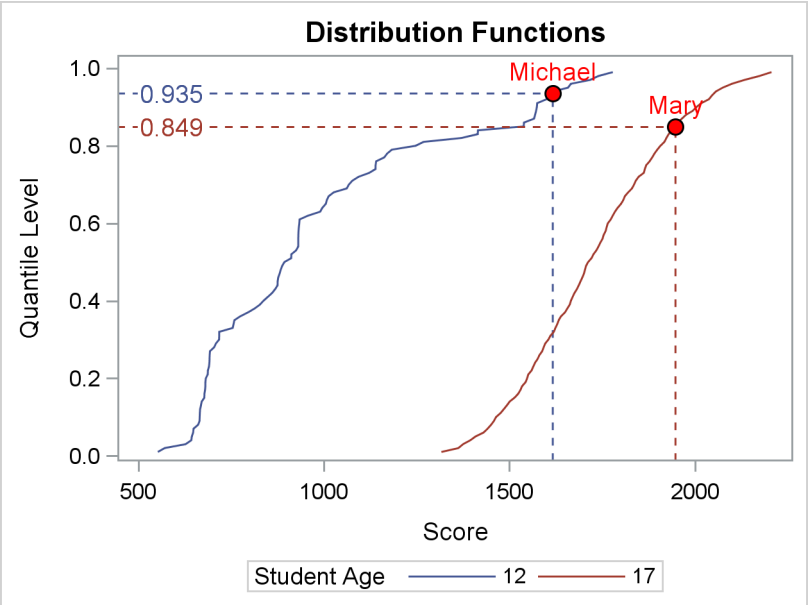


In the next statements, the model variables serve as input to the QPRFIT macro, which refits the model for an extensive grid of quantile levels ($\tau = 0.01, 0.02, \ldots, 0.99$). The macro then forms sets of predicted quantiles that condition on the values of **Age** for Mary and Michael, whose observations are identified by **Name** in the IDDATA= data set. From each set, the macro constructs a conditional CDF, which is used to compute the adjusted quantile levels.

```
data ScoreID;
   Name='Mary';    output;
   Name='Michael'; output;
run;

%qprFit(data=Score, depvar=Score, indvar=Age Age2 Age3 AgeInv, onevar=Age,
        nodes=99, iddata=ScoreID, showPDFs=1, showdist=1)
```

The INDVAR= option specifies the predictors **Age**, **Age2**, **Age3**, and **AgeInv**. The ONEVAR= option indicates that the last three predictors are derived from **Age**. As shown in Figure 20, the macro plots the CDFs for 12-year-old and 17-year-old students.

Figure 20 Conditional Distribution Functions of Scores for Ages 12 and 17



The drop lines indicate the scores and quantile levels for Mary and Michael. The macro also produces the table shown in Figure 21, which summarizes the results.

Figure 21 Regression-Adjusted and Univariate Quantile Levels for Mary and Michael

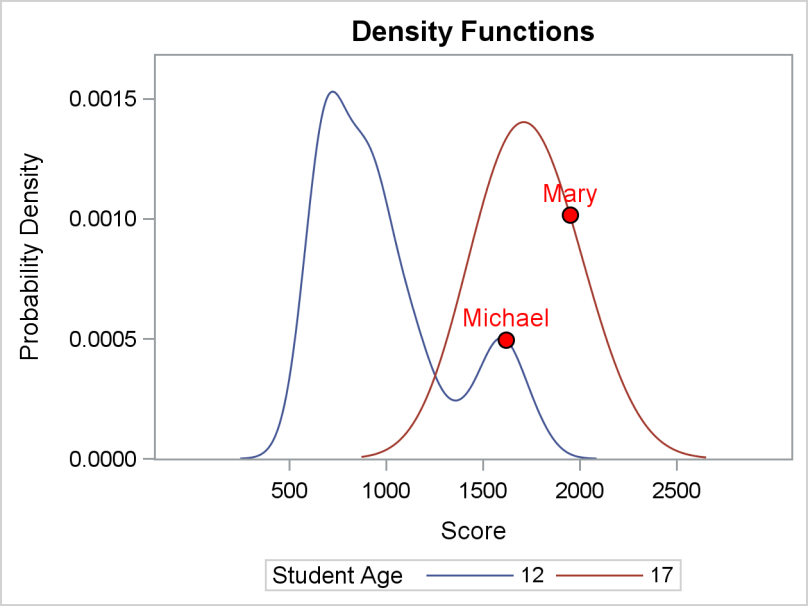
Statistics for the Highlighted Observations

Obs	Name	Score	Age	Mean	Median	Regression Quantile Level	Sample Quantile Level
1	Michael	1617	12	971.43	893.45	0.93500	0.50075
2	Mary	1948	17	1709.94	1712.36	0.84851	0.90025

Based on the regression-adjusted quantile levels, Michael is at the 93.50 percentile for 12-year-olds, and Mary is at the 84.85 percentile for 17-year-olds.

The SHOWPDFS=1 option requests the density estimates shown in Figure 22.

Figure 22 Conditional Density Functions of Exam Scores for Ages 12 and 17



The Appendix explains the QPRFIT macro in more detail.

Summary

This paper makes five key points:

1. Quantile regression is a highly versatile statistical modeling approach because it uses a general linear model to fit conditional quantiles of the response without assuming a parametric distribution.
2. Quantile process regression estimates the entire conditional distribution of the response, and it allows the shape of the distribution to depend on the predictors.
3. Quantile process plots reveal the effects of predictors on different parts of the response distribution.
4. Quantile regression can predict the quantile levels of observations while adjusting for the effects of covariates.
5. The QUANTREG and QUANTSELECT procedures give you powerful tools for fitting and building quantile regression models, making them feasible for applications with large data.

Note that SAS/STAT software also provides the QUANTLIFE procedure, which fits quantile regression models for censored data, and the HPQUANTSELECT procedure, a high-performance procedure for fitting and building quantile regression models that runs in either single-machine mode or distributed mode (the latter requires SAS® High-Performance Statistics). SAS® Viya™ provides the QTRSELECT procedure, which fits and builds quantile regression models.

Appendix: The QPRFIT Macro

The QPRFIT macro fits a quantile process regression model and performs conditional distribution analysis for a subset of specified observations. The macro is available in the SAS autocall library starting with SAS® 9.4M4, and it requires SAS/STAT and SAS/IML® software. You invoke the macro as follows:

```
                          /*-------------------------------------------------*/
%macro qprFit(            /* Quantile regression specialized output.         */
                          /*-------------------------------------------------*/
      data=_last_,        /* Input data set.                                 */
      depvar=,            /* Dependent or response variable.                 */
      indvar=,            /* Independent or explanatory variables.           */
      onevar=,            /* 1, y, Y, t, T - show fit and scatter plots,     */
                          /* which are appropriate for a single independent  */
                          /* variable. (Only the first character is checked.) */
                          /* Other nonblank - do not show fit plot.          */
                          /* By default, ONEVAR is true when there is a      */
                          /* single independent variable and false otherwise. */
                          /* Set ONEVAR= to true when there are multiple     */
                          /* independent variables but they form a polynomial */
                          /* or other nonlinear function of a single         */
                          /* variable. When ONEVAR is true, the first        */
                          /* independent variable is used in the fit and     */
                          /* scatter plots.                                  */
      nodes=19,           /* Quantile process step size is 1 / (1 + NODES).  */
                          /* The default step size is 0.05.                  */
      peData=qprPE,       /* Output parameter-estimates data set for the     */
                          /* quantile process regression model.              */
                          /* This data set is used in the qprPredict macro.  */
      iddata=,            /* Data set with ID variable for the observations  */
                          /* to highlight.  Only one variable is permitted in */
                          /* the data set, and the same variable must be in  */
                          /* the DATA= data set.                             */
      showPDFs=0,         /* 1, y, Y, t, T - show probability density        */
                          /* function plot.                                  */
                          /* Other nonblank - do not show density plots.     */
      showdist=1,         /* 1, y, Y, t, T - show distribution functions plot.*/
                          /* Other nonblank - do not show this plot.         */
      );                  /*-------------------------------------------------*/
```

You specify the dependent variable by using the DEPVAR= option and the independent variables by using the INDVAR= option. You specify ONEVAR=0 if there are two or more independent variables. You specify ONEVAR=1 if there is a single independent variable or if the INDVAR= list includes variables that are derived from a single independent variable (see the example on page 14).

For data that contain a dependent variable Y and independent variables $X_1, \ldots, X_p$, the QPRFIT macro uses the QUANTREG procedure to fit the conditional quantile regression model

$$Q_\tau(y_i|x_{i1}, \ldots, x_{ip}) = \beta_0(\tau) + \beta_1(\tau)x_{i1} + \cdots + \beta_p(\tau)x_{ip} \ , \quad i = 1, \ldots, n$$

for t equally spaced quantile levels: $\tau_1 = 1/(t+1), \tau_2 = 2/(t+1), \ldots, \tau_t = t/(t+1)$. You specify t by using the NODES= option. Estimates for $\boldsymbol{\beta}(\tau_1), \ldots, \boldsymbol{\beta}(\tau_t)$ are saved in an output data set that you can name in the PEDATA= option. The default output data set is named **QPRPE**.

Let $y_{i_1}, \ldots, y_{i_m}$ denote the values of Y for a subset of m observations that you identify in the IDDATA= data set, and let $x_{i_1 1}, \ldots, x_{i_m p}$ denote the corresponding covariate values. For observation i_j, the macro forms the set Q_{i_j} of predicted quantiles. These quantiles are sorted and used to construct a conditional cumulative distribution function (CDF) that corresponds to the covariate values $x_{i_1 1}, \ldots, x_{i_m p}$. When you specify SHOWDIST=1, the macro plots the CDFs that correspond to the covariate values and the predicted quantile levels for the specified observations, which it computes from the CDFs; see Figure 20 for an example. When you specify SHOWPDFS=1, the macro plots smooth density estimates that correspond to the covariate values; see Figure 22 for an example.

REFERENCES

Hao, L., and Naiman, D. Q. (2007). *Quantile Regression*. London: Sage Publications.

Koenker, R. (2005). *Quantile Regression*. New York: Cambridge University Press.

Koenker, R., and Bassett, G. W. (1978). "Regression Quantiles." *Econometrica* 46:33–50.

Koenker, R., and Xiao, Z. (2006). "Quantile Autoregression." *Journal of the American Statistical Association* 101:980–1006.

Koenker, R., and Zhao, Q. (1996). "Conditional Quantile Estimation and Inference for ARCH Models." *Econometric Theory* 12:793–813.

Xiao, Z., Guo, H., and Lam, M. S. (2015). "Quantile Regression and Value at Risk." In *Handbook of Financial Econometrics and Statistics,* edited by C.-F. Lee and J. Lee, 1143–1167. New York: Springer.

Xiao, Z., and Koenker, R. (2009). "Conditional Quantile Estimation for Generalized Autoregressive Conditional Heteroscedasticity Models." *Journal of the American Statistical Association* 104:1696–1712.

Acknowledgments

The authors thank Warren Kuhfeld for assistance with the QPRFIT macro and the graphical displays in this paper. The authors also thank Ed Huddleston for editorial assistance.

Paper SAS2033-2018

Regression Model Building for Large, Complex Data with SAS® Viya® Procedures

Robert N. Rodriguez and Weijie Cai, SAS Institute Inc.

Abstract

Analysts who do statistical modeling, data mining, and machine learning often ask the following question: "I have hundreds of variables—even thousands. Which should I include in my regression model?" This paper describes SAS® Viya® procedures for building linear and logistic regression models, generalized linear models, quantile regression models, generalized additive models, and proportional hazards regression models. The paper explains how these procedures capitalize on the in-memory environment of SAS Viya, and it compares their syntax, features, and output with those of high-performance regression modeling procedures in SAS/STAT® software.

Introduction

High-dimensional data now provide the foundation for many business applications and fields of scientific research. Because these data are increasingly large and complex, they require greater computational power for building regression models that are interpretable or that accurately predict future responses. When interpretability is the goal, you need inferential results, such as standard errors and p-values, to decide which effects are important. When prediction is the goal, you need to evaluate the accuracy of prediction and assess whether it could be improved by a sparser, more parsimonious model.

This paper describes six statistical procedures available in SAS Viya that meet these needs. In addition to fitting models, these procedures provide modern approaches for building models by selecting variables and effects, such as classification effects and spline effects. These approaches rely on penalized least squares and penalized likelihood as theoretical frameworks for variable selection and feature extraction.

The paper is organized into six main sections, one for each procedure:

- "Building Least Squares Regression Models with the REGSELECT Procedure"
- "Building Logistic Regression Models with the LOGSELECT Procedure"
- "Building Generalized Linear Models with the GENSELECT Procedure"
- "Building Quantile Regression Models with the QTRSELECT Procedure"
- "Fitting Generalized Additive Models with the GAMMOD Procedure"
- "Building Proportional Hazards Regression Models with the PHSELECT Procedure"

Each section introduces a procedure by explaining its approach and illustrating its use with a basic example.

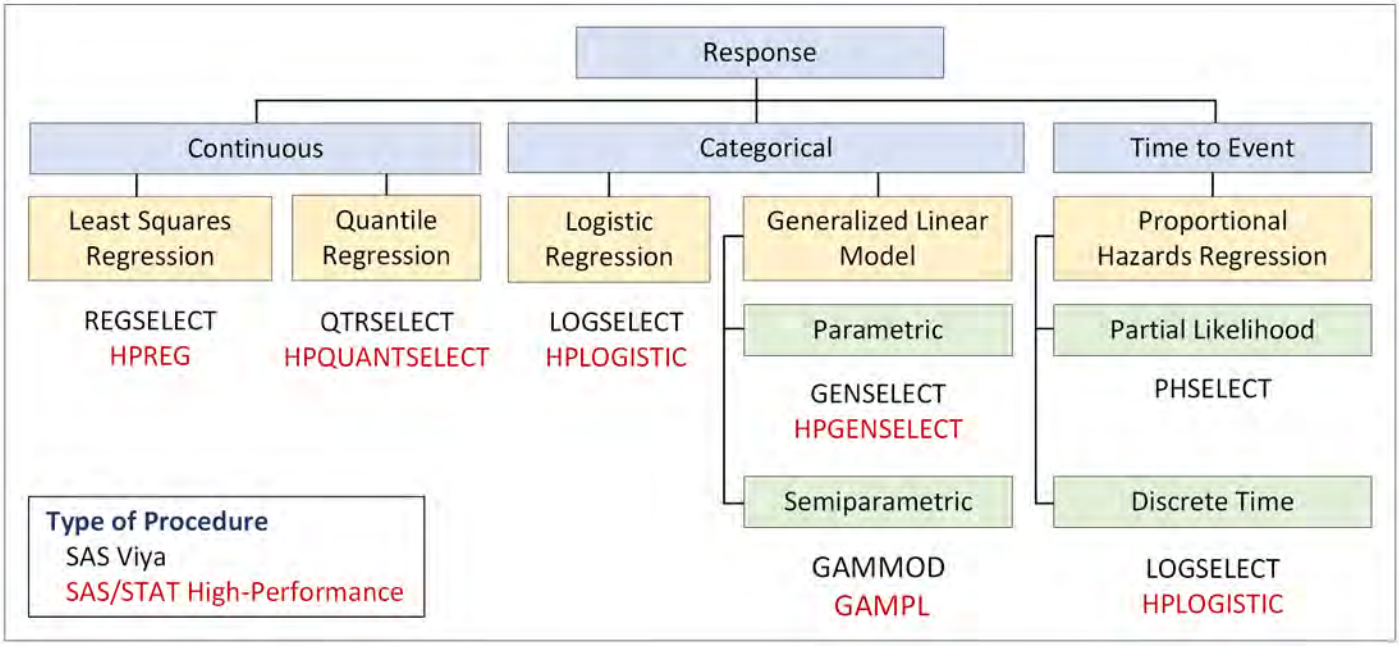
Figure 1 shows the different classes of regression models that are supported by the six SAS Viya procedures. With the exception of the PHSELECT procedure, these procedures are successors to high-performance regression procedures in SAS/STAT® software that have similar functionality, and which are described by Rodriguez (2016).

SAS Viya is the third generation of SAS® software for high-performance in-memory analytics, and the analytic engine in SAS Viya is SAS® Cloud Analytic Services (CAS). Because the SAS Viya statistical procedures were developed specifically for CAS, they enable you to do the following:

- run on a cluster of machines that distribute the data and the computations
- run in single-machine mode
- exploit all the available cores and concurrent threads

These procedures operate only on in-memory CAS tables, and you must license SAS® Visual Statistics to run them. If you also have SAS® 9.4M5 installed, you can run procedures in both SAS Viya and SAS 9.4M5 from the same SAS interface, such as the SAS windowing environment, SAS® Enterprise Guide®, and SAS® Studio. (Note that if you have only licensed SAS Visual Statistics, SAS/STAT procedures are included and you can access them through SAS Studio.)

Figure 1 SAS Viya Procedures and SAS/STAT High-Performance Procedures for Regression Modeling



Building Least Squares Regression Models with the REGSELECT Procedure

The REGSELECT procedure fits and builds general linear models of the form

$$y_i = \beta_0 + \beta_1 x_{i1} + \cdots + \beta_p x_{ip} + \epsilon_i, \quad i = 1, \ldots, n$$

where the response y_i is continuous and the predictors $x_{i1}, \ldots, x_{ip}$ represent main effects that consist of continuous or classification variables and can include interaction effects or constructed effects of these variables.

With too many predictors, the model can overfit the training data, leading to poor prediction with future data. To deal with this problem, the REGSELECT procedure supports the model selection methods summarized in Table 1.

Table 1 Effect Selection Methods in the REGSELECT Procedure

Method	Description
Forward selection	Starts with no effects in the model and adds effects
Forward swap	Before adding an effect, makes all pairwise swaps of in-model and out-of-model effects that improve the selection criterion
Backward elimination	Starts with all effects in the model and deletes effects
Stepwise selection	Starts with no effects in the model and adds or deletes effects
Least angle regression	Starts with no effects and adds effects; at each step, $\hat{\beta}$s shrink toward zero
Lasso	Constrains the sum of absolute $\hat{\beta}$s; some $\hat{\beta}$s are set to zero, others shrink toward zero

For each method, PROC REGSELECT supports modern model evaluation criteria for selection and stopping, which penalize large numbers of parameters in a principled manner. The procedure also supports stopping and selection based on external validation and leave-one-out cross validation. In practice, no single method consistently outperforms the rest, but—depending on your goal—an informed and judicious choice of these features can lead to models that have greater predictive accuracy or models that are more interpretable.

The first four methods in Table 1 estimate the regression coefficients for candidate models by solving the following least squares problem:

$$\min_{\beta_0,\ldots,\beta_p} \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j \right)^2$$

In contrast, the lasso method places an ℓ_1 penalty on the coefficients:

$$\min_{\beta_0,\ldots,\beta_p} \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j \right)^2$$

$$\text{subject to } \sum_{j=1}^{p} \left|\beta_j\right| \leq t$$

For large values of t, the lasso method produces ordinary least squares estimates. Decreasing t in discrete steps leads to a sequence of coefficient estimates, where some are exactly zero and the rest, which correspond to selected effects, are shrunk toward zero. This mitigates the influence of selected effects that do not belong in the model, and it distinguishes the lasso method from methods such as forward selection, as illustrated in the following example.

Example: Predicting the Mean Close Rate for Retail Stores

The close rate for a retail store is the percentage of shoppers who enter the store and make a purchase. Understanding what factors predict the mean close rate is critical to the profitability and growth of large retail companies, and a regression model is constructed to study this question.

The close rates for 500 stores are saved in a CAS table named **Stores**. Each observation provides information about a store. The variables available for the model are the response **Close_Rate** and the following candidate predictors:

- **X1**, . . . , **X20**, which measure 20 general characteristics of stores, such as floor size and number of employees
- **P1**, . . . , **P6**, which measure six promotional activities, such as advertising and sales
- **L1**, . . . , **L6**, which measure special layouts of items in six departments

In practice, close rate data can involve hundreds of candidate predictors. A small set is used here for illustration.

Results with the Forward Selection Method

The following statements use the forward selection method in the REGSELECT procedure to build a model:

```
ods graphics on;
proc regselect data=mycas.Stores;
   model Close_Rate = X1-X20 L1-L6 P1-P6;
   selection method=forward plots=all;
run;
```

The DATA= option specifies a CAS table named **mycas.Stores**. The first level of the name is the CAS engine libref, and the second level is the table name. The SELECTION statement requests model selection. The settings for the selection process are listed in Figure 2.

Figure 2 Selection Information

The REGSELECT Procedure

Selection Information	
Selection Method	Forward
Select Criterion	SBC
Stop Criterion	SBC
Effect Hierarchy Enforced	None
Stop Horizon	3

By default, the REGSELECT procedure uses the Schwarz Bayesian criterion (SBC) as the selection criterion for determining the order in which effects enter at each step. The effect that is selected is the one whose addition maximizes the decrease in SBC. By default, the procedure also uses SBC as the stop criterion. Selection stops at the step where the next step yields a model that has a larger SBC value. The stop horizon, which is 3 by default, specifies the number of consecutive steps at which the stop criterion must decrease in order to detect a minimum.

As shown in Figure 3, the minimum value of SBC is reached at Step 9, when **P1** enters the model.

Figure 3 Selection Summary with Forward Selection

The REGSELECT Procedure

Selection Details

Selection Summary

Step	Effect Entered	Number Effects In	SBC
0	**Intercept**	1	47.8155
1	**X2**	2	-27.1875
2	**X4**	3	-52.4996
3	**P3**	4	-60.6381
4	**P4**	5	-67.4347
5	**L1**	6	-73.7232
6	**L3**	7	-79.3681
7	**P5**	8	-83.1847
8	**L2**	9	-86.2457
9	**P1**	10	-88.6068*
10	**L5**	11	-87.8923
11	**P2**	12	-84.7692

*** Optimal Value Of Criterion**

The coefficient progression plot in Figure 4, requested by the PLOTS= option, visualizes the selection process.

Figure 4 Coefficient Progression with Forward Selection

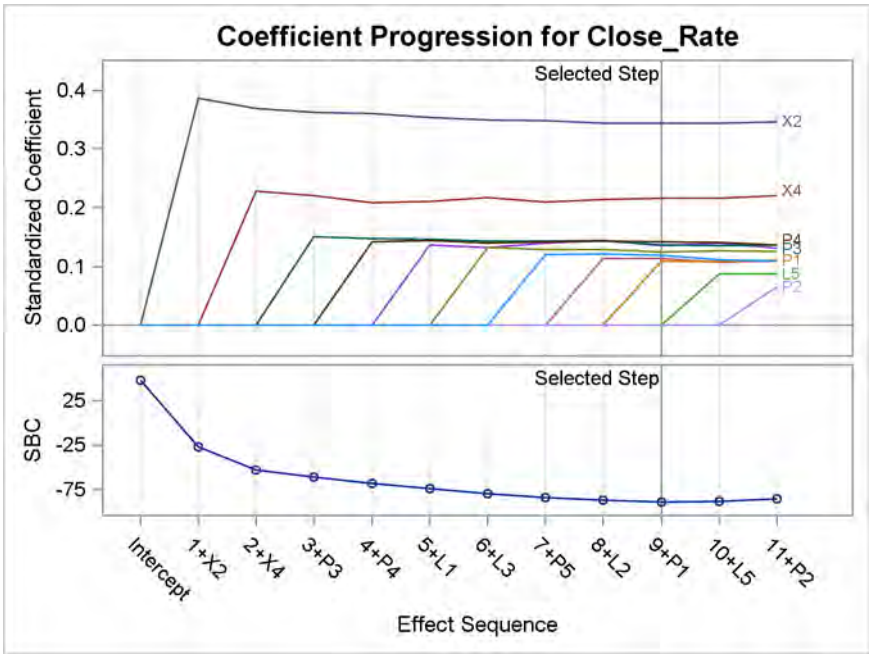


Figure 5 shows the parameter estimates for the final model. The estimates for **X2** and **X4** are larger than the estimates for the seven other predictors, and all the standard errors are comparable. The p-values should be interpreted with care because they are computed conditionally on the final selected model and do not take into account the process by which the model was selected.

Figure 5 Parameter Estimates for Model Selected with Forward Selection

Parameter Estimates

Parameter	DF	Estimate	Standard Error	t Value	Pr > |t|
Intercept	1	60.412202	0.119136	507.09	<.0001
X2	1	1.225952	0.133595	9.18	<.0001
X4	1	0.798252	0.138799	5.75	<.0001
L1	1	0.496037	0.137290	3.61	0.0003
L2	1	0.379632	0.125270	3.03	0.0026
L3	1	0.438092	0.131785	3.32	0.0010
P1	1	0.400154	0.137440	2.91	0.0038
P3	1	0.479429	0.131241	3.65	0.0003
P4	1	0.520183	0.136973	3.80	0.0002
P5	1	0.420284	0.132103	3.18	0.0016

Results with the Lasso Method

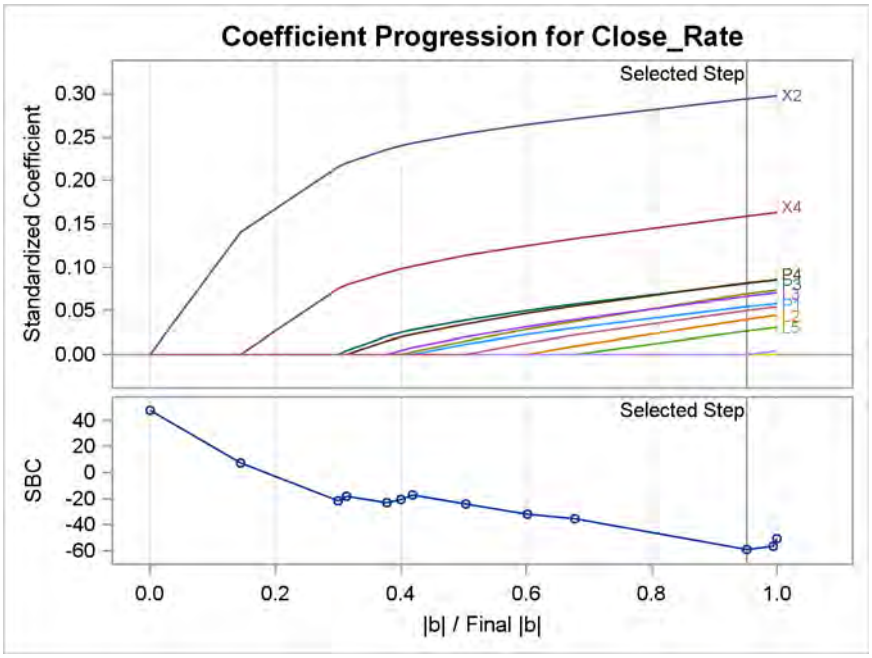
The following statements use the lasso method to build a model:

```
proc regselect data=mycas.Stores;
   model Close_Rate = X1-X20 L1-L6 P1-P6;
   selection method=lasso plots(stepaxis=normb)=all;
run;
```

For the lasso method, the REGSELECT procedure uses the least angle regression algorithm, introduced by Efron et al. (2004), to produce a sequence of regression models in which one parameter is added at each step. By default, the selection criterion is SBC, the stop criterion is SBC, and the stop horizon is 3.

The lasso method selects a model that has 10 variables when the minimum value of SBC is reached at Step 10, as shown in Figure 6. The stop horizon enables the small local minima of SBC at Steps 3 and 5 to be ignored.

Figure 6 Coefficient Progression with Lasso Method



The scale for the horizontal axis, requested by the STEPAXIS= suboption, is more appropriate for the lasso method than the default step scale in Figure 4 because it expresses the size of the ith step as the ℓ_1 norm of the parameters relative to the ℓ_1 norm of the parameters at the final step.

The parameter estimates for the final model are shown in Figure 7. These estimates are closer to zero than the corresponding estimates in Figure 5.

Figure 7 Parameter Estimates for Model Selected with Lasso Method

Parameter Estimates		
Parameter	**DF**	**Estimate**
Intercept	1	60.720592
X2	1	1.046158
X4	1	0.584167
L1	1	0.254223
L2	1	0.133271
L3	1	0.235543
L5	1	0.095443
P1	1	0.199461
P3	1	0.285626
P4	1	0.298742
P5	1	0.176449

Comparison with the HPREG and GLMSELECT Procedures

The functionality of the REGSELECT procedure closely resembles that of the HPREG procedure, which is a SAS/STAT high-performance procedure. Both procedures perform model selection for ordinary least squares regression models, which you can specify as general linear models. You request model selection by using the SELECTION statement.

The functionality of the REGSELECT procedure also resembles that of the GLMSELECT procedure in SAS/STAT, which is multithreaded. Both procedures offer multiple methods of effect selection, the ability to use external validation data and cross validation as selection criteria, and extensive options to customize the selection process. Both procedures provide the ability to specify constructed effects in the EFFECT statement. The preceding example produces the same results when run with the GLMSELECT procedure (Rodriguez 2016), provided that you specify a stop horizon of 1. The default stop horizon in the REGSELECT procedure is 3 because it provides better protection against small local minima in the stop criterion.

Building Logistic Regression Models with the LOGSELECT Procedure

The LOGSELECT procedure fits and builds binary response models of the form

$$g(\pi_i) = \beta_0 + \beta_1 x_{i1} + \cdots + \beta_p x_{ip} + \epsilon_i, \quad i = 1, \ldots, n$$

where π_i is the predicted probability of an event and the link function g is the logit, probit, log-log, or complementary log-log function. As in models supported by the REGSELECT procedure, the predictors represent main effects that consist of continuous or classification variables and can include interaction effects or constructed effects of these variables. When the response has more than two values, the LOGSELECT procedure fits and builds ordinal response models and generalized logit models.

With too many predictors, the model can overfit the training data, leading to poor prediction with future data. To deal with this problem, the LOGSELECT procedure supports the selection methods summarized in Table 2.

Table 2 Effect Selection Methods in the LOGSELECT Procedure

Method	Description
Forward selection	Starts with no effects in the model and adds effects
Backward elimination	Starts with all effects in the model and deletes effects
Backward (fast)	Starts with all effects in the model and deletes effects without refitting the model
Stepwise selection	Starts with no effects in the model and adds or deletes effects
Lasso	Constrains the sum of absolute $\hat{\beta}$s; some $\hat{\beta}$s are set to zero, others shrink toward zero

Example: Predicting High Customer Satisfaction for Retail Stores

The CAS table **Stores** in the previous example contains a binary response variable named **HighSatisfaction**, which is equal to 1 if a store achieved the highest level of satisfaction in a customer survey and is equal to 0 otherwise. The following statements use the LOGSELECT procedure to build a logistic regression model for predicting high satisfaction:

```
proc logselect data=mycas.Stores;
   model HighSatisfaction(event='1') = X1-X20 L1-L6 P1-P6;
   selection method=forward(choose=validate) plots=all;
   partition fraction(validate=0.25 seed=14591);
   code file='HighPredict.sas';
run;
```

The PARTITION statement partitions the observations into disjoint subsets for model training (75%) and model validation (25%). The SELECTION statement requests model selection based on the forward method. At each step, the training data are used to fit the candidate model. The CHOOSE=VALIDATE suboption requests that the average square error (ASE) be computed on the validation data for the model at each step of the selection process. The selected model is the smallest model at any step that yields the lowest ASE. The CODE statement writes SAS DATA step code for predicting high satisfaction to a file named *HighPredict.sas*.

Figure 8 shows the number of observations at each level of the response in each of the partitions.

Figure 8 Partition Counts

The LOGSELECT Procedure

Response Profile

Ordered Value	HighSatisfaction	Total Frequency	Training	Validation
1	0	379	293	86
2	1	121	90	31

Probability modeled is HighSatisfaction = 1.

As shown in Figure 9, the minimum validation ASE is reached at Step 5, when **X4** enters the model.

Figure 9 Coefficient Progression with Forward Selection

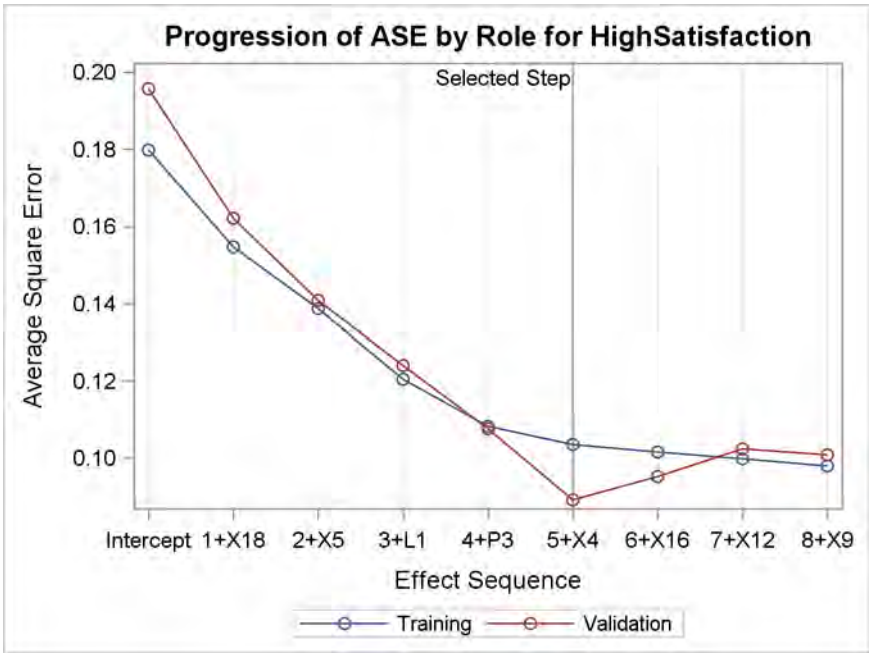


Figure 10 displays the fit statistics for the selected model, which are computed for both the training and validation data. The ASE, misclassification rate, and difference of means are comparable for the two groups of data, indicating a good predictive fit.

Figure 10 Fit Statistics for Training and Validation Partitions

Fit Statistics		
Description	**Training**	**Validation**
-2 Log Likelihood	247.84673	65.04619
AIC (smaller is better)	259.84673	77.04619
AICC (smaller is better)	260.07014	77.80983
SBC (smaller is better)	283.53494	93.61923
Average Square Error	0.10347	0.08912
-2 Log L (Intercept-only)	417.64791	135.29376
R-Square	0.35811	0.45141
Max-rescaled R-Square	0.53938	0.65864
McFadden's R-Square	0.40657	0.51922
Misclassification Rate	0.14621	0.14530
Difference of Means	0.42865	0.51168

Figure 11 shows the parameter estimates for the selected model.

Figure 11 Parameter Estimates for Selected Model

Parameter Estimates					
Parameter	**DF**	**Estimate**	**Standard Error**	**Chi-Square**	**Pr > ChiSq**
Intercept	1	-4.063480	0.494970	67.3967	<.0001
X4	1	1.955431	0.588277	11.0489	0.0009
X5	1	-4.346695	0.689653	39.7243	<.0001
X18	1	-5.018940	0.705420	50.6207	<.0001
L1	1	3.916054	0.677884	33.3723	<.0001
P3	1	-3.249121	0.634598	26.2140	<.0001

Comparison with the HPLOGISTIC Procedure

The functionality of the LOGSELECT procedure closely resembles that of the HPLOGISTIC procedure, which is a SAS/STAT high-performance procedure. In addition to the selection methods available in the HPLOGISTIC procedure, the LOGSELECT procedure provides the lasso method. The LOGSELECT procedure also produces selection plots and constructs complex effects, such as univariate spline effects and polynomial effects—features not available in the HPLOGISTIC procedure.

Building Generalized Linear Models with the GENSELECT Procedure

The GENSELECT procedure fits and builds generalized linear models, which can analyze many types of responses. A generalized linear model consists of three components:

- A linear predictor, which is defined in the same way as for general linear models:

$$\eta_i = \beta_0 + \beta_1 x_{i1} + \cdots + \beta_p x_{ip}, \quad i = 1, \ldots, n$$

- A specified link function g, which describes how μ_i, the expected value of y_i, is related to η_i:

$$g(\mu_i) = \eta_i = \beta_0 + \beta_1 x_{i1} + \cdots + \beta_p x_{ip}$$

- An assumed distribution for the responses y_i. For distributions in the exponential family, the variance of the response depends on the mean μ through a variance function V,

$$\mathrm{Var}(y_i) = \frac{\phi V(\mu_i)}{w_i}$$

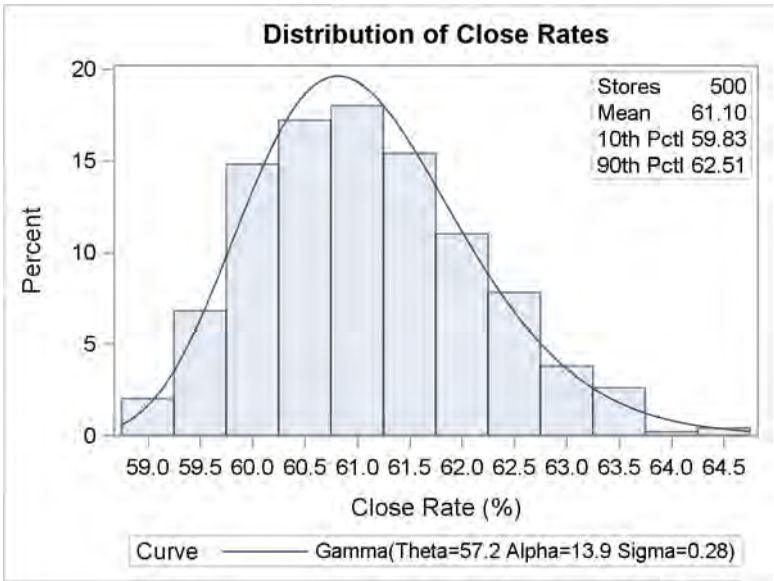
where ϕ is a constant and w_i is a known weight for each observation. The dispersion parameter ϕ is either estimated or known (for example, $\phi = 1$ for the binomial distribution).

The GENSELECT procedure supports standard response distributions in the exponential family, such as the normal, Poisson, and Tweedie distributions. In addition, the procedure supports ordinal and unordered multinomial response distributions. For all these distributions, the GENSELECT procedure estimates model parameters by using maximum likelihood techniques. To deal with the problem of overfitting, the procedure provides the forward, backward, fast backward, stepwise, and lasso methods of effect selection.

Example: Predicting the Mean Close Rate for Retail Stores (continued)

Figure 12 shows the marginal distribution of the close rates in **Stores**. A gamma distribution provides a good fit, suggesting that a gamma regression model for the conditional mean of close rate might improve on the model that was obtained with the REGSELECT procedure in the example on page 3.

Figure 12 Distribution of Close Rates for 500 Stores



The following statements use the GENSELECT procedure to build a gamma regression model. A preliminary shift transformation is applied to **Close_Rate** because the gamma distribution has a threshold at 0.

```
data mycas.Stores; set mycas.Stores;
   Close_Rate_0 = Close_Rate - 58;
run;

proc genselect data=mycas.Stores;
   model Close_Rate_0 = X1-X20 L1-L6 P1-P6 / distribution=gamma link=log;
   selection method=forward;
run;
```

The METHOD= option requests the forward selection method. The default criterion for choosing the model at each step is SBC, which is also the default stop criterion. The default stop horizon is 3.

Figure 13 shows that the minimum SBC value is reached at Step 9, when **P1** enters the model. The selected variables happen to be the same as those selected by the REGSELECT procedure when it uses the forward method, as shown in Figure 3. However, an additional dispersion parameter is estimated for the gamma regression model.

Figure 13 Selection Summary with Forward Method

The GENSELECT Procedure

Selection Details

Selection Summary

Step	Effect Entered	Number Effects In	SBC
0	Intercept	1	1456.6448
1	X2	2	1391.4830
2	X4	3	1359.0476
3	P3	4	1348.9659
4	P4	5	1341.2893
5	L3	6	1334.9530
6	L1	7	1330.0412
7	P5	8	1325.6188
8	L2	9	1322.8629
9	P1	10	1320.4778*
10	L5	11	1322.7962
11	X3	12	1325.2128
12	P2	13	1327.5805

* Optimal Value Of Criterion

Figure 14 shows the parameter estimates for the selected model. As in Figure 5, the estimates for **X2** and **X4** are larger in magnitude than the estimates for the other predictors.

Figure 14 Parameter Estimates for Gamma Regression Model Selected with Forward Method

Parameter Estimates

Parameter	DF	Estimate	Standard Error	Chi-Square	Pr > ChiSq
Intercept	1	0.882152	0.039605	496.1157	<.0001
X2	1	0.412054	0.044286	86.5731	<.0001
X4	1	0.273341	0.046033	35.2594	<.0001
L1	1	0.161623	0.045409	12.6684	0.0004
L2	1	0.124663	0.041233	9.1406	0.0025
L3	1	0.153185	0.043612	12.3371	0.0004
P1	1	0.132601	0.045129	8.6334	0.0033
P3	1	0.168153	0.043051	15.2564	<.0001
P4	1	0.183079	0.045080	16.4931	<.0001
P5	1	0.142619	0.043513	10.7428	0.0010
Dispersion	1	12.194522	0.760933		

Comparison with the HPGENSELECT and GENMOD Procedures

The functionality of the GENSELECT procedure resembles that of the SAS/STAT high-performance HPGENSELECT procedure. The GENSELECT procedure is additionally capable of constructing complex effects, such as univariate spline and polynomial expansions. The HPGENSELECT procedure (but not the GENSELECT procedure) provides models for zero-inflated data. The GENSELECT procedure uses the log link function as the default for both the gamma and the inverse Gaussian distributions. The HPGENSELECT procedure uses the reciprocal link function as the default for the gamma distribution, and it uses the reciprocal squared link function as the default for the inverse Gaussian distribution.

The GENSELECT procedure, the HPGENSELECT procedure, and the GENMOD procedure in SAS/STAT fit generalized linear models. However, there are important design differences in their capabilities, as summarized in Table 3.

Table 3 Comparison of the GENSELECT, HPGENSELECT, and GENMOD Procedures

GENSELECT and HPGENSELECT Procedures	GENMOD Procedure
Fit and build generalized linear models	Fits generalized linear models
Analyze large to massive data	Analyzes moderate to large data
Designed for predictive modeling	Designed for inferential analysis
Run in single-machine or distributed mode	Runs in single-machine mode
Use all cores and concurrent threads	Is single threaded

Building Quantile Regression Models with the QTRSELECT Procedure

The QTRSELECT procedure fits and builds quantile regression models, which predict the quantiles (or equivalently, the percentiles) of a continuous response variable. The quantile regression model for the τth quantile (100τth percentile) is of the form

$$Q_\tau(y_i) = \beta_0(\tau) + \beta_1(\tau)x_{i1} + \cdots + \beta_p(\tau)x_{ip}, \quad i = 1, \ldots, n, \quad 0 < \tau < 1$$

where the predictors $x_{i1}, \ldots, x_{ip}$ represent main effects that consist of continuous or classification variables and can include interaction effects or constructed effects of these variables. The regression coefficients $\beta_j(\tau)$ are estimated by solving the minimization problem

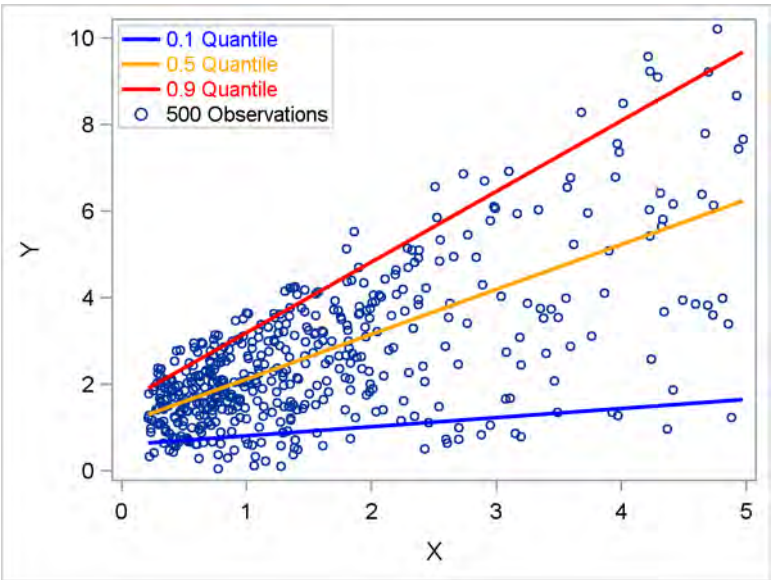
$$\min_{\beta_0,\ldots,\beta_p} \sum_{i=1}^{n} \rho_\tau \left(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j \right)$$

where $\rho_\tau(r) = \tau \max(r, 0) + (1-\tau)\max(-r, 0)$. The function $\rho_\tau(r)$ is referred to as the check loss function. To avoid overfitting, the QTRSELECT procedure provides the forward, backward, and stepwise methods of effect selection.

Quantile regression was introduced 40 years ago by Koenker and Bassett (1978), but only recently—because of computational advances—has it become practical for large data. With sufficient data, quantile regression can potentially describe the entire conditional distribution of the response. General linear models and generalized linear models are computationally less expensive, but the only aspect of the conditional distribution that they describe is the mean.

You should consider using quantile regression when the conditional distribution of the response varies with the predictors. Figure 15 illustrates data in which the conditional variance of the response (Var$[Y|X]$) increases with X.

Figure 15 Quantile Regression Models for Heteroscedastic Data



The three lines in Figure 15 represent quantile regression models for Y that correspond to the quantile levels 0.1, 0.5, and 0.9, or equivalently the 10th, 50th, and 90th percentiles. Fitting such models for a more extensive grid of quantile levels yields a description of the entire conditional distribution.

Table 4 summarizes the differences between least squares regression and quantile regression.

Table 4 Quantile Regression Compared with Least Squares Regression

Least Squares Regression	Quantile Regression
Predicts the conditional mean	Predicts conditional quantiles
Often assumes normality	Assumes no parametric distribution
Is sensitive to outliers	Is robust to outliers
Applies even with a small number of observations	Needs sufficient data
Is computationally inexpensive	Is computationally intensive

In many fields, such as financial risk management and fraud detection, important questions can be answered by modeling extreme percentiles of critical factors. Quantile regression can yield valuable insights that would not be readily obtained with standard regression methods. This is illustrated in the next example.

Example: Predicting Low and High Percentiles of Close Rates for Retail Stores

The examples on page 3 and page 9 show how you can use the REGSELECT and GENSELECT procedures to predict the average close rate for a store. This example shows how you can use the QTRSELECT procedure to predict low and high percentiles of close rates. Here the close rate for a store is considered to be low if it is less than the 10th percentile for stores that have the same combination of predictor values. Likewise, a close rate is considered to be high if it is greater than the 90th percentile for stores that have the same combination of predictor values.

The following statements use the QTRSELECT procedure to build regression models that predict the 10th and 90th percentiles, which correspond to the quantile levels 0.1 and 0.9:

```
proc qtrselect data=mycas.Stores;
   model Close_Rate = X1-X20 L1-L6 P1-P6 / quantile=0.1 0.9;
   selection method=forward(choose=aic) stophorizon=1 plots=all;
run;
```

Figure 16 visualizes the forward selection process for quantile level 0.1.

Figure 16 Coefficient Progression for Quantile Level 0.1

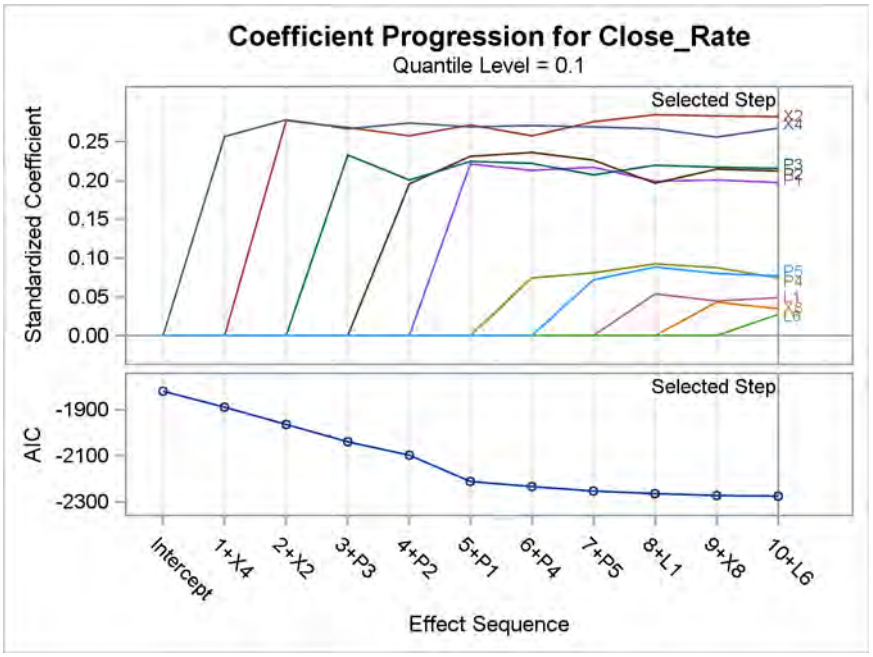


Figure 17 shows the fit statistics for the final model for quantile level 0.1.

Figure 17 Fit Statistics for Model Selected for Quantile Level 0.1

The QTRSELECT Procedure

Quantile Level = 0.1
Selected Model

Objective Function	50.27587
R1	0.37774
Adj R1	0.36501
AIC	-2275.08286
AICC	-2274.54187
SBC	-2228.72217
ACL	0.10055

Figure 18 shows the parameter estimates for the final model for quantile level 0.1.

Figure 18 Parameter Estimates for Model Selected for Quantile Level 0.1

Parameter Estimates

Parameter	DF	Estimate	Standard Error	t Value	Pr > |t|
Intercept	1	59.98266	0.01989	3015.08	<.0001
X2	1	1.00585	0.02599	38.71	<.0001
X4	1	0.98467	0.02765	35.62	<.0001
X8	1	0.12872	0.02640	4.88	<.0001
L1	1	0.17769	0.02703	6.57	<.0001
L6	1	0.09756	0.02415	4.04	<.0001
P1	1	0.72146	0.02530	28.52	<.0001
P2	1	0.76697	0.02626	29.21	<.0001
P3	1	0.75435	0.02601	29.00	<.0001
P4	1	0.27254	0.02725	10.00	<.0001
P5	1	0.27256	0.02350	11.60	<.0001

The QTRSELECT procedure produces a distinct set of results for quantile level 0.9 because the corresponding check loss function is different from the check loss function for quantile level 0.1. Figure 19 shows the parameter estimates for the final model for quantile level 0.9.

Figure 19 Parameter Estimates for Model Selected for Quantile Level 0.9

Parameter Estimates

Parameter	DF	Estimate	Standard Error	t Value	Pr > |t|
Intercept	1	60.26032	0.10430	577.77	<.0001
X2	1	1.30182	0.11886	10.95	<.0001
X4	1	0.81080	0.11156	7.27	<.0001
X14	1	-0.33776	0.11109	-3.04	0.0025
L1	1	0.61383	0.11502	5.34	<.0001
L2	1	0.91291	0.10822	8.44	<.0001
L3	1	0.93599	0.11590	8.08	<.0001
L4	1	0.63198	0.10882	5.81	<.0001
L5	1	0.61580	0.11252	5.47	<.0001
L6	1	0.51865	0.10089	5.14	<.0001
P4	1	0.70377	0.12686	5.55	<.0001

Two of the store characteristic variables (**X2** and **X4**) are selected in both the model for quantile level 0.1 and the model for quantile level 0.9. Five of the promotion variables (**P1–P5**) are selected in the model for level 0.1, but only one (**P4**) is selected in the model for level 0.9. All the layout variables (**L1–L6**) are selected in the model for level 0.9, but only two (**L1** and **L6**) are selected in the model for level 0.1. These results give you information about low- and high-performing stores that you would not obtain directly from least squares regression.

Comparison with the HPQUANTSELECT and QUANTSELECT Procedures

The functionality of the QTRSELECT procedure closely resembles that of the HPQUANTSELECT procedure, which is a SAS/STAT high-performance procedure. The QTRSELECT procedure is additionally capable of constructing complex effects (such as univariate spline and polynomial expansions) and producing plots that visualize the effect selection process.

Both the QTRSELECT procedure and the QUANTSELECT procedure in SAS/STAT fit and perform model selection for quantile regression models. The QTRSELECT procedure (but not the QUANTSELECT procedure) provides confidence limits and Wald tests for parameters and prediction limits for quantiles. The QUANTSELECT procedure (but not the QTRSELECT procedure) provides the lasso and adaptive lasso effect-selection methods and effect selection for quantile process regression. See Rodriguez and Yao (2017) for an analysis of the close rate data that uses the QUANTSELECT procedure.

Fitting Generalized Additive Models with the GAMMOD Procedure

The GAMMOD procedure fits generalized additive models that are based on low-rank regression splines (Wood 2006). Generalized additive models are extensions of generalized linear models. In addition to allowing linear predictors, they allow spline terms in order to capture nonlinear dependency that is either unknown or too complex to be characterized with a parametric effect such as a linear or quadratic term. Table 5 summarizes the components of a generalized additive model.

Table 5 Components of Generalized Additive Models

Component	Description
Linear predictor	Effects that involve continuous or classification variables
Nonparametric predictor	Spline terms that involve one or more continuous variables
Link function	Log, logit, log-log, complementary log-log, probit, reciprocal, reciprocal square
Distribution	Binary, binomial, gamma, inverse Gaussian, negative binomial, normal, Poisson, Tweedie

The GAMMOD procedure constructs spline terms by using the thin-plate regression spline technique (Wood 2003). A roughness penalty is applied to each spline term by a smoothing parameter that controls the balance between goodness of fit and roughness of the spline curve.

Unlike the other procedures discussed in this paper, the GAMMOD procedure does not select variables or effects. Instead, it finds optimal models by automatically selecting smoothing parameters based on global model-evaluation criteria such as generalized cross validation (GCV) and unbiased risk estimation (UBRE).

Generalized additive models are useful for problems that involve unknown—possibly nonlinear—relationships between the response and the predictors, and relationships that can be assumed to be linear. Frigo and Osterloo (2016) describe a problem of this type in the context of insurance pricing.

In some situations, the spline fits that you obtain using PROC GAMMOD suggest parametric effects in a model that you can then fit with the GENSELECT procedure, as illustrated in the following example.

Example: Predicting Claim Rates for Loans

This example is drawn from the mortgage insurance industry, where analysts create models to predict conditional claim rates for specific types of loans. Understanding how claim rates depend on predictors is critical, because the model is used to assess risk and allocate funds for potential claims.

Claim rates for 10,000 mortgages are saved in a CAS table named **Claims**. The response variable **Rate** is the number of claims per 10,000 contracts in a policy year, and it is assumed to follow a Poisson distribution whose mean depends on the predictors listed in Table 6.

Table 6 Predictors for Claim Rate

Predictor	Description	Contribution
Age	Age of loan	Unknown, possibly quadratic
Price	Price of house	Unknown, nonlinear
RefInd	Indicator of a refinanced loan	Linear
PayIncmRatio	Payment-to-income ratio	Linear
RefInctvRatio	Refinance incentive ratio	Linear
UnempRate	Unemployment rate	Linear

In practice, models of this type involve many more predictors. A subset is used here for illustration.

The following statements use the GAMMOD procedure to fit an additive Poisson regression model for **Rate**:

```
proc gammod data=mycas.Claims plots=components;
   class RefInd;
   model Rate = param(RefInd PayIncmRatio RefInctvRatio UnempRate)
                spline(Age) spline(Price) / dist=poisson;
run;
```

The PARAM option requests parametric linear terms for **RefInd**, **PayIncmRatio**, **RefInctvRatio**, and **UnempRate**. The SPLINE options request spline effects for **Age** and **Price**.

Figure 20 displays information about the model fitting process. The Poisson mean of **Rate** is modeled by a log link function. The performance iteration algorithm (Gu and Wahba 1991) is used to obtain optimal smoothing parameters for the spline effects. The unbiased risk estimator (UBRE) criterion is used for model evaluation during the process of selecting smoothing parameters for the spline effects.

Figure 20 Model Information

The GAMMOD Procedure

Model Information	
Data Source	CLAIMS
Response Variable	Rate
Distribution	Poisson
Link Function	Log
Fitting Method	Performance Iteration
Fitting Criterion	UBRE
Optimization Technique for Smoothing	Newton-Raphson
Random Number Seed	1494796320

Figure 21 shows the fit statistics. You can use effective degrees of freedom to compare generalized additive models with generalized linear models, which do not involve spline terms. You can also use the information criteria, AIC, AICC, and BIC, for model comparisons, and you can request either the GCV criterion or the UBRE criterion for comparisons with other generalized additive models or penalized models.

Figure 21 Fit Statistics from the GAMMOD Procedure

Fit Statistics	
Penalized Log Likelihood	-26776
Roughness Penalty	7.83929
Effective Degrees of Freedom	16.54638
Effective Degrees of Freedom for Error	9982.63859
AIC (smaller is better)	53577
AICC (smaller is better)	53577
BIC (smaller is better)	53697
UBRE (smaller is better)	-0.00359

Figure 22 shows estimates for the parametric effects in the model.

Figure 22 Estimates for Parametric Terms

Parameter Estimates

Parameter	DF	Estimate	Standard Error	Chi-Square	Pr > ChiSq
Intercept	1	2.484732	0.020876	14166.0978	<.0001
RefInd 0	1	-0.008894	0.005571	2.5488	0.1104
RefInd 1	0	0	.	.	.
PayIncmRatio	1	0.035731	0.009740	13.4582	0.0002
RefInctvRatio	1	-0.031308	0.009627	10.5765	0.0011
UnempRate	1	0.008047	0.002764	8.4763	0.0036

Figure 23 shows the effective degrees of freedom (DF) for the smoothing components of the model. The component for **Age** has a lower DF, indicating a more linear contribution than the contribution of the component for **Price**.

Figure 23 Estimates for Smoothing Components

Estimates for Smoothing Components

Component	Effective DF	Smoothing Parameter	Roughness Penalty	Number of Parameters	Rank of Penalty Matrix	Number of Knots
Spline(Age)	3.54638	35807.3	7.8393	9	10	24
Spline(Price)	8.00000	1.0000	1.3E-6	9	10	2000

Figure 24 shows tests for the existence of a contribution for each smoothing component. The results should be interpreted with caution because the tests do not take into account the process of selecting the smoothing parameter.

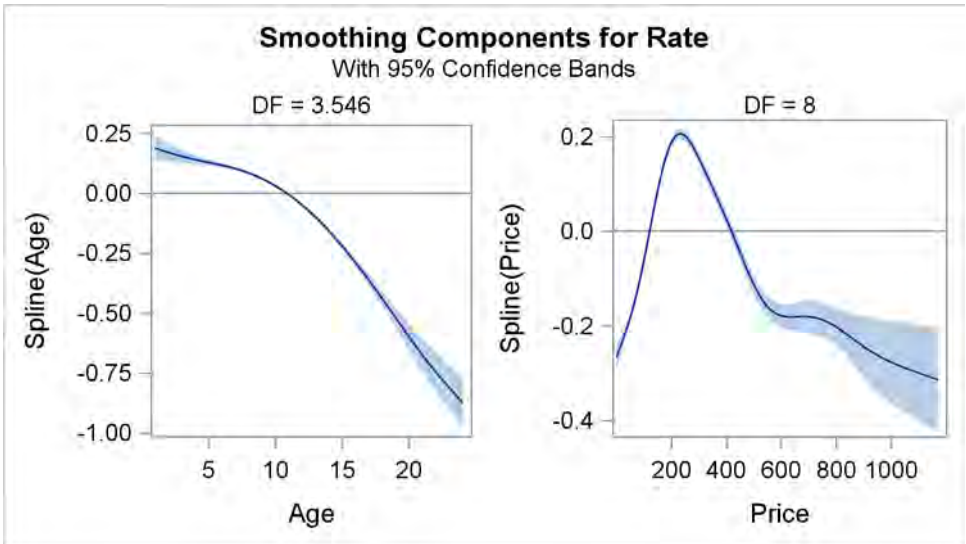
Figure 24 Tests for Smoothing Components

Tests for Smoothing Components

Component	Effective DF	Effective DF for Test	Chi-Square	Pr > ChiSq
Spline(Age)	3.54638	5	1685.6996	<.0001
Spline(Price)	8.00000	8	2844.2140	<.0001

Figure 25 displays plots of the components for **Age** and **Price**.

Figure 25 Spline Components for Age and Price



The plots suggest that quadratic effects might characterize the nonlinearity in **Age** and **Price**. The following statements incorporate these effects in a generalized linear model that is fitted with the GENSELECT procedure (you could also use the GENMOD procedure):

```
proc genselect data=mycas.Claims;
   class RefInd;
   model Rate = RefInd PayIncmRatio RefInctvRatio UnempRate
                Age Age*Age Price Price*Price / dist=poisson link=log;
run;
```

Fit statistics for the model that is fitted with PROC GENSELECT are shown in Figure 26.

Figure 26 Fit Statistics from the GENSELECT Procedure

The GENSELECT Procedure

Fit Statistics	
-2 Log Likelihood	54754
AIC (smaller is better)	54772
AICC (smaller is better)	54772
SBC (smaller is better)	54837

The SBC statistic is also referred to as the BIC statistic. The AIC, AICC, and BIC statistics in Figure 21 are smaller than the corresponding statistics in Figure 26, indicating that the generalized additive model produces a better fit.

Comparison with the GAMPL and GAM Procedures

The GAMMOD procedure resembles the GAMPL and GAM procedures in SAS/STAT; all three procedures fit generalized additive models. The results of the GAMMOD and GAMPL procedures should be very similar, but in general you should not expect similar results between these two procedures and the GAM procedure. Table 7 summarizes important design differences in these procedures.

Table 7 Comparison of the GAMMOD, GAMPL, and GAM Procedures

GAMMOD and GAMPL Procedures	GAM Procedure
Use low-rank regression splines for smoothers	Uses smoothing splines and loess for smoothers
Use performance iteration or outer iterations	Uses backfitting to fit models
Search for smoothing parameters by optimizing global criteria	Searches for smoothing parameters by fitting splines that have fixed degrees of freedom
Analyze large to massive data	Analyzes moderate to large data
Run in single-machine or distributed mode	Runs in single-machine mode
Use all cores and concurrent threads	Is single threaded

The GAMMOD procedure and the GENSELECT procedure use the log link function as the default for the gamma and the inverse Gaussian distributions. The GAMPL procedure uses the reciprocal link function as the default for the gamma distribution, and it uses the reciprocal squared link function as the default for the inverse Gaussian distribution.

Building Proportional Hazards Regression Models with the PHSELECT Procedure

Time-to-event models predict the probability that the lifetime of a subject exceeds t—denoted as the survivor function $S(t)$—from lifetime data that are incomplete because of censoring. These models are broadly applicable to data that range from patient survival times in medical research to customer lifetimes in business applications where turnover is a concern.

The PHSELECT procedure fits and builds Cox proportional hazards models. These models are semiparametric; they assume a linear parametric form for the effects of the predictors, but they do not require a parametric form for the

underlying survivor function. The survival time of each member of a population is assumed to follow its own hazard function, $\lambda_i(t)$, which is the instantaneous risk that an event will occur at time t and is expressed as

$$\lambda_i(t) = \lambda_0(t) \exp(\beta_1 x_{i1} + \cdots + \beta_p x_{ip}) \quad i = 1, \ldots, n$$

The function $\lambda_0(t)$ is called the baseline hazard function. The predictors $x_{i1}, \ldots, x_{ip}$ represent main effects that consist of continuous or classification variables and can include interaction effects or constructed effects of these variables.

The PHSELECT procedure uses the partial likelihood approach of Cox (1972, 1975) to estimate the coefficients $\beta_1, \ldots, \beta_p$. These estimates can then be used to predict $S(t)$ at specified times t for a new subject with specified covariates. The prediction is based on Breslow's estimator of the baseline cumulative hazard rate (Breslow 1974).

Like the other procedures discussed in this paper that build regression models, the PHSELECT procedure offers extensive capabilities for effect selection, including the backward, fast backward, forward, stepwise, and lasso methods, together with a wide variety of selection and stopping criteria for customizing the selection. The PHSELECT procedure also provides Cox regression diagnostics that are conditional on the selected model.

Example: Predicting the Retention of Insurance Customers

A health insurance company carries out a study of younger customers who are experiencing life changes. The goals are to identify factors that explain the risk of switching to a different insurance plan, and to predict the probability of retaining a customer from one to five years in the future.

The customer lifetimes are saved in a CAS table named **Customers**. Each observation provides information about a customer. The variables available for building a Cox regression model are **Time** (the customer lifetime, measured in months), **Status** (the censoring indicator variable), and the candidate predictors shown in Table 8.

Table 8 Candidate Predictors for Customer Lifetime Model

Predictor	Type
Age	Continuous
Area	Classification with levels `'Urban'`, `'Rural'`
CurrentPlan	Classification with levels `'A'`, `'B'`
Education	Continuous
Income	Continuous
LifeChange	Classification with levels `'Married'`, `'New Job'`, `'Child'`, `'None'`
Satisfaction	Classification with levels `'Excellent'`, `'Good'`, `'Poor'`
Smoking	Classification with levels `'Yes'`, `'No'`, `'Quit'`

The following statements build a Cox model by using the forward selection method:

```
proc phselect data=mycas.Customers;
   class Area(ref='Urban') CurrentPlan(ref='B') LifeChange(ref='None')
         Satisfaction(ref='Poor') Smoking (ref='No') / param=ref;
   model Time*Status(0) = Age Area CurrentPlan Education Income LifeChange
                          Satisfaction Smoking;
   selection method=forward(select=sbc stop=sbc) plots=all ;
   code file='ScoreCode.sas' timepoint=12 24 36 48 60;
run;
```

The CLASS statement specifies that **Area**, **CurrentPlan**, **LifeChange**, **Satisfaction**, and **Smoking** are classification variables, and the PARAM= option specifies reference cell coding for these variables. The REF= option specifies the reference level of each variable. Various parameterizations are available, including effect coding and less-than-full-rank reference (GLM) coding, which is the default.

The SELECTION statement requests the forward method, and the SELECT= option requests that the selected model minimize the Schwarz Bayesian criterion (SBC). By default, all the design columns that correspond to a classification variable enter the model together. If you specify the SPLIT option in the CLASS statement, the design columns will enter independently.

The CODE statement writes SAS DATA step code for computing predicted survival probabilities to a file named *ScoreCode.sas*. The TIMEPOINT= option specifies time points (in months) at which survival probabilities are to be predicted from the selected model.

Effects that provide the best improvement in SBC are added until no more effects can improve the criterion. As shown in Figure 27, the minimum value of SBC is reached when **Area** enters the model.

Figure 27 Model Selection Summary

The PHSELECT Procedure

Selection Details

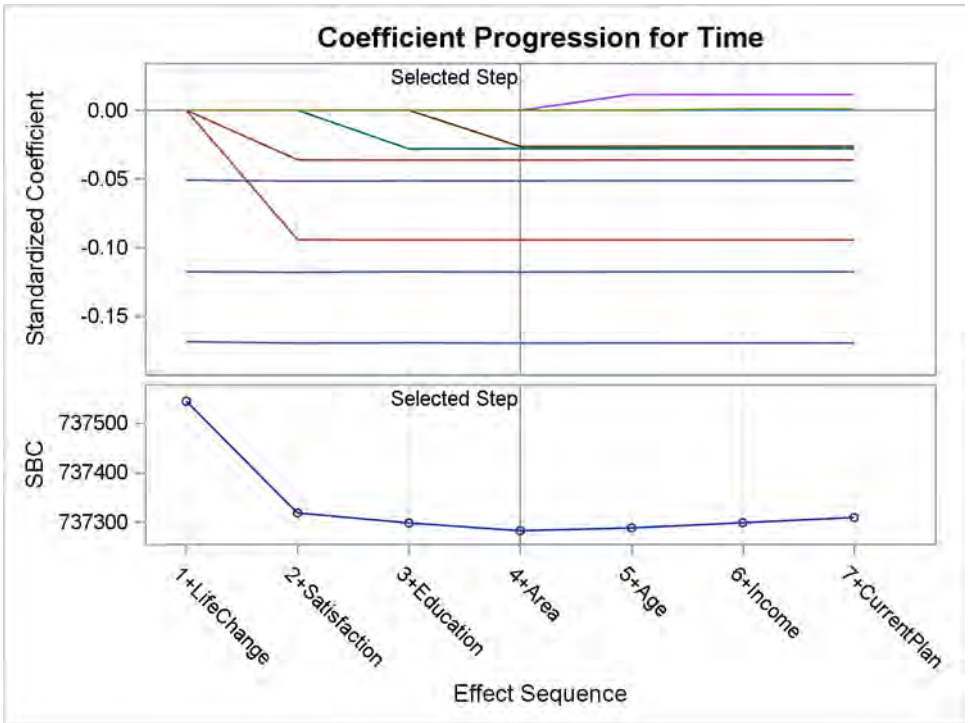
Selection Summary

Step	Effect Entered	Number Effects In	SBC
1	LifeChange	1	737543.656
2	Satisfaction	2	737318.101
3	Education	3	737297.864
4	Area	4	737282.214*
5	Age	5	737287.812
6	Income	6	737298.311
7	CurrentPlan	7	737308.843

*** Optimal Value Of Criterion**

The plot in Figure 28 visualizes the selection process. The forward method selects a model with seven parameters that involve one continuous variable and three classification variables.

Figure 28 Coefficient Progression with Forward Selection



The parameter estimates are shown in Figure 29. Effects that have negative coefficients decrease the predicted hazard function at time t and therefore increase the predicted survival time $\hat{S}(t \mid x_1, \ldots, x_p)$ because

$$\hat{S}(t \mid x_1, \ldots, x_p) = \exp(-\hat{\Lambda}_0(t) \exp(\hat{\beta}_1 x_1 + \cdots + \hat{\beta}_p x_p))$$

where $\hat{\Lambda}_0(t)$ is the Breslow estimator of the cumulative baseline hazard rate $\Lambda_0(t) = \int_0^t \lambda_0(u)du$.

Figure 29 Parameter Estimates for Selected Model

Parameter Estimates

Parameter	DF	Estimate	Standard Error	Chi-Square	Pr > ChiSq
Area Rural	1	-0.052843	0.010327	26.1831	<.0001
Education	1	-0.008096	0.001496	29.2890	<.0001
LifeChange Child	1	-0.391075	0.014684	709.2694	<.0001
LifeChange Married	1	-0.273049	0.014655	347.1629	<.0001
LifeChange New Job	1	-0.119519	0.014620	66.8349	<.0001
Satisfaction Excellent	1	-0.200931	0.012690	250.7106	<.0001
Satisfaction Good	1	-0.077127	0.012630	37.2929	<.0001

Predicting Survival Probabilities

The following statements use the generated code to compute retention probabilities for five new customers whose covariates are saved in a data set named **NewCustomers**:

```
data Predictions;
   set NewCustomers;
   %include 'ScoreCode.sas';
run;
```

Figure 30 lists the predicted retention probabilities in **NewCustomers**.

Figure 30 Listing of **Scores**

Area	Years of Education	Life Change	Satisfaction	Retention Probability at 1 Year	Retention Probability at 2 Years	Retention Probability at 3 Years	Retention Probability at 4 Years	Retention Probability at 5 Years
Rural	13	New Job	Poor	0.671	0.455	0.315	0.221	0.155
Urban	14	Married	Good	0.718	0.520	0.383	0.285	0.212
Rural	8	New Job	Excellent	0.711	0.512	0.373	0.276	0.204
Urban	11	New Job	Poor	0.652	0.431	0.290	0.198	0.136
Rural	17	Child	Excellent	0.786	0.622	0.498	0.402	0.324

Comparison with the PHREG Procedure

Compared with the PHREG procedure in SAS/STAT, the PHSELECT procedure provides many more features for model selection, including the lasso method and options for selection and stopping that are based on information criteria and validation. The PHSELECT procedure also enables you to partition the data into logical subsets for training, validation, and testing. On the other hand, the PHREG procedure provides more flexibility for fitting the Cox model, such as the ability to specify time-dependent covariates, and more inferential methods, such as the following:

- extensive postfitting analyses of regression parameters
- hazard ratios for any variable in the model at customized settings, and confidence limits for hazard ratios
- Schemper-Henderson and concordance statistics for model assessment
- time-dependent ROC curves for model assessment

The PHREG procedure also enables you to analyze competing risks and frailty models, and to perform Bayesian analysis of Cox models, piecewise exponential models, and frailty models.

Using the LOGSELECT Procedure with Discrete Time

In some applications of proportional hazards regression, events occur at regular, discrete points in time, or ties occur because continuous event times are grouped into intervals. Many ties present a problem for the Breslow partial likelihood approach that is implemented by the PHSELECT procedure, but this problem can be circumvented by using logistic regression and maximum likelihood methods that are implemented in the LOGSELECT procedure.

As explained by Allison (2010, chap. 7), the maximum likelihood approach treats the survival history of each individual as a series of distinct observations, one at each time unit. The observations are pooled, and a logistic regression model is used to predict the probability P_{it} that individual i has an event at time t. This model is of the form

$$\log\left(\frac{P_{it}}{1-P_{it}}\right) = \alpha_t + \beta_1 x_{it1} + \cdots + \beta_p x_{itp}, \quad i = 1,\ldots,n, \quad t = 1,2,\ldots$$

The maximum likelihood approach provides estimates of the effect of time on the hazard function, and it handles time-dependent covariates in a natural fashion. The total number of observations obtained by expanding individual survival histories can be large, especially if the time units are small, but keep in mind that statistical procedures in SAS Viya are designed to accommodate large data.

Summary of Benefits

Table 9 lists key benefits of methods that the SAS Viya procedures implement for regression modeling.

Table 9 Benefits of Methods for Regression Modeling

Method	Benefit	SAS Viya Procedure
Stepwise methods that use modern information and validation criteria for selection and stopping	Improved predictive accuracy and interpretability	GENSELECT, LOGSELECT, PHSELECT, QTRSELECT, REGSELECT
Lasso methods	Improved predictive accuracy and interpretability	GENSELECT, LOGSELECT, PHSELECT, REGSELECT
Data partitioning into training, validation, and testing roles	Improved predictive accuracy	GENSELECT, LOGSELECT, PHSELECT, QTRSELECT, REGSELECT
Effect selection for generalized linear models	Models for responses with a variety of discrete and continuous distributions	GENSELECT
Effect selection for time-to-event models	Prediction of survival probabilities by using censored lifetime data	PHSELECT
Effect selection for quantile regression	Models for conditional quantiles of a continuous response distribution	QTRSELECT
Generalized additive models with penalization	Flexibility for modeling complex, unknown dependency relationships	GAMMOD

No one method consistently outperforms the others. Furthermore, all the methods involve choices of tuning parameters and optimization techniques for which there are no universally best defaults. In order to decide which methods are appropriate for your work, you should understand their assumptions and characteristics; these are explained in the "Shared Concepts" and procedure chapters in *SAS Visual Statistics 8.2: Procedures*. You should also experiment with different combinations of options to learn about their behavior.

The ability to score future data is an essential aspect of predictive modeling. All the procedures discussed in this paper compute predicted values for observations in which only the response variable is missing (or only the censoring variable is missing in the case of the PHSELECT procedure). The values are saved in a CAS table that is created when you specify the OUTPUT statement. Except for PROC GAMMOD, all the procedures provide a CODE statement, which writes SAS DATA step code for computing predicted values to a file, to a catalog entry, or to a CAS table.

The regression modeling procedures in SAS Viya offer two general improvements in functionality over the SAS/STAT high-performance procedures that they succeed:

- capability for producing graphs with the PLOTS option
- capability for constructing special collections of columns for design matrices with the EFFECT statement (for example, you can use the EFFECT statement to specify polynomial and spline effects).

REFERENCES

Allison, P. D. (2010). *Survival Analysis Using the SAS System: A Practical Guide*. 2nd ed. Cary, NC: SAS Institute Inc.

Breslow, N. E. (1974). "Covariance Analysis of Censored Survival Data." *Biometrics* 30:89–99.

Cox, D. R. (1972). "Regression Models and Life-Tables." *Journal of the Royal Statistical Society, Series B* 34:187–220. With discussion.

Cox, D. R. (1975). "Partial Likelihood." *Biometrika* 62:269–276.

Efron, B., Hastie, T. J., Johnstone, I. M., and Tibshirani, R. (2004). "Least Angle Regression." *Annals of Statistics* 32:407–499. With discussion.

Frigo, C., and Osterloo, K. (2016). "exSPLINE That: Explaining Geographic Variation in Insurance Pricing." In *Proceedings of the SAS Global Forum 2016 Conference*. Cary, NC: SAS Institute Inc. http://support.sas.com/resources/papers/proceedings16/8441-2016.pdf.

Gu, C., and Wahba, G. (1991). "Minimizing GCV/GML Scores with Multiple Smoothing Parameters via the Newton Method." *SIAM Journal on Scientific Computing* 12:383–398.

Koenker, R., and Bassett, G. W. (1978). "Regression Quantiles." *Econometrica* 46:33–50.

Rodriguez, R. N. (2016). "Statistical Model Building for Large, Complex Data: Five New Directions in SAS/STAT Software." In *Proceedings of the SAS Global Forum 2016 Conference*. Cary, NC: SAS Institute Inc. http://support.sas.com/resources/papers/proceedings16/SAS4900-2016.pdf.

Rodriguez, R. N., and Yao, Y. (2017). "Five Things You Should Know about Quantile Regression." In *Proceedings of the SAS Global Forum 2017 Conference*. Cary, NC: SAS Institute Inc. http://support.sas.com/resources/papers/proceedings17/SAS0525-2017.pdf.

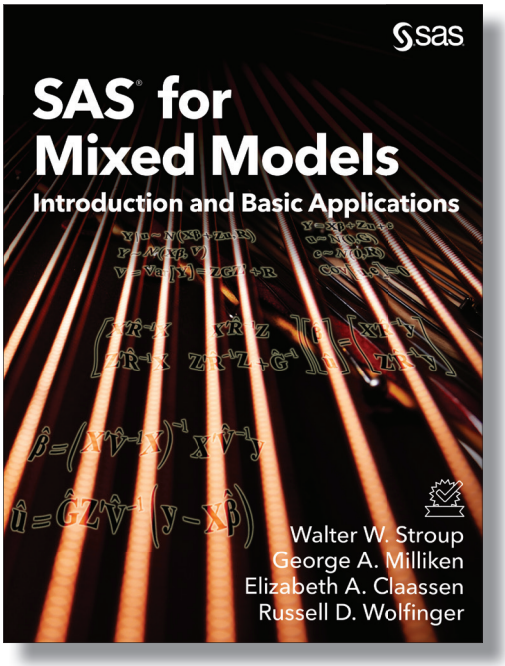
Wood, S. (2003). "Thin Plate Regression Splines." *Journal of the Royal Statistical Society, Series B* 65:95–114.

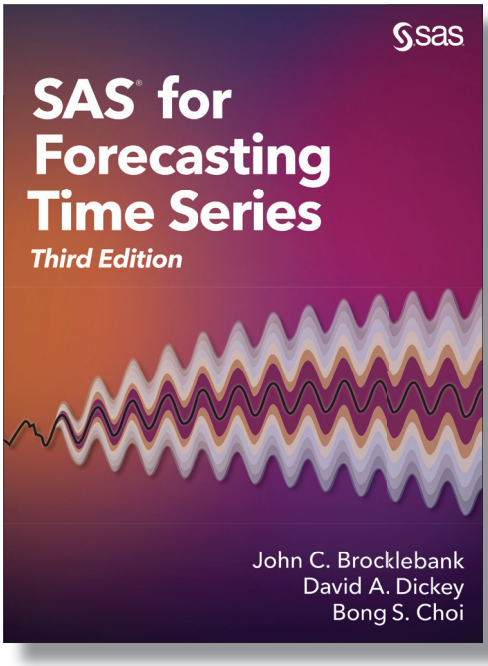
Wood, S. (2006). *Generalized Additive Models*. Boca Raton, FL: Chapman & Hall/CRC.

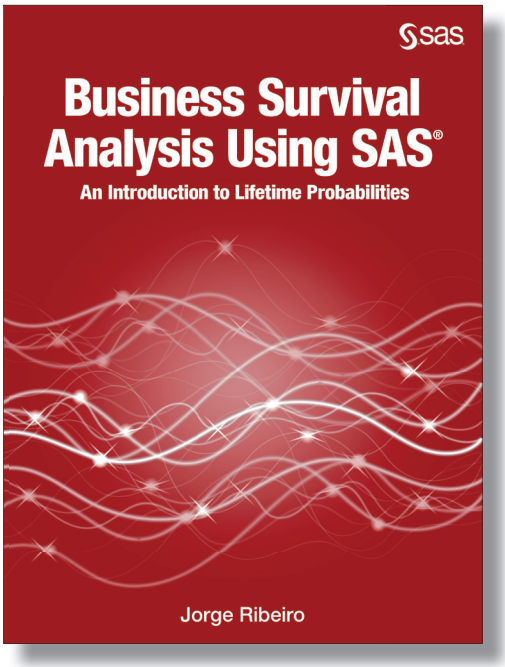
Acknowledgments

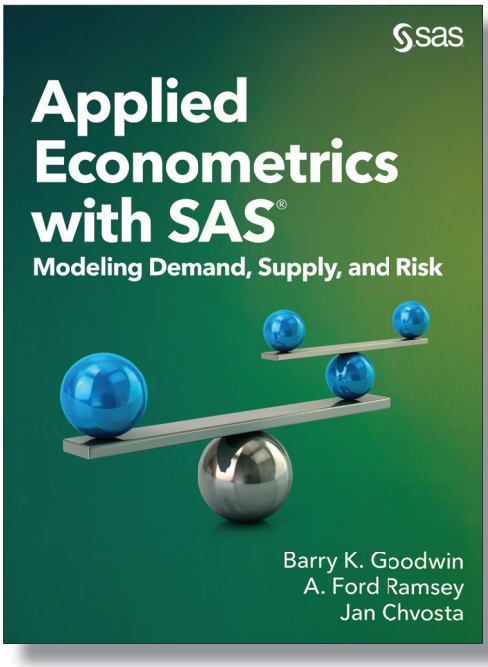
The following statistical software developers at SAS contributed to this paper: Robert Cohen, Bob Derr, Gordon Johnston, Ying So, and Yonggang Yao. The authors also thank Anne Baxter for editorial assistance.

Check out these related books in the SAS® bookstore:









For 20% off these e-books, visit **sas.com\books** and use the code **WITHSAS20**

sas.com/books
for additional books and resources.

sas
THE POWER TO KNOW®